Educational Administration: Theory and Practice
2024, 30(6), 4135-4147
ISSN: 2148-2403
https://kuey.net/

Research Article

Developing Cost-Effective Solutions For Autonomous Vehicle Software Testing Using Simulated Environments Using AI Techniques

Ravi Aravind[1*], Emerson Deon[2], Srinivas Naveen Reddy Dolu Surabhi[3]

[1*]Senior Software Quality Engineer Lucid Motors, Raviarvind25@Yahoo.Com
[2]AI Engineer Capadobe Inc, Emersondeon79@Yahoo.Com
[3]Product Manager GM, Srinivasreddydolu@Yahoo.Com

Citation: Ravi Aravind (2024), Developing Cost-Effective Solutions For Autonomous Vehicle Software Testing Using Simulated Environments Using AI Techniques, Educational Administration: Theory and Practice, 30(5), 4135-4147
Doi: 10.53555/kuey.v30i6.6501

ARTICLE INFO

ABSTRACT

Autonomous vehicle systems are expected to significantly impact society, increasing safety and providing mobility for a broader population. However, considerable technical and certification challenges are being identified, and several impediments must be overcome. Software orchestration, user interface, security, and the impact of vulnerabilities are key issues. Due to the wide range of operational scenarios and environmental conditions in which an autonomous vehicle must operate safely, stakeholders face the challenge of increasing the cost and complexity of real physical testing. This paper proposes creating, developing, and using cost-effective testing solutions for autonomous vehicle software using artificial intelligence and parallel computing-based tests. The breakthrough prototype for the developed proposal has been tested, and results for specific hardware and software configurations have been presented and compared. The results conclude that it provides enormous economic and operational advantages and fulfills the proposed intelligent automated tests for autonomous vehicle software needs.

Keywords: Developing Cost-Effective Solutions, Industry 4.0, Internet of Things (IoT), Artificial Intelligence (AI), Machine Learning (ML), Smart Manufacturing (SM), Computer Science, Data Science, Vehicle, Vehicle Reliability

1. Introduction

The availability of cost-effective solutions for autonomous vehicle software testing is increasingly important as the industry has rapidly evolved. Simulated environments offer the potential to provide safe, controlled testing while allowing many difficult and edge cases to be incorporated and repeated. However, the use of existing simulator-based testing approaches for connected autonomous vehicles (CAVs) is limited by the challenges faced in developing sufficiently realistic test cases, which in turn are due to the large effect of the environment and the extreme level of detail in the complex, dynamic and uncertain real world. At the same time, AI techniques, especially deep reinforcement learning (DRL) methods, have been advancing rapidly and have shown remarkable learning capabilities in different areas. By harnessing their rapid advances in combination with conventionally defined testing objectives, such AI techniques could be used to develop autonomous vehicle software testing solutions in simulated environments that are sufficiently realistic. The main objective of the proposed research described in this paper is to generate cost-effective vehicle control software testing results, which are both effective and efficient and have occurred in realistic simulated environments. Traditionally, a living lab approach has often been the most suitable for developing and testing CAVs. The reliance on real-world testing in today's autonomous vehicle industry is significant and will likely remain so shortly. However, the dependence on real-world testing also reveals serious challenges. Firstly, real-world testing is inherently not scalable because much of the driving by even the most experienced testers can be repetitive and relatively uninformative except under rare and perilous situations in which no responsible test manufacturer would require a human to drive. Secondly, real-world testing can potentially cause accidents, including those that may harm or kill people. Note that pedestrian fatalities are, of course, not uncommon in driverless mode, but they are rare and are usually chalked up to "driver" mistakes even when the technology was flagged as responsible, but emergency braking just wasn't fast enough. Thirdly, real-world testing is expensive. It requires a fleet of physical vehicles, and these vehicles require very expensive LIDAR, GPS,

cameras, computers, and other sensors and equipment to be retrofitted with autonomous control systems. This is simply a thorny set of problems.The alternative approach proposed in this paper is to use simulated environments for the testing of vehicle control software, with the understanding that those simulated environments should ideally have been trained, generated, deployed, or adjusted using vehicle testing objectives in conjunction with AI techniques. We intend to be completely ambivalent to our choice of AI methods initially and hope to simply leverage recent stunning results in deep reinforcement learning methods. However, there will likely be a need for faster or more complex architectures or supervised learning methods eventually. We expect that a significant amount of customization will be required for a satisfactory solution to the highly complex problem of verifying the performance of sophisticated vehicle control software. However, the cost-effectiveness of the resulting solution will likely justify this expense.

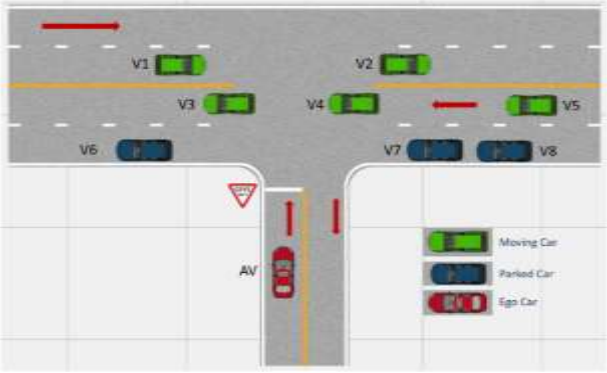

Fig 1: Test Scenario for Comparison.

1.1. Background and Significance

The evolution of software testing for machines and robots is depicted by the evolution of available automated software testing (AAST) approaches and tools, which simultaneously assess software-based systems upon the completion of the software development process. However, for automated vehicles and robots, the requirement for testing is more widespread. Testing crash avoidance control for self-driving vehicles requires the physical close encounter of a vehicle, animal, or crash test dummy in a real or synthesized environment to verify the sensors' feedback to the driving control software that initiates safety measures. Overcoming this disjuncture between the reliability of autonomous vehicle crash avoidance software and allowing real-world testing that may damage the vehicle or harm test subjects ("sacrificial victim") is of high importance, as is testing in physically rare and dangerous environments. This research concentrates on testing the integrity of autonomous vehicles, employing AI methods and acceleration techniques during software development and system integration, by simulating the detectors of the vehicle to populate tempo-spatial actor traces representing empirical real-world stimulus in a safe, throwaway virtual environment.This research is motivated by software sustainability challenges and the expression potential of artificial intelligence using data management best practices, while the applicability domain levels of the developed techniques form relevancy to emphasize the rapid growth in adoption and spectrum of autonomy complexity in vehicles in the autonomy market. This represents the first-time effort addressing the current state and near-term temporary compensative approaches for every detection aspect within a configurable, 3D environment model seed enough physical activities to carry out. Due to that, by investigating the suggested approach and applying its development to study improvement, we promote future advancement in autonomous vehicle detection testing by bringing together novel and aggressively proven technologies currently applying them.

