3. Framework for Comparative study

The level of suitability of an autonomous vehicle simulator to a specific user various diversly depending on various factor. Some of the factor constitute to this issue are the application, availability of sensors, integration of real-world components such as vehicle, maps…, computation resource… In this proposed method, 69 such parameters are identified and those parameters are used to devise a single score for a simulator which can be used to compare.

3.1 Comparison Parameters

The considered parameters are further classified into 6 broad classifications. They are Sensors, Actors, Environment, Vehicle, Framework and Algorithm. Each parameter is measured by a score between 0 to 1. In case of binary classification type for example availability of a feature, this corresponding parameter gets score 0 if it’s not available and 1 if it’s available

3.1.1 Sensors

Sensors are the tool used to perceive the environment in the vehicle is driving on. It plays a crucial role in driverless vehicle as it provides input to the algorithm and the action is taken based on that. Various types of sensors are used in vehicles and in autonomous vehicles it’s very common to use multiple sensors at a time. It’s also important to model and integrate those sensors within the simulator which the real vehicle uses. This part examines various aspects of the sensors that can be used in autonomous vehicle simulator

3.1.1.1 Ready to use sensors

Most of the simulators provides some default sensors which are ready to use. A simulator will be assigned a score of 1 if it provides the below mentioned sensors else it will be assigned 0.

1. RGB camera

RGB Camera is a device which maps the 3d environment in its field of view as a 3-channel 2D image (Red Green Blue). Figure 11 illustrates the example data of a RGB camera mounted on a car. These images can be used to understand the surroundings such as traffic lights, sign boards, pedestrian crossing…

1. Depth Camera

Depth camera also captures a single channel image in which depth information is embedded. Each pixel in this image measures the distance between the mounting point of the camera and object in the 3d environment to which the pixel referring to. Figure 12 illustrates the example data of a Depth Camera mounted on a car. This data is often used to measure the distance between the vehicle and other fellow vehicles.

1. 3D Lidar

Unlike the other 2 other cameras 3D lidar maps the environment in 3D point cloud. Position of each voxel refers to a point on the object in the surrounding. This sensor projects laser beam and calculate the position based on time of flight. Figure 13 illustrates the isometric view of a point cloud measured by 3D lidar. However, the energy laser beam can be interrupted by various factors like fog, texture of the object… These sensors are sometimes attached to a rotary motor to read 360 degrees around the device.

1. Radar

This sensor similar to 3D lidar, which maps the environment to 3d point cloud. However this sensor uses EM (electro-magnetic) waves as an alternative to Laser beam. This wave has various advantages such as traveling long distances, less distortion to external factors. Both 3D lidar and Radar are used to read the surroundings

1. IMU sensor

Inertial Measurement unit (IMU) sensor is that measure the specific forces like acceleration, angular velocity. It reports the information about orientation and change in its position. This unit is composed of sensing elements such as accelerometer and gyroscope. These sensors help determine the vehicle's acceleration, changes in direction, and rotation

1. Semantic information

Semantic Information is a not an actual sensor that is directly used in autonomous vehicles. It is an ability of a simulator to provide sematic information of every object within the environment. The semantic information could have class or name of the object such as trees, dogs, child, women… with the coordinates of the bounding box which encapsulates the object. Figure 14 illustrates the environment with bounding boxes and class label of the objects within it. This information could be useful in some training algorithms such a reinforcement learning to provide rewards and penalties.

1. Force Impact sensor

It is a sensor which can be used to measure the force experienced by the vehicle during an impact with another object. This measure directly corelates to the damage occurred to the vehicle and the object. This measure could also be used to provide weighted rewards and penalties

1. Lane detection or infringement sensor

Like Semantic information, this is also not an actual sensor but provides information about the driving lanes. This feature is an ability of a simulator to provide the information about the boundaries of the lane and level of infringement (percentage of body of the ego vehicle infringing the lane). This method is used in various level 2 autonomous vehicle which uses ADAS.

1. Fuel/ Battery sensor

This sensor monitors the information about the fuel or power consumed by the vehicle. The self-driving vehicle should be efficient and researchers around the world are constantly working to improve the fuel efficiency. In such cases, it’s important for a simulator to provide such information.

1. Vehicle speed sensor

This sensor is a common sensor which can be found on almost all the vehicle. This measures the velocity of the vehicle. This measure is crucial especially when driving in areas with speed limits such as city roads, warehouses…

1. GPS

Global positioning system (GPS) is a geo location sensor which can be used to locate and navigate. Its also most common sensor found in the commercial vehicles for path planning. This sensor crucial both on road and in house driving.

In terms of simulator, the simulator should able to provide information about the coordinates of the vehicle and its destination with respect to its global boundaries. It is also important that it should follow a same scale throughout the simulator.

1. Visibility sensor

The driving visibility can be impacted by various environmental cause such as fog, smog, pollution etc… Visibility is a measure which defines the maximum distance at which an object can be clearly seen. This measure plays an important role in driving. It’s also important for a simulator to provide this information especially when the simulation environment mimics the foggy surroundings. This measure is usually measured in meters.

1. Any other vehicle sensor related to vehicle dynamics

A typical car can have numerous sensors which can measure various properties of the car such as tyre pressure, parking aid, engine rpm… However not all these vehicle sensors are usually used in autonomous vehicle research. An ideal simulator should provide an infrastructure to model any of these vehicle sensors whenever required by the user. This parameter quantifies the ability of a simulator to provide facility to define a sensor and provide its data related to the ego vehicles.

1. Any other Environment sensor related to perception

This parameter corresponds to ability of a defining a sensor by a user that can measure properties of an environment. These user defined sensors could provide data related to the surroundings of the car such as no of people / vehicle in the car’s field of perception, audio sensor to recognize priority vehicles such as fire truck.

1. Any Macroscopic global sensor

This is also a user defined sensor which provides information about the simulation world. This sensor could be modelled to provide information not only about the local area near to the vehicle but also some global information such as traffic in a certain part of the map, possible weather condition in future, which can be used for path planning.

3.1.1.2 Multiple synchronous sensor

In most of the cases, multiple sensors are used in autonomous vehicle simultaneously. The data from multiple sensors provide more reliable information about the environment rather than one from the single sensor and it can be used as backup in case any of the sensor fails. An AV simulator should handle multiple sensor model and provide its data. It is important that the sensors are synchronized both in data and time, which means the data simulated by multiple sensors at a time should refer to a same timestep and relative to each other. This parameter corresponds to the ability of a simulator to simulate synchronized data.

3.1.1.3 User defined position and orientation of sensor.

The placement of a sensor in a vehicle plays a crucial role in the useability of that sensor. The mounting point and the orientation of a sensor are designed by engineers. In most of the cases the it varies especially when using multiple sensors where sensor blind spot should be avoided. This parameter is a check for an ability of a simulator in which the user can define the location and orientation of the sensor with respect to vehicle.

3.1.1.4 Precision of the sensor readings

The precision of a sensor can be defined as the smallest change in measurement which the sensor can sense. This is be applied for any type of sensor and its measurement. The precision by the user varies depending upon the application. A versatile simulator should accommodate different levels of precision and it should be configurable by the user. This parameter confirms the ability of a simulator in which precision of the simulators can be configured

3.1.1.4 Real time sensor Models

Many Vehicles manufactures and develop their sensors in house with unique features. Digital models of these devices could be used as sensors within the simulator which make it useful for the user to retain those features in the simulator. Moreover, there are many popular and frequently sensor available in the market which can be used in vehicle. This parameter checks the ability of a simulator to import real time sensors’ digital models and use it within the simulator.

3.1.1.5 User defined sensors and observers

In the growing field of autonomous driving, lot of new sensors are being developed and the performance of existing sensors are updated to better understand the environment. Moreover, Within the simulator a user can be interested in and need to monitor any data of his interest for their application. A simulator should facilitate a user to define an observer which monitors the data or model a sensor. This parameter ensures this property of user defined sensors and observers.

3.1.2 Actors

Actors are the different objects that makeup the environment within the simulator. These actors can be broadly classified into stationary and non- stationary actors. Stationary actors are the objects which position stays the same, some examples of stationary actors are Buildings, trees, traffic signs, roads, parked vehicles… on the other hand non stationary objects changes it position during the course of simulation and some examples are People, animals, fellow cars. The actor are the constructors of the environment. This part explains various parameter of actors in a simulator that can be used to compare the simulators. Most of the parameter in this section will be scored 1 if a simulator possesses that feature and 0 if not.

3.1.2.1 Geometry

a) Pre defined of scenarios

There are various types of environments in which a vehicle can drive such as rural, urban, industrial… Each type of environment possesses unique features and actors. For example, it less probable to see a child in an industrial environment whereas very likely to see a cargo truck. A simulator could have presets of environments with reasonable actors and this ability of possessing pre-defined set of actors is checked by this parameter

b) Pre defined models for actors

This parameter checks the ability of the simulator to provide access to some pre-defined digital assets (actors) under various categories. For example, a simulator should able to provide 3d models of Buildings, vehicles, person, traffic signs which are meticulously designed and ready to deploy and use.

c) User defined Actors

Apart from the pre-defined libraries of 3d models of actors, a simulator should facilitate a user to import 3d models of actors within the simulator. Moreover, the imported models should follow a common scale which the simulator follows. This can be useful in some unusual cases such as Golf field where a user should able to import 3d models of golf cart and players

d) No of actors and spawning location

The nature of an environment also depends on no of actors within the environment. This measure should be changeable by the user such that the nature of environment can be configurable. For example, a traffic in a particular road can be defined by no of vehicles and people commuting in that road. Moreover, the location of the actors should also be configurable for example location of Traffic light pole. This parameter measures the ability of a simulator to facilitate user to define the number and location of actors.

e) Recreate real time models

This parameter ensures the usage of models of real time actors within the simulator. The simulator should posses real time 3d models of actors such as cars, buildings… and their physical properties like geometry, texture, should be maintained. This helps the users who intended to work with real time vehicles.

f) Path/ destination of actors

The moving trajectory of non-stationary actors should be configurable and the actors should respawn in a new location when the actor moves out of the environment boundaries. This ability of defining paths and destination of non-stationary object is checked by this parameter.

g) Context aware spawning

The spawning/ respawning of actors should be reasonable and context aware. It’s more sensible to spawn more ambulance near hospital than any other location. This type of intelligent spawning is marked by this parameter.

3.1.2.2 Behaviour

The Behaviour refers to the rules for actions taken by the actors and their feedback to actions taken by other actors and change in environment.

1. Pre-defined Rules

Some of the actor’s behaviour can be predefined and regulated. The Simulator coud define certain set of rules for the actors’ behaviour. For example, Traffic rules of certain countries can be pre-defined and ready to use

1. User defined rules

A simulator should facilitate the user to define certain behaviour pattern for the actors within the simulator in addition to pre-defined rules. This can provide more freedom for customization for the user. This parameter evaluates this feature of user defined rule within a simulator.