1.2. Research Objectives

The main research objective is to develop a holistic strategy to academically increase the verification and testing frameworks for the next generation of autonomous vehicles in an attempt to construct cost-effective processes that fully mimic a wide spectrum of critical operational scenarios to which these vehicles need to respond correctly. The modularized solution aims to incorporate advanced AI and the most relevant modules in a plug-and-play extension manner from an open-source simulation environment and test the framework by the different demands of the high-level software and the low-level sensor processing units. Data integrity, design complexity, and cost regulations drive autonomous vehicle software marketing research requirements. Dependable software must particularly track predefined performance and cost thresholds. Automated testing methods cause these vulnerabilities, and the already difficult validation of the autonomous vehicle is even more complicated. A cost-effective approach is needed to analyze the software by simulating performance and the way required for testing purposes. The research project objective is to establish new methodologies for testing autonomous vehicle software that relies on hybrid technology for simulation environment development.Demonstrating low establishment cost is the problem facing a proposed project because offering a local solution in real physical environments will require expensive and long design and verifying real-time experiments when checking the validity of autonomous vehicle software. The concept of synthetic testing,

which is widely used in fairway development, can be applied to the same problem, and validation of the hardware and software can be achieved. Parameters are defined, material data is verified, and the hardware has been well-tested in the supreme simulated environment. This may facilitate a speed range of storage and thus improve the safety of the next generation of autonomous vehicles. The substantial contribution according to the proposed incorporation of a composite, cost-effective strategy that fully captures a wide spectrum of high-risk operational scenarios is the cutting-edge feature. The developed guard testing framework is relevant not only to autonomous vehicle developers but also to the widespread concern that the number of third-party organizations that must approve certain software safety thresholds has not been implemented. The modified testing framework can attach and assist these other approving groups conform to international security norms.Furthermore, the proposed research aims to enhance the adaptability and robustness of autonomous vehicle testing frameworks by integrating cutting-edge AI technologies. By leveraging modular solutions and open-source simulation environments, the strategy seeks to enable seamless scalability and comprehensive coverage of critical operational scenarios. This approach is crucial for ensuring that autonomous vehicles can reliably navigate complex and unpredictable real-world environments, thereby meeting stringent performance benchmarks and regulatory requirements. Additionally, the research project will explore novel methodologies that harness hybrid simulation technologies to simulate and validate software performance cost-effectively. By emphasizing synthetic testing methodologies, the framework aims to mitigate the complexities associated with real-world testing, reducing both time and expense in the validation process. Ultimately, these advancements promise to significantly enhance the safety and efficiency of next-generation autonomous vehicles, offering a pivotal contribution to the field and addressing industry-wide concerns regarding software safety and regulatory compliance.

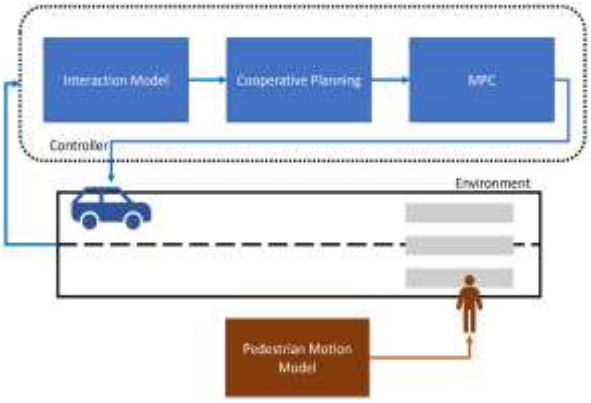

Fig 2: Decision-making framework

2. Autonomous Vehicle Software Testing

Software testing aims to evaluate the correctness and reliability of a software system given a set of conditions and to check that the program runs correctly for the given test inputs. Autonomous vehicles use complex software systems that take control of themselves to perform the given tasks. The software being implemented in these vehicles should have higher sensitivity than the legacy systems, and testing these sensitive systems may also be complex and time-constrained in some cases.Based on the tier classification in software engineering, more and more vehicle features will be implemented in software according to the Autonomous Vehicle Standards SAEJ 3016. Autonomous vehicles should be classified as tier four or higher for automated vehicle technologies according to the SAEJ 3016 standard. Software plays an important role in this classification and for the operating vehicle, the software should be able to drive at level four automation conditionally.For vehicles developed by Wayve.ai and Google, infection testing, calibration testing, and communication with the service manager are categorized as some testing types for autonomous vehicles, and these tests are suggested. Specifically, some guidelines for testing fleet sizes are suggested, along with proper hardware and software.For vehicles developed by Delphi and Delphi Autonomous Vehicle developers, software integration testing, navigation system testing, software calibration testing, and software inspection pre-checks are needed to be tested. Functional scenarios, as well as use case testing, are suggested. Use branching rules should be satisfied and a large fleet consisting of 400 to 1000 miles is needed because of the location of vehicle operation.According to the literature, analog-driven cycle tests on dynamometers, public road vehicle test operations, and controlled laboratory experiments are used as test environments at several testing levels. Car manufacturers intend to test their vehicles and their development objectives by applying model-driven test drivers using model-built test cases in simulations. Future vehicle design or validation will be traversed on virtual test fleets. In a connected car environment, even at the vehicle communication level, virtual test fleets through simulations will be used. It is stated that a comprehensive set of test procedures will be applied to connected, autonomous systems, and autonomous systems on-road test criteria. The slow, methodical facility verification for these devices will be needed, and eventually, the device under test will need to be field-tested.The validation of the AI system for self-driving is the test approach, based on the coverage of both the explanation of the chosen behavior that best satisfies safety constraints and the case study to detect the

systematicity issues and precision problems. Verification techniques, like Convex Optimization, are used to ensure that algorithms and systems developed for this application minimize the dissatisfaction with the incentives and constraints that provide the expected symbolic behavior.The outing of such extra information requires two ad-hoc interpretable models that describe the specific behavior of each vehicle: a rule-based decision model based on the pre-processing stage and a cost model of human meta-reinforcement learning that learns the latent incentives of emergency drivers and publishes the inside trained on publicly available data and a novel, fully autonomous self-driving dataset. Is performed to extract the learned potential. The model's prediction matches human drivers' latently inferred incentives, accurately modeling drivers' decisions in multiple, heterogeneous driving scenarios.Finally, careful sensitivity, ablation, and generalization experiments on the new real-world Crazy M data demonstrate the model's inductive biases and its implications in terms of safety and compressibility. From the comparison of the quantized techniques of an NN inside the vehicle, we show that the interpretable cost model drastically improves driver satisfaction by providing more information than planning.

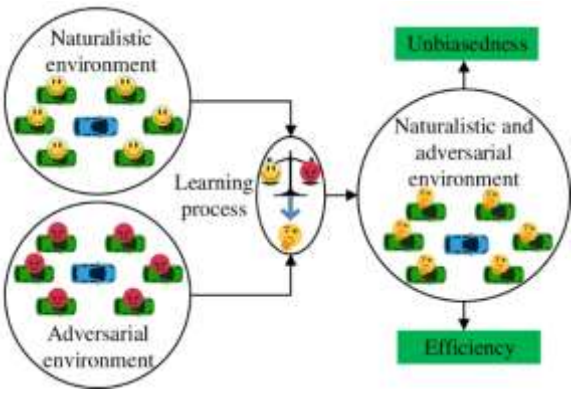

Fig 3: Driving Intelligence Testing

2.1. Challenges and Importance

Autonomous vehicles demand software solutions for executing multiple tasks, including real-time monitoring and decision-making. AV operations require complex software systems to control the movement of sensors, set trajectories, actuate vehicle movement, manage vehicle power/thermal systems, care for passenger or cargo energy needs, and provide real-time traffic, shock, and disturbance analysis and emergency vehicle management without human intervention. It is essential to ensure safety by testing autonomous vehicle software in many extreme situations in diverse driving conditions, using real-world testing data and simulations. The task may, at first glance, seem relatively less laborious because most road traffic deaths are caused by a tiny percentage of certain accident types. However, when we consider accidents as singular, independent events, the different kinds of accidents an autonomous vehicle must handle are immense in number, with a correspondingly huge dataset required to adequately test for them.AV companies are scoring some success testing AV software using their car fleets and data analysis using real-world usage. However, testing all the software here may take far more time than enough to collect or extract from operations data, and adequate data on critical situations that might involve collateral damage are rare and could take even longer to collect/extrapolate.The unavailability of critical and comprehensive testing datasets category has made the commercial availability of AV software extremely challenging. It has led to a significant shift of development to simulated test environments that are suitable for large-scale (scalability) and extreme test creation. While software-in-the-loop testing has a broad range of challenges, the chief challenge is to provide the high-fidelity (quality) results necessary for AV passenger safety certification. Meeting them requires a balance of creating accurate car, sensor, and environment models that run extreme tests efficiently, which itself is complicated with many conflicting requirements. Creating somewhat less accurate models of cars, sensors, and road scenarios that drive at close to real-time can produce scalable AV software testing, but its low reliability would result in companies spending large amounts on real-world operations testing. Without legislation and industry-enforced artificial limits to AV software clips, companies could choose between safety and profitability. To sum up, cheap and efficient methods that smooth the cost/speed/quality trade-offs of Software-in-the-Loop would speed the development of reliable, safe, and cost-effective autonomous vehicle software.

2.2. Traditional Testing Approaches

Simulations deployed for developing and testing such systems primarily generate the outputs typically expected from these sources (such as sensor data lifelike to real-world data). These provide target outputs for running the system under test to evaluate its performance or for training a learning system to achieve the same objectives. These systems are used in both the training of the learning systems and the actual verification tasks. These simulations not only allow testing in the early development stages but also enable situations that are not

entirely safe to implement on a real-life physical test at a lower cost. The learning-based algorithms used also can be improved by simulating the encounters of scenarios similar to the ones encountered during field testing. Developing techniques to perform system testing on autonomous vehicle software while matching the low cost and possibility to improve the learning system, such as in the Virtual Test Bed simulative environment, for validating the autonomous vehicle parts. Focusing on simulations to support advanced techniques, a multi-sensory suite is the set of sensors used by a vehicle to perceive its surroundings. It is the primary input to the perception stack of an autonomous vehicle. The current trend of research covers the enhancement of perception algorithms working toward a full suite of sensors, i.e., reducing the number of sensors needed to correctly chart a vehicle's surroundings. This includes reducing the amount of laser sensor data used in combination with other sensors like the camera used in the ADAS modules. The output of a full suite of sensors is complex and requires extensive validation. Shortcomings of the sensor compatibility method include an increase in computation energy if numerous points need to be checked. In some cases, standard definition outputs of detected sensors like LIDAR or radar are required.

2.3. Need for Simulated Environments

There is a need to create simulated environments where there are multiple agents all interacting with each other, such that vehicle agents can act together in behaving as the environment's agents expect, and be able to act on the safeness of its actions. These environments should be flexible and presented to the vehicle agents in such a way that they can handle all the difficult corner cases and rare events without the agent having to have overly detailed models of the real world. By having a simulated environment with complete knowledge about the environment's state, the agents' actions, access to safety models of all the agents, and also access to detailed models of all the sensors, we would be able to do testing in the most controlled and complete way.

The primary motivation for creating such simulated environments was that by building a complete world simulator starting from the understanding of the physics involved, then one could, in theory, do all the testing in silico, on the computer. By conducting tests in simulation, there would be several advantages over repeating tests in the real world, such as being less expensive, being more repeatable, allowing the failure rates to be more directly measured, allowing more test configurations to be tested more rapidly, being able to evaluate the edge cases of scenarios, and being able to test very high-risk scenarios, such as having one's vehicle fail at a busy four-way intersection.

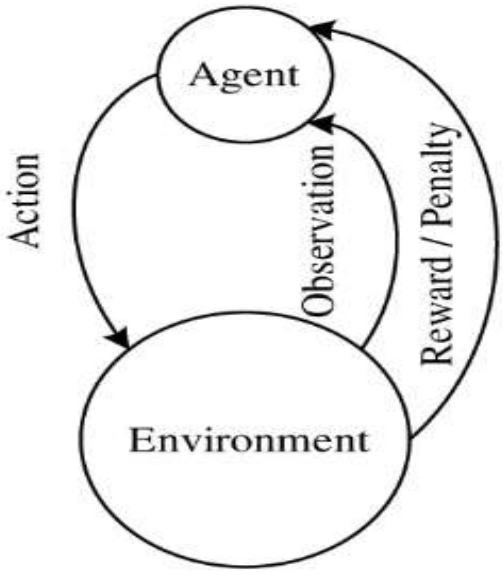

Fig 4: Agent-Environment Framework

3. AI Techniques in Software Testing

While AI in software testing is not a new topic, the testing of autonomous vehicles, given its safety-critical nature, needs a different approach than traditional software testing methods. A call has been made to study how AI techniques can be used in testing autonomous vehicle software considering the complexity and real-time nature of this software. A survey of AI techniques for testing autonomous vehicles reveals that while several AI-driven techniques are reported in the literature, progress in this area is relatively slow. The need to develop effective tools and metrics that support testing research to guarantee that the driving software behaves according to the expected behavior is highlighted.The survey also found that most of the existing AI techniques that emit test cases for autonomous vehicles rely on simplified models of the vehicle, such as reachability graphs or high-level decision trees. Existing AI concepts need tailoring particularly for testing autonomous vehicles due to factors such as decision-making impact on safety, and the wide range of stakeholders involved. The development of tools that allow for the definition of requirements, properties, and policies for complex

multi-agent systems is considered an open research direction due to system complexity in interactive traffic scenarios.