1. Distinguishable behavioural pattern

For a simulator, to reproduce real environment, it should distinguish the behavioural pattern within the actors. For example, the behavioural pattern of an adult and a kid and a dog walking a pavement differs a lot. This parameter is given score 1 if a simulator can simulate this phenomenon

1. Sensitivity

Sensitivity refers to feedback or counter action taken to the action taken by fellow actor. The fellow actor could be either the ego vehicle or any other actor within the environment. An example for this feature could be an adult refrain to cross the road in response to the horn honked by a car. A score of 0.5 is given if the actors within the simulator respond only to ego vehicle and 1 is given if the actor responds to all other fellow actors and 0 if not responds

1. Level of aggressiveness in driving

This feature appeals particularly to the vehicle within the simulator. The driving pattern varies widely among the people. A simulator should facilitate a user to define the level of aggressiveness using some metrics like violations per km driven. This parameter of defining behavioural pattern in driving can help the user to train and test in different driving condition

1. Reproducibility

Certain random behaviour of the actors can give unique response from the ego vehicle and user should able to reproduce the same action from the actor to further study the case. In this case a simulator should able to reproduce a set of random action performed by actors again this method is called seeding and the parameter refers to the seeding feature of the simulator.

3.1.3 Environment

The Environment in the simulator describes the Weather, Terrains, Maps… that make up the conditions where the car drives. In this part various parameters of environment that distinguishes the performance of the simulator will be discussed.

3.1.3.1 Topography

1. Pre-defined libraries

This parameter constitutes the ability of a simulator which can possess some of pre designed environment with diverse maps and terrain. In addition to that it could offer diverse weather condition and various time of the day. A simulator will score 0.5 if it satisfies either one of the conditions and 1 if both.

1. User defined environment

The simulator could offer some tools for the user to define their own maps and various weather conditions. This makes the simulator efficient in terms of customization and adaptability.

1. Real world maps

The replication of real world maps within the simulator is crucial especially when developing on road commercial autonomous vehicles as it could directly reflect the intricate design of road system. A simulator will be scored 1 if it facilitate the user to import or use the maps and other associated features into the simulator

1. Photorealism

Photorealism is the concept how close an environment in the simulator resembles the real world in terms of appearance. It is often subjective and very difficult to define a scoring system that is accepted by everyone. Therefore, the score for this parameter is defined by the user through visual inspection in the scale of 0 to 1. The simulator that can render scene more close reality will get higher score and vice versa

1. Variable resolution

The intricate details of the rendered environment can be varied by increased resolution. The lower resolution can consume less computational resource at the cost of information loss and vice versa. This resolution should to changeable and this parameter corresponds to this ability.

1. Context- aware Map planning

The Map generated by the simulator should posses some contextual meaning and the actors and terrain should be in accordance with context. For example, the location of buildings, roads, traffic lights should be co-related. This feature scores 1 in this parameter.

1. User defined scenario

In some special cases, the user might wish to generate some scenarios like a road construction, fallen tree in the mid of the road. A simulator to facilitate the user to generate these cases and this ability is marked by this parameter

1. Material

The materials that make up the environment can have various optical properties which can affect the data of sensors like Lidar and Camera. For example, some Lidar find it difficult to detect glass walls as the laser can pass through it instead of reflecting. This parameter marks the ability of the simulator which allows the user to define material of the objects in environment such as buildings

1. Variable level of sun shades

The shades of the sun vary a lot with respect to time of the day, weather and it can impact the sensors reading of the sensor. This parameter refers to the ability of the simulator which allows the user to change the

1. Special optical features

In real time various special optical feature such as sun glare, Mirage, dust, fog that can affect the visibility of the driver, for driverless vehicle it affect the sensor data and cause impact in it subsequent action. Therefore, it’s important to model these features in the simulator.

1. Data visualization and analytical tools

Visualization of the environment during training and testing provides various insights to the user. The simulator should able to display various data and analysis during the process. This parameter will be scored 1 if a simulator can facilitate a user to define a data visualization and monitoring tool of their interest. The data could be any sensory data mounted on the car or data related to algorithm and training or bounding boxes, semantic truth...

1. Rendering

During testing, the rendering of the scene aids the user to analyse the performance of algorithm. The quality of the rendering such as view angle, frames per second should be changeable by the user. This feature will account for the configurable rendering of display and scored 1 if its available.

1. Recording

Recording of the training can help the user to analyse the performance of the training process and can playback for future analysis. This parameter accounts for the recording of process along with associated data within the simulator

3.1.4 ego Vehicle

The Agent vehicle is the vehicle which is trained or tested in the simulator to drive autonomously. This section explores various parameters that evaluates the simulator based on the ego vehicle.

1. Vehicle geometry

The vehicle geometry refers to the dimensions of the car in the 3d environment. In other words, the bounding box that encapsulates the car in 3d. This parameter refers to ability of a simulator to facilitate the user to configure the geometry of the ego vehicle

1. Wheel Geometry

This parameter refers to feature of changing the parameters of suspension and steering system of the ego vehicle within the simulator. In addition, the car has to follow the physics in accordance with the wheel geometry. This feature is crucial for automaker as it can impact the driving performance

1. Real world models

Carmaker develop the digital models of their vehicle which can be used across various computation platforms. This parameter represents the feature of importing and using the 3d models of the real-world vehicles.