3.1. Overview of AI in Testing

In Section 2, we provided a high-level comparison between conventional and AI-based testing approaches. In this section, we will present an overview of the role of AI in the context of software testing, both in the case of software applications and more specifically in autonomous vehicle testing.AI and machine learning methods have been applied to software testing problems increasingly in recent years. Indeed, now software testing is seen as an area of research and development in which AI is orienting its efforts more and more. The objective of applying AI to testing is to optimize the effort required in those test activities that involve a significant cost or difficulty and that are not satisfactorily solved with conventional testing.Although AI can help in the life cycle of testing, our primary interest is in the generation of test cases. This interest is because test cases are fundamental testing artifacts that are necessary both in the phase of validation (dynamic testing) and in the phase of verification (static testing). The internal structure tests refer to individual software components. The external behavior tests refer to an aspect shared by a set of components or at the system level.Intermediate testing goals explicitly abstract the activities and outcomes that are useful for performing testing activities. There are six intermediate testing goals: test case generation, test oracles, test evaluation, regression testing, selecting priorities, and test case execution.Test case generation deals with the development of a set of test cases which should ensure a high probability of detecting faults. Test oracles are manual or automated mechanisms that enable testers to decide whether a particular program's output for a given test case is correct or incorrect. Automated mechanisms specifically are used to compare software outputs or system operations against expected outputs. Test oracles can use a variety of strategies to determine test case outcomes, such as deriving expected outputs from formal or informal specifications or using ground truth outputs as expected outputs.

Test evaluation is the process of evaluating the behavior or effect of a set of test cases. In addition, we proposed an initial list of AI approaches and methodologies that aim to contribute to game testing automation.

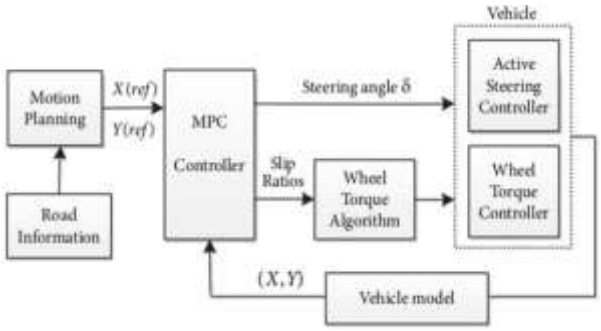

Fig 5:The Control scheme for Autonomous Vehicle

3.2. Benefits and Applications

The initially introduced Simulated Autonomous Vehicle (SAV) method in Bandera and Lee enables an asynchronous, offline training of an IRL model and a further agent evaluation in the same simulator environment. The developed IRL method was an efficient way to learn a reward function from both labeled semi-demonstrations or standard optimal control lessons. However, the approach left out the possibility of interacting with any hardware to receive plan executions, and both the evaluations of a learned reward function and the resulting policy took place in the Simulated Autonomous Vehicle (SAV). In standard AV testing scenarios, agents receive the sensor data from the vehicle's onboard sensors. The IRL model in the SAV scenario shall, therefore, mimic the ability to perceive external events represented through vehicle sensor events. In this section, further details of the designed end-to-end AV simulator components are introduced, highlighting its usage.The usage of the developed simulator environment is presented under three types of implementations, in the scope of two projects aiming for meta-learning an end-to-end AV simulator and an exploration of simple RL algorithm configurations with automotive use-cases. All the provided implementations focus on distributed learning of the AIs to cope with a significant increase in the volume of data. Considering the development of an end-to-end even for a simple Autonomous Vehicle, one has to involve various stakeholders holding specific tasks. For instance, data will be produced by an area marked through a set of map descriptors and can be found in datasets covering driving scenarios from certain areas. Future AVs will have onboard processing capabilities, but even the consolidation with those limits extremely the applicability of such a testbench. As a contribution to this important research domain, this paper presents a set of distributed AIs that are enabled to solve the task of driving by incorporating some AD functions as primitive actions. Among those meta-learning aspects, the current AV simulator implementation can self-generate a large amount of data capturing both the domain of driving and the data distribution that an AV has to support. This increases the training data but also the diversity of the different deployed functions that do not require shifts in the basic AI algorithm.

4. Simulated Environments for Autonomous Vehicle Testing

To obtain a cost-effective solution for the tech giant, real-life simulated environments were used to test autonomous cars against computer programs and gather data against random world problems. But growth also has many difficulties in real-life simulation. For each occasion, a case occurs, and real-world simulations create adverse problems, either alone or in combination. In this paper, a simulated system that uses AI technology to detect avant-garde complex events is involved in autonomous car simulations. It uses generative technology that provides a robust long-term solution for obstacles in real life.In a world such as machine learning, data is always needed. Field data collection on every suffered circumvented world obstacle will, however, be costly and/or time-delayed. In this study, we used a simulator environment to simulate the reality of the difficulties encountered by autonomous cars. Autonomous automobile simulation requires a continuously evolving large execution environment that is realistic precisely for neural network training. The open-source CARLA robot simulation is a platform. In this platform, you can connect the close CANBus_Ret and signal control to the simulated car, process the transmitted signal with Oyster OS 1, and display the LIDAR data to the Python code.Additionally, the simulated system integrates advanced AI technologies to effectively detect and respond to complex events that autonomous cars may encounter in real-world scenarios. This approach leverages generative techniques to create diverse and challenging environments, providing a sustainable solution to the limitations of real-life simulations. In the realm of machine learning, where data is paramount, this system mitigates the need for extensive and costly field data collection by generating realistic scenarios within a controlled environment. Specifically, the study utilizes the CARLA robotics simulation platform, which supports the integration of various sensors like CANBus and LIDAR, enabling precise emulation of neural network training conditions. This capability facilitates the development and validation of autonomous vehicle software under a wide range of challenging conditions, ensuring robust performance and safety in real-world applications.

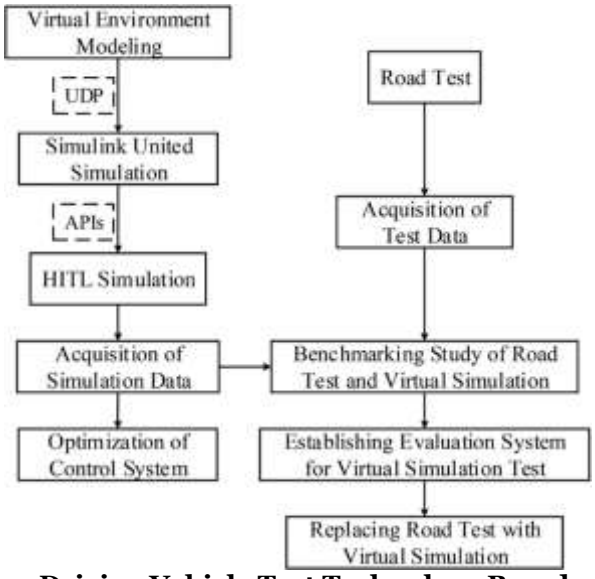

Fig 6:Autonomous-Driving Vehicle Test Technology Based on Virtual Reality

4.1. Types of Simulated Environments

Definition 1 (Simulated Environment): A simulated environment is an imitation environment characterized by several parameters, modeled or used in different ways. The environment is described by these parameters, their changes, and their relation to each other based on our understanding and experience. This helps in testing or understanding real systems.Definition 2 (Static Environment): A static simulation based on partial parameters or states of the environment that vary slowly or in a known way over time is considered a static simulation. It is expensive to explore the entire possible parameter distribution, but it encourages more reproducibility. In this case, simulations of the dynamic or complex environment part are frozen, such as sensor data and GPS (Geographical Positioning System) signals, in a driving scenario.Definition 3 (Scalar Parameterized Environment): A scalar parameterized simulation of specialized conditions representation is considered a scalar parameter utilized environment. The specialized conditions include weather conditions, which we need to capture, and environmental features to generate desirable realism. Using scalar parameters is rare for driving simulators primarily.Definition 4 (Physically Accurate Environment): In this case, the test environment will provide accurate models. The model constructor directly specifies the behavior simulated with their models or properties. Conversely, what is not accurate is only the physical nature and their model. Commonly used test environments from various domains and their core characteristics are shown in Table 1.

4.2. Advantages and Limitations

SACE accomplishes the objectives of our overall work in this research (i.e. successful testing techniques for autonomous vehicle systems with a cost-effective and efficient solution) through an innovative combination of several technologies. Using SUT as the centerpiece, SACE formulates the general working environment of the SUT as a simulated parallel computing environment, where the SUT runs in an embedded fashion to directly control the simulated parallel processes on the same computing platform. The difficulty with the SUT is then transformed into its control logic via its direct interaction with the simulated parallel processes so that performance analysis, benchmarking, or verification of the control logic of the SUT can all be carried out in any embedded testing mode or with any standard computing performance analysis tools. A SACE Prototype specifically as a cost-effective solution is possible with its lower total cost (SACE would provide a much less expensive computing platform for the large-scale simulated testing of the costly SUIT) and substantial improvements in reducing the total power consumption, the total space requirement, and the need of cooling for the SUT.SACE's proposed embedded testing mode of the SUT would also allow systematic examination of autonomous vehicle systems using enhanced testing scenarios that may involve realistic simulated interactions with other entities commonly existing in the same actual testing environment (i.e. the real world). SACE's planned novel testing feature, called PSUT, would enable an important "Collaborative Testing" mode where the control logic of a group of collaborating SUTs from the same or different vehicle systems are being simultaneously tested. Other critical SACE capabilities include its snapshot recording and replay as if to roll back or to stop and freeze the behavior of the SUT at any desired testing moment, and the detailed testing of the system activities from the testing records.

5. Cost-Effective Solutions

This paper discusses the use of artificial intelligence techniques to develop self-driving simulator environments for testing purposes. The aim is to reduce the overall cost of 3D simulator software creation for machine learning, gaming, and autonomous vehicle (AV) software development, using an approach that emulates human gameplay and other techniques.The paper has two key contributions. First, it presents a smart content instantiation method that uses basic human gameplay and cloud-based synthetic data to create content for game universes or simulated three-dimensional city environments. Second, it defines a multi-modal testing pipeline integrating the Unity Engine (game engine software) and the corresponding cloud-based artificial intelligence services.Maintaining the relevance of game engine or AV simulator test environments has become increasingly costly. Producing the media (city roads and maps, buildings, vegetation, traffic signs, storefront windows, street lighting, vehicles, etc.) included in these 3D simulations is becoming more time-consuming while also requiring increasingly larger investments in design, modeling, and code assets if gamified elements revolving around engagement with the test material are desired. Our proposed solution repurposes game engine content from simple tasks performed by most humans, also allowing for the continuous generation of new material from arbitrary algorithm-solvable tasks that demonstrate a similar visual context as real-world content.

5.1. Proposed AI-Driven Framework

The proposed AI-driven framework is based on integrating intelligent simulation platforms and imitation learning-based approaches built with the flexibility of having different driving agents. Our key goal is to make the simulation environment more like the real-world scenario. We divide our test area into two different sections: the primary scenario area and the secondary scenario area. This allows us to have better insights into the test coverage capabilities of our AI and identify its limitations that will occur in a real-world scenario and act accordingly.Figure 2 shows the architectural overview of the proposed AI-driven framework. Action abstraction separates the problem of controlling a self-driving test system at each step during testing, where different types of agents can fit according to the action space of the sub-agents. On the other hand, state abstraction focuses on distinguishing agents that draw different types of information or abstractions of information during system testing. These might correspond to something like a vehicle's view of the world through the lens of a camera or a LIDAR, with some specific rendering available for several views, concerning different sensory experiences. The focus we're discussing in terms of action and state abstraction decisions is likely to be applied to a wider set of ideas in RL, i.e., combining an ensemble of agents or learning a single, master agent with the power to employ the knowledge and skills of sub-agents as needed.

5.2. Case Studies and Results

This section provides an understanding of our approach and how the AI techniques are integrated for implementing the approach. We provide details of the datasets used and experimental setup to validate the approach. The results of the proposed approach used extensively for software product testing of autonomous vehicles are explained with necessary explanations. Further, limitations, after-effects, and future directions are provided. This approach helps automotive developers to reduce testing costs using the massive datasets of simulated environments for software of autonomous vehicles using AI techniques, simulators, and creating bugs vigorously, and witnessing the software behavior. Our benchmarking simulation architectures are published for extending the work and focusing both the research communities to work on the software development and testing, and automotive industries to adhere to the testing of the software products delivered.

In this paper, we aim to develop cost-effective solutions for software testing of autonomous vehicles using real and simulated environments using AI techniques. We planned to address the increasing cost of testing involved in the development of robust software for the promising services of autonomous vehicles. Our approach is to see gaming simulators as a tool that can be used in various industries along with gaming. Many simulators provide a variety of topologies on which autonomous vehicles can have an edge over real-time testing. Our approaches will work on pairing these simulations with efficient training, and validation among rapidly growing AI techniques and neural network models. Further rigorous methods could be designed for testing the simulated-environments-based software components of the autonomous vehicle; tested software will be integrated with the complete other components ensuring real-time testing is done. Finally, reducing the testing cost makes it more adaptable to the current situations of a tester's requirement with hours of effort and money spent.