1. Real world physics

This parameter checks the ability of a simulator to simulate various physical parameters associated with the driving. This includes Wheel slip, damage due to crash and inertial behaviour of the vehicle. Replication of these feature are crucial especially when developing an algorithm for real world on-road autonomous vehicle.

1. Input commands

The driving algorithm outputs the command for the vehicle such as value for acceleration, steering angle and direction based on the observation. The input command is important as it drives the vehicle and it varies from algorithm to algorithm. Input for every component in the ego vehicle should be configurable, so that the users can use any component of their choice. This parameter ensures the ability of the simulator to configure the input commands to vehicle

1. Other components

Few other components such as Head lights and horn can have minor impact in driving and within a simulator the intensity of these components should be configurable. This parameter scores for this feature.

3.1.5 Framework

3.1.5.1 Cross platform compatibility

This parameter scores for various operating system on which the simulator software runs. It is important for a simulator to run across various platform. Some of the popular and relevant Operating system is listed below, the simulator will score 1 if it is supported in that OS

1. Linux
2. ROS
3. Windows
4. MacOS
5. Any other Operating system

3.1.5.2 Opensource

Though many features within the simulator is customizable, users with special needs want the simulator to be tailored for their use case. In that case, a simulator has to entirely or part of it should be open source enabling the user to import the part of the simulator such as simulation background as a plugin and develop a software on top of that. This parameter will be scored 1 if the simulator is open source and enables the user to develop own features on top the simulator.

3.1.5.3 Scripting language

Scripting language is programming language which is used by the user to communicate with the simulator. The communication can include configuring some featuring, setting up I/O pipeline… A versatile simulator should support various scripting language. Some of the popular and relevant Scripting language is listed below, the simulator will score 1 if it is supported in that Scripting language.

1. Python
2. C++
3. C
4. R
5. Java
6. Other programming language

3.1.5.4 Scalability

In many cases, users don’t require all the feature which the simulator offers, in that case, a simulator can offer various individual versions of the simulator with unique feature and a full version with all the features, this feature can take make the software light for the user and this feature is marked by this parameter

3.1.5.5 Computational resource

Simulator consumes huge amount of computational resource and often require expensive device to work with. This parameter evaluates the computational resource based on electrical energy consumed by a baseline system under maximum operation of a simulator. In this method the baseline system is configured with following configuration:

The maximum electrical energy that this baseline system can consume is calculated as approximately 750 watts. The score for the computational resource of a simulator is defined as

1 – (power consumed during operation / maximum power consumption of the base line system). Any simulator which consumes more than 750 watts is scored 0.

3.1.5.6 Community support

Community support is essential for any public software and this parameter identifies 3 streams of community support where each gets a score of 1 if available.

1. Structured documentation
2. Tutorials
3. Discussion forums

3.1.5.6 Distributed computing

Distributed computing is the process of running multiple instances of simulation on multiple different machines converging into a single coordinated and coherent simulation system. This reduces can improve the efficiency of simulation by decreasing the runtime and increasing the number of episodes for a given time. The parameter checks this ability of parallelization in computing.

3.1.6.1 Driving Algorithm

Driving algorithm is the model which processes the observation and predicts the input commands for ego vehicle. During training this algorithm is trained by different methods for better performance. This part explores various comparative parameter with respect to algorithm used in simulator

1. Data exchange

Mostly the algorithm is trained in a different computational kernel and the simulation engine is deployed as a plugin. The IO data of the ego vehicle is communicated to the base code from environment thorough communication channels such as APIs, and some communication bridge. This parameter will be scored 1 if there exist a pre-defined communication plugin.

1. Multithreading

Multithreading is the process of parallelizing the computation of simulation engine and training of algorithm using GPUs, which significantly decreases the computation time and an algorithm can be trained on more episodes for given time. This parameter ensures the availability of Parallelization of computation and algorithm within the simulator

1. Data Logging

Data logging is the process of collecting and storing the data generated during the simulation process which can be use to analyse in future. This parameter will be scored 1 if a simulator aids the users to log certain data of their interest.

1. Hardware in Loop (HIL)

Hardware in Loop (HIL) is the method of involving real time equipment such a sensor, vehicle controls… within the simulator to test the performance. This method appeals to OEMs who wish to test their hardware. This parameter will be checked if a simulator supports any hardware within its operation.

1. ML Libraries

Various popular machine learning libraries such as Tensorflow, Pytorch, Scikit facilitates training of Machine learning models. It’s easy for user if they can integrate such libraries within the simulator for training models. This parameter ensures the support of any ML libraries in the simulator

3.2 Methodology

This section explains how a single distinctive score can be calculated to a simulator based on the above-mentioned parameters. To derive a score based on user’s requirement and the capability of simulator 2 different sets of score is defined 1. Base score 2. User weight

3.2.1 Base score

This scores of the parameter are purely based on the simulator and it is constant of every user. A simulator will get a score 1 if it fulfils the criteria of parameter and 0 if not. Every simulator will be score for each parameter and this forms a base score vector of a particular simulator. Usually, the vectors are expected to assigned by the simulation software developers.

3.2.2 User weight

These value forms a vector similar to base score vector but the scores are purely given by the users. The score explains how important a parameter to a user from the scale of 0 to 1. The user is expected to assign a score for all the parameter and these scores forms the user weight vector

3.2.3 Final score

The final score which explains how suitable a set of simulators considered S = [S1, S2, S3…] Sn for a set of user A is calculated by the dot product of Base score vectors of simulators with the user weight vector of user U. This results in a vector which gives use ability score for each simulator. In this way a user can derive score for each simulator.

**CLEAN TEXT**

3. Framework for Comparative Study

The suitability of an autonomous vehicle simulator for a particular user varies significantly based on several factors. These factors include various aspects such as the intended application, sensor support availability, integration of real-world components like vehicles and maps, computational resources... Within this proposed method 69 such parameters are identified and these parameters collectively contribute to formulating a single score for each simulator. This score serves as a means for comparison between different simulators.

3.1 Comparison Parameters

The identified parameters are categorized into six broad classifications: Sensors, Actors, Environment, Vehicle, Framework, and Algorithm. Each parameter is evaluated using a scoring system ranging from 0 to 1 where 0 being worse and 1 is best. In cases of binary classification, where a feature's availability is assessed, the corresponding parameter receives a score of 0 if the feature is unavailable and 1 if it is available.

3.1.1 Sensors

Sensors serve as the tool to perceive the vehicle's surrounding environment. They play a crucial role in driverless vehicles by providing inputs to the algorithm, which determines subsequent actions of the vehicle. In autonomous vehicles, multiple sensor types are commonly utilized simultaneously. It is also important to accurately model and integrate these sensors within the simulator to mimic those employed in real vehicles. This section examines the various aspects of sensors that can be used within an autonomous vehicle simulator.

3.1.1.1 Default Sensors

The majority of simulators offer preexisting sensors that are readily available for use. A simulator will receive a score of 1 if it includes the specified sensors below; otherwise, it will be assigned a score of 0.

a) RGB Camera

An RGB camera functions by translating the 3D environment within its field of view into a 3-channel 2D image (Red, Green, Blue). Refer to Figure 11 for an example of data captured by an RGB camera mounted on a vehicle. These images are valuable for comprehending the surroundings, including aspects such as traffic lights, signage, and pedestrian crossings.

Figure 11: Image of vehicles surroundings captured by a RGB camera

b) Depth Camera

A depth camera captures a single-channel image embedding depth information. Each pixel in this image denotes the distance between the camera's mounting point and objects in the 3D environment it references. Figure 12 showcases an example dataset obtained by a Depth Camera fixed on a vehicle. This data is commonly utilized to measure distances between the vehicle and other nearby vehicles.

Figure 12: Depth Image of vehicles surroundings captured by a depth camera

c) 3D Lidar

Unlike traditional cameras, the 3D lidar creates a 3D point cloud representation of the environment. Each voxel's position corresponds to a point on objects in the surroundings. This sensor projects laser beams and calculates positions based on time of flight. Figure 13 illustrates an isometric view of a point cloud generated by 3D lidar. However, factors such as fog or object texture can distort the energy of the laser beam. These sensors are occasionally affixed to a rotary motor to achieve a 360-degree scan around the device.

Figure 13: Isometric view of point cloud projection of surrounding captured by 3D Lidar

d) Radar

Similar to the 3D lidar, the Radar sensor maps the environment into a 3D point cloud. However, unlike the 3D lidar, this sensor uses electromagnetic (EM) waves instead of laser beams. Radars takes the advantages of EM waves such as covering longer distances and experiencing less distortion from external factors. Both the 3D lidar and Radar are utilized to map the surrounding environment in 3D data.

e) IMU Sensor

The Inertial Measurement Unit (IMU) sensor measures specific forces like acceleration and angular velocity. It provides information about the orientation and changes in position of the vehicle on which it is mounted on. IMU consists of sensing elements such as accelerometers and gyroscopes which aids in determining the vehicle's acceleration, change in direction, and rotation.

f) Semantic Information

Semantic Information isn't a physical sensor used directly in autonomous vehicles. Rather, it denotes a simulator's ability to offer semantic information about every object within the environment. This information typically includes the class or name of objects, such as trees, dogs, children, women, along with the coordinates of bounding boxes encapsulating these objects. Refer to Figure 14 for an illustration of the environment with bounding boxes and class labels of the objects. This information can be valuable in certain training algorithms like reinforcement learning, facilitating the provision of rewards and penalties.

Figure 14: Semantic Information of various object in environment with bounding boxes

g) Force Impact Sensor

This sensor measures the force experienced by a vehicle during impact with another object. This measurement directly correlates with the damage caused by both the vehicle and the object. Moreover, it can be utilized to assign weighted rewards and penalties.

h) Lane Detection or Infringement Sensor

Similar to Semantic Information, this feature isn't a physical sensor but provides information about driving lanes. This sensor reflects a simulator's capability to provide details about lane boundaries and the extent of infringement, indicated as the percentage of the ego vehicle's body overlapping the lane.

i) Fuel/Battery Sensor

This sensor monitors information regarding fuel or power consumption by the vehicle. As self-driving vehicles aim for efficiency, researchers worldwide continually work to enhance fuel efficiency. Therefore, it's crucial for a simulator to offer such data.

j) Vehicle Speed Sensor

A widely used sensor present in nearly all vehicles, it measures the vehicle's velocity. This sensor is particularly important in areas with speed limits, such as city roads or warehouses within the simulator.

k) GPS

The Global Positioning System (GPS) is a geolocation sensor utilized for locating and navigation. It's among the most common sensors found in commercial vehicles, essential for path planning both on roads and within closed environments. Regarding simulators, it's vital for them to provide information about the coordinates of the vehicle and its destination within global boundaries with respect to the simulator scale.

l) Visibility Sensor

Visibility while driving can be affected by various environmental factors like fog, smog, or pollution. It is a measure of the maximum distance at which objects can be clearly seen, playing a critical role in safe driving. For simulators, providing this information becomes crucial, particularly when simulating foggy conditions. Typically measured in meters.