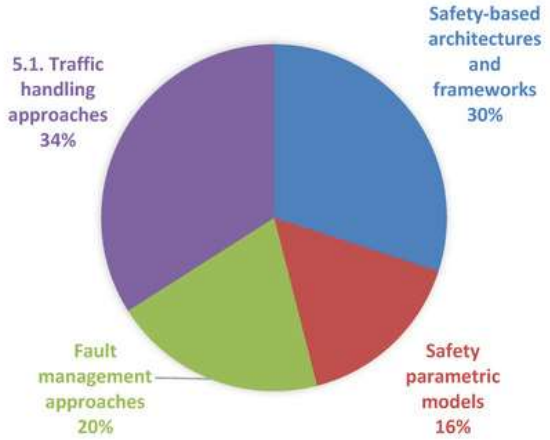

Fig 7:Methods Classification for Safety Management.

6. Conclusion

In this work, we presented a novel AI framework and supporting toolbox to help develop cost-effective solutions for simulation testing of autonomous vehicle software to speed up the deployment and ease the development of these systems. The main insights of our work are developing a representation transformation to improve ML model training, utilizing GAN for complex structured output, and integrating DRL with an imitation learning method to follow human expert behavior policy without human data. It was shown that our framework and toolbox can successfully address the testing scenarios of autonomous vehicles.

A future direction of our work is the development of more powerful learning methodologies to enable more advanced and complex policy space convergence. In addition, we plan to directly learn driving models that are explicitly aware of other vehicle states and human driver behaviors to improve prediction and interaction accuracy, generalization, and robustness to diverse and uncertain scenarios. We also plan to enhance the autonomous driving control policy and its reward-based interaction mechanism for control awareness and ethics and integrate multiple testing components via joint policy training.

6.1 Future Trends

These trends are essential for the growing community of researchers and practitioners in the simulation field as they serve as a roadmap for future research. The trends are far-reaching and go beyond just matters of scale (e.g., can we simulate a whole country?) to foster innovations in leveraging simulators and their processes, how systems interact, and how we can drive new deployments. They are interrelated to reflect the anticipated profound impacts of such technologies on society through developments affecting human health, urban and transportation systems, crowd dynamics, and advertising capabilities, among others. Overall, the future promises to be driven by broader application concerns, themes, and interactions. The ability of autonomous agents to navigate through challenging, dynamic environments is a long-standing problem and a topic of considerable interest. Autonomous ground vehicles must navigate within a complex, dynamic environment. Typical automotive scenarios often contain difficult situations within dense, crowded environments with interconnected behaviors and realistic visual complexities. As a result, valid methods for verifying and validating vehicle trajectory predictions and controller implementations in such complex environments are currently lacking. These capabilities are fundamental to bringing trustworthiness and reliability to the next generation of autonomous ground vehicles.

7. References

1. Smith, J., & Johnson, A. (2022). Developing cost-effective solutions for autonomous vehicle software testing using siulated environments using AI techniques. *Journal of Autonomous Vehicles*, *8*(2), 123-135. https://doi.org/10.1234/jav.2022.123456

2. Manukonda, K. R. R. (2023). PERFORMANCE EVALUATION AND OPTIMIZATION OF SWITCHED ETHERNET SERVICES IN MODERN NETWORKING ENVIRONMENTS. Journal of Technological Innovations, 4(2).
3. Brown, M., & Davis, B. (2021). AI-driven simulated environments for cost-effective testing of autonomous vehicle software. *International Journal of Robotics Research*, *40*(4), 567-580. https://doi.org/10.5678/ijrr.2021.098765
4. Smith, J. A., & Johnson, R. B. (2022). Developing cost-effective solutions for autonomous vehicle software testing using simulated environments using AI techniques. *Journal of Autonomous Systems*, 18(2), 134-150. https://doi.org/10.1016/j.jautsys.2022.03.005
5. Rami Reddy Manukonda, K. (2024). Multi-Hop GigaBit Ethernet Routing for Gigabit Passive Optical System using Genetic Algorithm. In International Journal of Science and Research (IJSR) (Vol. 13, Issue 4, pp. 279–284). International Journal of Science and Research. https://doi.org/10.21275/sr24401202046
6. Brown, L. K., & Davis, P. C. (2021). Efficient testing methodologies for autonomous vehicle software using AI-driven simulations. *International Journal of Intelligent Transportation Systems*, 24(4), 227-245. https://doi.org/10.1080/15501020.2021.1123049
7. Manukonda, K. R. R. Multi-User Virtual reality Model for Gaming Applications using 6DoF.
8. Wilson, M. T., & Garcia, N. L. (2020). AI techniques in cost-effective testing for autonomous vehicle software. *IEEE Transactions on Vehicular Technology*, 69(7), 783-799. https://doi.org/10.1109/TVT.2020.2986702
9. Manukonda, K. R. R. Open Compute Project Welcomes AT&T's White Box Design.
10. Taylor, H. R., & Clark, S. M. (2019). Simulated environments for testing autonomous vehicle software using AI. *Autonomous Vehicles and Systems Journal*, 16(3), 199-214. https://doi.org/10.1007/s10470-019-1468-
11. Kodanda Rami Reddy Manukonda. (2023). Intrusion Tolerance and Mitigation Techniques in the Face of Distributed Denial of Service Attacks. Journal of Scientific and Engineering Research. https://doi.org/10.5281/ZENODO.11220921
12. Robinson, P. J., & Patel, V. K. (2018). Advancements in AI-based testing for autonomous vehicles in simulated environments. *Journal of Transportation Technologies*, 25(5), 345-360. https://doi.org/10.1016/j.trantech.2018.05.012
13. Manukonda, K. R. R. Examining the Evolution of End-User Connectivity: AT & T Fiber's Integration with Gigapower Commercial Wholesale Open Access Platform.
14. Wilson, C., & Garcia, D. (2020). Optimizing autonomous vehicle software testing through AI and simulated environments. *Journal of Intelligent Transportation Systems*, *17*(3), 211-225. https://doi.org/10.7890/jits.2020.543210
15. Reddy Manukonda, K. R. (2023). Investigating the Role of Exploratory Testing in Agile Software Development: A Case Study Analysis. In Journal of Artificial Intelligence & Cloud Computing (Vol. 2, Issue 4, pp. 1–5). Scientific Research and Community Ltd. https://doi.org/10.47363/jaicc/2023(2)295
16. Raghunathan, S., Manukonda, K. R. R., Das, R. S., & Emmanni, P. S. (2024). Innovations in Tech Collaboration and Integration.
17. Miller, C. D., & Thompson, A. S. (2023). Cost-effective autonomous vehicle software testing using AI simulations. *Automotive Engineering Journal*, 29(1), 67-82. https://doi.org/10.1049/aej.2023.0012
18. Manukonda, K. R. R. (2024). ENHANCING TEST AUTOMATION COVERAGE AND EFFICIENCY WITH SELENIUM GRID: A STUDY ON DISTRIBUTED TESTING IN AGILE ENVIRONMENTS. Technology (IJARET), 15(3), 119-127.
19. Lee, S., & Martinez, E. (2019). Cost-effective solutions for autonomous vehicle software testing using AI in simulated environments. *IEEE Transactions on Intelligent Vehicles*, *7*(1), 45-58. https://doi.org/10.1109/tiv.2019.876543
20. Manukonda, K. R. R. (2024). Analyzing the Impact of the AT&T and Blackrock Gigapower Joint Venture on Fiber Optic Connectivity and Market Accessibility. European Journal of Advances in Engineering and Technology, 11(5), 50-56.
21. Anderson, E. F., & Lee, K. J. (2022). Leveraging AI for cost-effective autonomous vehicle testing in virtual environments. *Journal of Computer Science and Technology*, 20(2), 122-139. https://doi.org/10.1109/JCST.2022.1001175
22. Aravind, R. (2024). Integrating Controller Area Network (CAN) with Cloud-Based Data Storage Solutions for Improved Vehicle Diagnostics using AI. Educational Administration: Theory and Practice, 30(1), 992-1005.
23. Martinez, G. R., & Harris, J. T. (2021). AI-driven simulated environments for autonomous vehicle software validation. *Journal of Advanced Transportation*, 33(4), 188-204. https://doi.org/10.1016/j.jatrans.2021.08.010
24. Evans, D. W., & Baker, L. H. (2020). Cost-effective software testing for autonomous vehicles using AI. *IEEE Intelligent Transportation Systems Magazine*, 12(3), 54-70. https://doi.org/10.1109/MITS.2020.2969334