m) Other Vehicle Sensors related to Vehicle Dynamics

Conventional cars possess numerous sensors measuring various car properties such as tire pressure, parking aid, engine RPM, among others. However, not all these vehicle sensors are typically utilized in autonomous vehicle research. An ideal simulator should offer the infrastructure to model any of these vehicle sensors when needed, quantifying its ability to define and provide data related to ego vehicle.

n) Other Environment Sensors related to Perception

This parameter denotes the user's ability to define sensors measuring environmental properties. These user-defined sensors could offer data regarding the car's surroundings, like the number of people/vehicles within the car's field of perception. Additionally, sensors like audio recognition for identifying priority vehicles, such as ambulance and fire-fighting vehicles, fall under this category.

o) Macroscopic Global Sensor

Another user-defined sensor, this provides information about the simulation world, not just the local area near the vehicle. Such a sensor could furnish global details like traffic conditions in specific map sections or future weather forecasts, which can significantly aid in path planning.

3.1.1.2 Multiple Synchronous Sensors

In autonomous vehicles, multiple sensors are commonly used simultaneously to enhance reliability and provide backup in case of sensor failure. Simultaneous data from multiple sensors provide more dependable environmental information compared to a single sensor. An AV simulator's capability to handle multiple synchronized sensor models and provide their data becomes crucial. Ensuring synchronized data implies that data simulated by multiple sensors at a given time corresponds to the same time step and is coherent with each other.

3.1.1.3 User-Defined Sensor Position and Orientation

The placement and orientation of sensors within a vehicle significantly influence their application. AV Engineers meticulously design the mounting points and orientations, especially when multiple sensors are used to avoid sensor blind spots. This parameter evaluates a simulator's capacity to enable users to define sensor location and orientation with respect to the vehicle.

3.1.1.4 Precision of Sensor Readings

Sensor precision refers to the smallest measurable change in a sensor's readings. This attribute applies to various sensor types and their measurements, with precision varying based on application needs. An adaptable simulator should accommodate diverse levels of precision, allowing users to configure it for various sensor according to their requirements.

3.1.1.4 Real-Time Sensor Models

Some vehicle manufacturers develop unique in-house sensors with unique features. Simulators incorporating digital models of these sensors help users retain their specific functionalities within the simulation. Additionally, providing digital models of popular sensors readily available in the market becomes advantageous. This parameter evaluates a simulator's capability to import real-time digital models of sensors and integrate them into the simulation environment.

3.1.1.5 User-Defined Sensors and Observers

In the dynamic field of autonomous driving, new sensor technologies are frequently emerging, and existing sensors undergo performance updates to better understand the environment. Within a simulator, users might have specific data of interest for their application or need to monitor certain aspects. The simulator's capability to enable users to define observers to monitor data or model new sensors becomes important to cope up with evolving sensor needs and user-specific requirements.

3.1.2 Actors

Actors within the simulator constitute the diverse objects forming the environment. These actors categorize broadly into stationary and non-stationary, with stationary actors maintaining fixed positions (e.g., buildings, trees, traffic signs) and non-stationary actors altering their positions during simulation (e.g., people, animals, moving vehicles). This part explores criteria of various parameter related to actors that influences the comparison

3.1.2.1 Geometry

a) Predefined Scenarios

Environments for vehicle operation vary widely such as rural, urban, industrial… each possessing unique features and actors. For instance, encountering a child is less likely in an industrial setting compared to seeing a cargo truck. This parameter evaluates a simulator's capability to offer presets of environments with appropriate actors corresponding to various scenarios.

b) Predefined Models for Actors

This parameter evaluates the simulator's ability to offer access to meticulously designed digital assets (actors) across various categories such as buildings, vehicles, people, and traffic signs. These models are preconfigured across various scenarios and ready for deployment within the simulator environment.

c) User-Defined Actors

Apart from the preconfigured libraries of 3D models, the simulator should allow users to import custom 3D models of actors while maintaining a standardized scale within the simulator. This feature can aid user to tailor the simulator for their use case. For example, develop a simulator for driverless vehicle in golf field demands integration of 3D models of golf carts and players.

d) Number of Actors and Spawning Location

The nature of an environment is influenced by the number of actors within it. Users should have the ability to adjust the number of actors to configure the environment. Furthermore, the simulator should enable users to configure the locations of actors, such as defining the traffic density in a particular road or placing specific objects like traffic light poles.

e) Recreation of Real-Time Models

Simulators should possess real-time 3D models of actors such as cars, buildings while preserving their physical properties like geometry and textures. This feature appeals to users working with real-time vehicles or entities.

f) Path/Destination of Actors

Configurability of the moving trajectories for non-stationary actors, allowing users to define paths and destinations. Actors should also respawn in new locations when they move beyond environment boundaries. This parameter ensures the capability of a simulator providing flexibility in defining their movement patterns.

g) Context-Aware Spawning

Ensuring that the spawning or respawning of actors aligns with contextual reasoning. For instance, spawning more ambulances near hospitals demonstrates intelligent spawning based on context. This parameter evaluates the simulator's ability to spawn actors sensibly in relation to their surroundings.