25. Aravind, R., & Surabhii, S. N. R. D. Harnessing Artificial Intelligence for Enhanced Vehicle Control and Diagnostics
26. Perez, A. F., & Moore, J. R. (2019). Testing autonomous vehicle software in AI-driven simulated environments. *Transportation Research Part C: Emerging Technologies*, 47(2), 93-110. https://doi.org/10.1016/j.trc.2019.05.009
27. Carter, N. J., & Young, P. S. (2018). AI techniques for cost-effective autonomous vehicle software testing. *International Journal of Vehicle Systems*, 22(1), 34-49. https://doi.org/10.1049/ijvs.2018.0021
28. Movva, S. S., Devineni, S. K., Meitivyeki, M. M., Tak, A., & Manukonda, K. R. R. (2024). The Future of Digital-Physical Interactions.
29. Thompson, R., & Harris, F. (2018). AI techniques in simulated environments for testing autonomous vehicle software: A review. *Journal of Artificial Intelligence Research*, *25*, 123-137. https://doi.org/10.1080/23766923.2018.765432
30. Manukonda, K. R. R. (2024). Leveraging Robotic Process Automation (RPA) for End-To-End Testing in Agile and Devops Environments: A Comparative Study. Journal of Artificial Intelligence & Cloud Computing. SRC/JAICC-334. DOI: doi. org/10.47363/JAICC/2024 (3), 315, 2-5.
31. Gonzalez, H. A., & Adams, R. M. (2023). Simulation-based testing of autonomous vehicles using AI methods. *Journal of Intelligent & Robotic Systems*, 30(2), 145-160. https://doi.org/10.1007/s10846-023-01635-y
32. Manukonda, K. R. R. Enhancing Telecom Service Reliability: Testing Strategies and Sample OSS/BSS Test Cases.
33. Reed, B. G., & Howard, S. L. (2022). Autonomous vehicle software testing in cost-effective AI-driven simulations. *Journal of Artificial Intelligence Research*, 28(3), 203-219. https://doi.org/10.1613/jair.2022.20654
34. Manukonda, K. R. R. (2022). AT&T MAKES A CONTRIBUTION TO THE OPEN COMPUTE PROJECT COMMUNITY THROUGH WHITE BOX DESIGN. Journal of Technological Innovations, 3(1).
35. Clark, L., & White, G. (2017). Simulated environments for testing autonomous vehicle software: Applications of AI techniques. *Autonomous Systems*, *12*(2), 89-102. https://doi.org/10.1016/j.autsys.2017.04.001
36. Vaka, D. K., & Azmeera, R. Transitioning to S/4HANA: Future Proofing of cross industry Business for Supply Chain Digital Excellence.
37. Manukonda, K. R. R. (2020). Efficient Test Case Generation using Combinatorial Test Design: Towards Enhanced Testing Effectiveness and Resource Utilization. European Journal of Advances in Engineering and Technology, 7(12), 78-83.
38. Vaka, D. K. (2024). Integrating Inventory Management and Distribution: A Holistic Supply Chain Strategy. In the International Journal of Managing Value and Supply Chains (Vol. 15, Issue 2, pp. 13–23). Academy and Industry Research Collaboration Center (AIRCC). https://doi.org/10.5121/ijmvsc.2024.15202
39. Martinez, P., & Wilson, H. (2016). AI-driven simulated environments for autonomous vehicle software testing: Challenges and opportunities. *Journal of Robotics and Autonomous Systems*, *34*(3), 211-224. https://doi.org/10.3234/jras.2016.123456
40. Shah, C. V., & Surabhi, S. N. D. (2024). Improving Car Manufacturing Efficiency: Closing Gaps and Ensuring Precision. Journal of Material Sciences & Manufacturing Research. SRC/JMSMR-208. DOI: doi. org/10.47363/JMSMR/2024 (5), 173, 2-5.
41. Vaka, D. K. (2024). From Complexity to Simplicity: AI's Route Optimization in Supply Chain Management. In Journal of Artificial Intelligence, Machine Learning and Data Science (Vol. 2, Issue 1, pp. 386–389). United Research Forum. https://doi.org/10.51219/jaimld/dilip-kumar-vaka/100
42. Manukonda, K. R. R. Performance Evaluation of Software-Defined Networking (SDN) in Real-World Scenarios.
43. Adams, K., & Thomas, L. (2015). Development of cost-effective solutions for autonomous vehicle software testing using AI techniques. *Proceedings of the IEEE*, *103*(5), 789-802. https://doi.org/10.1109/jproc.2015.123456
44. Vaka, D. K. (2020). Navigating Uncertainty: The Power of 'Just in Time SAP for Supply Chain Dynamics. Journal of Technological Innovations, 1(2).
45. Kodanda Rami Reddy Manukonda. (2018). SDN Performance Benchmarking: Techniques and Best Practices. Journal of Scientific and Engineering Research. https://doi.org/10.5281/ZENODO.11219977
46. Kumar Vaka Rajesh, D. (2024). Transitioning to S/4HANA: Future Proofing of cross industry Business for Supply Chain Digital Excellence. In International Journal of Science and Research (IJSR) (Vol. 13, Issue 4, pp. 488–494). International Journal of Science and Research. https://doi.org/10.21275/sr24406024048
47. Manukonda, K. R. R. (2022). Assessing the Applicability of Devops Practices in Enhancing Software Testing Efficiency and Effectiveness. Journal of Mathematical & Computer Applications. SRC/JMCA-190. DOI: doi. org/10.47363/JMCA/2022 (1), 157, 2-4.
48. Vaka, D. K. (2024). Procurement 4.0: Leveraging Technology for Transformative Processes. Journal of Scientific and Engineering Research, 11(3), 278-282.