3.1.2.2 Behaviour

Behaviour of the actor refers to the rules guiding actor actions, their responses to other actors, and environmental changes.

a) Predefined Rules

Simulators can define certain preset rules governing actor behaviour, such as traffic regulations in specific countries, readily available for use within the simulation.

b) User-Defined Rules

Allowing users to define specific behaviour patterns for actors within the simulator in addition to predefined rules, provides better customization options.

c) Distinguishable Behavioural Patterns

Simulators aiming to replicate real environments must simulate distinct behavioural patterns among actors. For example, distinguishing between the behaviour of an adult, a child, and a dog walking on a sidewalk. A simulator earns a score of 1 for this parameter if it is capable of accurately simulating these behavioural differences.

d) Sensitivity

Sensitivity pertains to an actor's responsiveness or reaction to actions taken by other actors, including the ego vehicle within the environment. A score of 0.5 is given if actors respond solely to the ego vehicle, while a score of 1 is granted if actors respond to all fellow actors. This parameter evaluates the simulator's sensitivity to fellow actors' actions.

e) Level of Aggressiveness in Driving

The driving pattern varies among various parts of the world. Simulators should allow users to define driving patterns' aggressiveness using metrics like violations per kilometer driven. This parameter enables users to train and test in various driving conditions by defining behavioural patterns.

f) Reproducibility

Certain random behaviours exhibited by actors can elicit unique responses from the ego vehicle. The simulator should enable users to reproduce the same set of random actions performed by actors, known as seeding. This parameter evaluates the simulator's capability to reproduce a specific set of actions performed by actors for further analysis.

3.1.3 Environment

The simulator's environment includes the weather, terrains, maps, and conditions where the vehicle operates. This section explores various the criteria for various comparative parameters related to environment.

3.1.3.1 Topography

a) Pre-defined Libraries

This parameter assesses the simulator's capability to feature pre-designed environments consisting of diverse landscape and weather conditions. The simulator earns a score of 0.5 if it satisfies one condition and a score of 1 if it fulfils both conditions.

b) User-Defined Environment

Simulator tools enabling users to define their own maps and various weather conditions enhance customization and adaptability, allowing for efficient simulation tailored to specific needs.

c) Real World Maps

The capability to import or utilize real-world maps within the simulator holds significant importance, particularly for developing commercial autonomous vehicles that directly navigate on road. A simulator scores 1 if it facilitates user access to import or integrate real-world maps and associated features.

d) Photorealism

Photorealism is the concept of how closely a simulator's environment resembles the real world in appearance. Assessing photorealism is subjective, thus scored by users through visual inspection on a scale of 0 to 1. A simulator that closely replicates reality receives a higher score, while deviations affect the score inversely.

e) Variable Resolution

The ability to vary the level of detail in the rendered environment by adjusting resolution. Variable resolution provides flexibility, allowing users to balance computational resources against information accuracy, with lower resolution saving resources but potentially losing detail and vice versa. This parameter checks the ability of a simulator which enables the user to vary the resolution.

f) Context-Aware Map Planning

The simulator-generated map should possess contextual coherence, ensuring that elements such as buildings, roads, and traffic lights correlate logically within the environment. This feature scores 1 when a simulator has the capability of generating contextually plausible maps.

g) User-Defined Scenarios

Allowing users to generate specific scenarios, such as road constructions or obstacles like fallen trees on roads, contributes to the simulator's versatility. This parameter is scored by its ability to facilitate user scenario generation.

h) Material Definition

The optical properties of environment materials can influence sensor data, impacting sensors like Lidar and Cameras. For instance, Lidar may fail to detect glass walls due to the laser beam passing through it rather than reflecting. This parameter evaluates the simulator's capability to enable users to define materials and texture of objects in the environment.

i) Variable Level of Sun Shades

Sun shades influences significantly in replicating the time of day and weather and impacting sensor readings. This parameter refers to the simulator's ability to allow users to manipulate sun shades, considering factors like time of day and weather

j) Special Optical Features

Real-world optical phenomena like sun glare, mirage, dust, and fog can impact driver visibility and sensor data for driverless vehicles. Therefore, it's crucial for simulators to model these features which impacts sensor data and subsequent vehicle actions.

k) Data Visualization and Analytical Tools

Simulator tools facilitating the display of various data and analyses during training and testing offer valuable insights to users. This parameter assesses whether the simulator allows users to define data visualization and monitoring tools, which could include sensory data from the vehicle, algorithm-related data, or annotations like bounding boxes and semantic truth.

l) Rendering

During testing, the quality of scene rendering aids in analysing algorithm performance. Users should have the ability to modify rendering parameters such as view angle and frames per second, ensuring adaptable display quality. This parameter evaluates the simulator's capability to offer configurable rendering options, earning a score of 1 if available.

m) Recording

Recording the training process allows users to analyse performance and playback sessions for future evaluation. This parameter accounts for the simulator's ability to record the training process alongside associated data, aiding users in comprehensive analysis.

3.1.4 Ego Vehicle

The agent vehicle, or ego vehicle, is the vehicle which is subjected to training or testing for autonomous driving within the simulator. This section explores various parameters evaluating the simulator based on the ego vehicle.

a) Vehicle Geometry

Vehicle Geometry refers to the dimensions and bounding box encapsulating the vehicle in the 3D environment. This parameter evaluates the simulator's capability to allow users to configure the geometry of the ego vehicle.

b) Wheel Geometry

This parameter involves the ability to modify suspension and steering system parameters of the ego vehicle within the simulator. Ensuring adherence to physics related to wheel geometry is important as it impacts driving performance significantly.

c) Real-World Models

This parameter assesses the simulator's feature of enabling the user to import and use of 3D models of real-world vehicles developed by vehicle manufacturers. This feature is important and timesaving for researchers in vehicle manufacturing companies.

d) Real-World Physics

This parameter is a check for the simulator's capacity to simulate various physical parameters associated with driving, including wheel slip, crash damage, and vehicle inertial behaviour. Accurate replication of these features is vital for developing algorithms for real-world on-road autonomous vehicles.

e) Input Commands

The driving algorithm generates commands for the vehicle (e.g., acceleration, steering angle) based on observations. Configurability of input commands for every component within the ego vehicle is crucial to accommodate various algorithms. This parameter ensures the simulator's ability to configure input commands for the vehicle.

f) Other Components

Minor components like headlights and horns can impact driving. The simulator should allow users to configure the intensity or impact of these components. This parameter evaluates the simulator's feature for adjusting the effects of such components.