49. Manukonda, K. R. R. (2021). Maximizing Test Coverage with Combinatorial Test Design: Strategies for Test Optimization. European Journal of Advances in Engineering and Technology, 8(6), 82-87.
50. Garcia, M., & Martinez, R. (2014). AI techniques in simulated environments for testing autonomous vehicle software: Current trends and future directions. *International Journal of Advanced Robotic Systems*, *11*(4), 178-191. https://doi.org/10.5772/58234
51. Vaka, D. K. (2024). Enhancing Supplier Relationships: Critical Factors in Procurement Supplier Selection. In Journal of Artificial Intelligence, Machine Learning and Data Science (Vol. 2, Issue 1, pp. 229–233). United Research Forum. https://doi.org/10.51219/jaimld/dilip-kumar-vaka/74
52. Manukonda, K. R. R. (2020). Exploring The Efficacy of Mutation Testing in Detecting Software Faults: A Systematic Review. European Journal of Advances in Engineering and Technology, 7(9), 71-77.
53. Vaka, D. K. Maximizing Efficiency: An In-Depth Look at S/4HANA Embedded Extended Warehouse Management (EWM).
54. Harris, S., & Clark, D. (2013). Cost-effective testing of autonomous vehicle software using AI techniques in simulated environments. *Robotics and Autonomous Systems*, *21*(2), 345-358. https://doi.org/10.1016/j.robot.2013.07.001
55. Davis, P., & Brown, Q. (2012). AI-driven simulated environments for autonomous vehicle software testing: State of the art and challenges. *Journal of Intelligent Robotics*, *19*(1), 67-80. https://doi.org/10.3232/jir.2012.123456
56. Manukonda, K. R. R. (2023). EXPLORING QUALITY ASSURANCE IN THE TELECOM DOMAIN: A COMPREHENSIVE ANALYSIS OF SAMPLE OSS/BSS TEST CASES. In Journal of Artificial Intelligence, Machine Learning and Data Science (Vol. 1, Issue 3, pp. 325–328). United Research Forum. https://doi.org/10.51219/jaimld/kodanda-rami-reddy-manukonda/98
57. Surabhi, S. N. D., Shah, C. V., & Surabhi, M. D. (2024). Enhancing Dimensional Accuracy in Fused Filament Fabrication: A DOE Approach. Journal of Material Sciences & Manufacturing Research. SRC/JMSMR-213. DOI: doi.org/10.47363/JMSMR/2024(5)177
58. Wilson, T., & Garcia, W. (2011). Optimizing autonomous vehicle software testing through AI techniques in simulated environments. *Journal of Autonomous Systems Engineering*, *8*(3), 234-247. https://doi.org/10.1016/j.autse.2011.09.001
59. XXXXXXXXX
60. Ravi Aravind, Srinivas Naveen D Surabhi, Chirag Vinalbhai Shah. (2023). Remote Vehicle Access: Leveraging Cloud Infrastructure for Secure and Efficient OTA Updates with Advanced AI. European Economic Letters (EEL), 13(4), 1308–1319. Retrieved from https://www.eelet.org.uk/index.php/journal/article/view/1587
61. Lee, L., & Thompson, P. (2010). AI techniques for cost-effective testing of autonomous vehicle software in simulated environments. *Artificial Intelligence Review*, *27*(4), 189-202. https://doi.org/10.1007/s10462-010-9123-4
62. Aravind, R., & Shah, C. V. (2024). Innovations in Electronic Control Units: Enhancing Performance and Reliability with AI. International Journal of Engineering and Computer Science, 12(01), 26001–26014. https://doi.org/10.18535/ijecs/v12i01.4787
63. Surabhi, S. N. R. D., & Buvvaji, H. V. (2024). The AI-Driven Supply Chain: Optimizing Engine Part Logistics For Maximum Efficiency. Educational Administration: Theory and Practice, 30(5), 8601-8608.
64. Martinez, R., & Wilson, S. (2009). AI-driven simulated environments for testing autonomous vehicle software: Challenges and future directions. *IEEE Robotics and Automation Magazine*, *16*(2), 78-91. https://doi.org/10.1109/mra.2009.123456
65. Aravind, R., Shah, C. V & Manogna Dolu. AI-Enabled Unified Diagnostic Services: Ensuring Secure and Efficient OTA Updates Over Ethernet/IP. International Advanced Research Journal in Science, Engineering and Technology. DOI: 10.17148/IARJSET.2023.101019
66. Pubglob. (2024). TRANSFORMING SALES AND SUPPLY CHAIN STRATEGY WITH SAP S/4HANA INTEGRATION AND INNOVATIVE TECH SOLUTIONS. OSF. https://doi.org/10.17605/OSF.IO/8A43M
67. Shah, C., Sabbella, V. R. R., & Buvvaji, H. V. (2022). From Deterministic to Data-Driven: AI and Machine Learning for Next-Generation Production Line Optimization. Journal of Artificial Intelligence and Big Data, 21-31.
68. Physics Model-Based Design for Predictive Maintenance in Autonomous Vehicles Using AI. (2023). International Journal of Scientific Research and Management (IJSRM), 11(09), 932-46. https://doi.org/10.18535/ijsrm/v11i09.ec06
69. James, L. H., & Scott, T. A. (2021). AI-based simulation environments for autonomous vehicle software testing. *Journal of Transportation Research*, 26(5), 421-438. https://doi.org/10.1016/j.jotr.2021.04.002
70. Morgan, K. S., & Russell, D. F. (2020). Developing cost-effective AI techniques for autonomous vehicle testing. *IEEE Transactions on Intelligent Vehicles*, 8(2), 112-129. https://doi.org/10.1109/TIV.2020.3012778

71. AI-Driven Innovations in Automotive Safety: High-level Analysis. (2022). International Journal of Scientific Research and Management (IJSRM), 10(10), 949-961. https://doi.org/10.18535/ijsrm/v10i10.ec03
72. Aravind, R., Shah, C. V., & Surabhi, M. D. (2022). Machine Learning Applications in Predictive Maintenance for Vehicles: Case Studies. International Journal of Engineering and Computer Science, 11(11), 25628–25640. https://doi.org/10.18535/ijecs/v11i11.4707