3.1.5 Framework

The framework within which the simulator operates plays a crucial role in its versatility and usability. This section evaluates various comparative parameter related to simulator’s framework.

3.1.5.1 Cross-Platform Compatibility

This parameter assesses the compatibility of the simulator software across diverse operating systems, determining support for specific OS environments such as Linux, ROS, Windows, MacOS, and any other relevant OS. The simulator earns a score of 1 for each supported operating system from the listed ones.

3.1.5.2 Open Source

Evaluating whether the simulator or parts of it are open source, enabling users to tailor the simulator to their specific needs by importing components as plugins and developing customized software. A score of 1 is given if the simulator allows for such user-driven customizations.

3.1.5.3 Scripting Language Support

This parameter evaluates the breadth of scripting language support within the simulator, crucial for facilitating communication between users and the simulator. The assessment includes popular languages like Python, C++, C, R, Java, and other relevant scripting languages. The simulator earns a score of 1 for each supported scripting language from the listed options.

3.1.5.4 Scalability

This parameter evaluates the simulator's ability to provide tailored versions to suit diverse user needs. Offering various individual light versions with unique features, alongside a comprehensive pro version encompassing all features, the simulator ensures adaptability to different user requirements. This feature contributes to making the software more streamlined and efficient for users by allowing them to opt for lighter versions catering to their specific needs.

3.1.5.5 Computational Resource Evaluation

In this study, the computational resource consumed by a software is calculated based on electrical energy consumption by a baseline system under maximum operation. The components of the baseline system and its power consumption is tabulated in Table 2

|  |  |  |
| --- | --- | --- |
| Component | Model | Power Consumed (Appox.) |
| Processor | Intel i9 processor | 200 – 250 Watts |
| GPU | Nvidia RTX 3090 – 24 Gb | 400 Watts |
| RAM and ROM | 16 Gb and 500 Gb SSD | 15 Watts |
| Monitor | 27” 4K resolution Monitor | 50 Watts |
| Fan and other components |  | 25 Watts |

The score is calculated by comparing the simulator's power consumption during operation against the maximum power consumption of the baseline system (approximately 750 watts).

The score for the computation resource consumed by a simulator A is given by

Score = 1 – (power consumed by simulator/ maximum power consumption by baseline system)

3.1.5.6 Community Support

Community support is crucial for public software, evaluating three streams: structured documentation, tutorials, and discussion forums. Each stream receives a score of 1 if available.

3.1.5.7 Distributed Computing

This parameter assesses the simulator's ability to execute multiple simulation instances on different machines, converging into a coordinated and coherent simulation system. This parallelization capability enhances efficiency by reducing runtime and increasing the number of simulation episodes within a given time.

3.1.6 Driving Algorithm

This section delves into the comparative assessment of the driving algorithm used within the simulator, analysing various parameters critical to algorithm and associated data.

a) Data Exchange

The interaction between the algorithm and the simulation engine often occurs through communication channels like APIs or dedicated communication bridges. This parameter evaluates the presence of pre-defined communication plugins, scoring 1 if available.

b) Multithreading

Multithreading is pivotal for parallelizing computations between the simulation engine and algorithm training, enhancing speed and enabling more episodes for a given time. This parameter evaluates the availability of computation and algorithm parallelization within the simulator.

c) Data Logging

Data logging involves collecting and storing simulation-generated data for future analysis. This parameter assesses the simulator's support for user-assisted data logging, scoring 1 if it aids users in logging specific data of interest.

d) Hardware in Loop (HIL)

The incorporation of real-time hardware, such as sensors and vehicle controls, within the simulator aids in testing hardware performance. This parameter evaluates simulator support for any hardware involvement during operation.

e) ML Libraries

Support for popular machine learning libraries like TensorFlow, PyTorch, and Scikit simplifies machine learning model training. This parameter assesses the simulator's compatibility with integrating such ML libraries, enabling users to utilize these tools within the simulator environment.

3.2 Methodology

This section outlines the process of deriving a single score for a simulator using the above-mentioned parameters. The scoring system comprises two main components which constitute to form the final scores:

3.2.1 Base Score

Each parameter in the evaluation criteria is assigned a score of 1 if the simulator considered for evaluation meets the specified criteria and 0 if it does not. This results in a base score vector unique to each simulator, typically assigned by the simulation software developers.

3.2.2 User Weight

Users assign values ranging from 0 to 1 to parameters based on the individual user importance. These user-assigned scores create a user weight vector that reflects the significance of each comparative parameter to the user.

3.2.3 Final Score

The final score, denoting the suitability of a set of n simulators S = (S1, S2, S3… Sn) for a specific user (U), is calculated using the dot product of the base score vectors of simulators S and the user weight vector of user U. This computation generates a usability score for each simulator, allowing users to assess and compare simulators based on their preferences.