A Generalized Metamorphic Testing Platform for Regression Models in Autonomous Driving Systems

An Undergraduate Project Proposal Report Submitted to the

Department of Electrical and Information Engineering
Faculty of Engineering
University of Ruhuna
Sri Lanka

in partial fulfillment of the requirements for the

Degree of Bachelor of the Science of Engineering Honours

by

Akurana B.N.T.M.	–	EG/2020/3812
Pabasara P.M.G.T	–	EG/2021/4701
Peiris P.R.S	–	EG/2021/4706
Perera G.C.M	–	EG/2021/4707

. . . .

Dr. Kaveen Liyanage
(Supervisor)

Mrs. Sithara Mahagama
(Co-Supervisor)

Abstract

In order to carry out crucial functions including trajectory prediction, lane maintenance, steering control, and vehicle motion planning, autonomous driving systems are depending more and more on machine learning-based regression models. Even while these models have made great strides, it is still very difficult to guarantee their dependability and safety in actual driving situations due to the test oracle problem. A test oracle is defined as a mechanism capable of systematically verifying the correctness of a test result for any given input; however, a fundamental challenge arises when such an oracle either does not exist or is too expensive to be used. This problem presents a significant hurdle because it creates a verification difficulty, where it is impossible to clearly determine whether the continuous-valued output of a program has passed or failed for a specific input. Furthermore, this leads to methodological restrictions, as the lack of an oracle limits the effectiveness of traditional test case selection strategies that rely on known outcomes. Consequently, while traditional testing is inadequate for ensuring reliable behavior in unexpected circumstances, alternative approaches like metamorphic testing seek to alleviate this problem by using metamorphic relations properties represented as relations among inputs and outputs of multiple executions to verify program correctness instead of relying on a single, predetermined outcome [1].

By validating software behavior through metamorphic relations that specify expected interactions between many system executions under deliberately altered inputs, Metamorphic Testing (MT) has emerged as a promising solution to the oracle problem. The lack of a universal, reusable, and scalable testing framework has prevented MT from being widely used in autonomous driving regression models, despite its successful application in other domains.

Regression models used in autonomous driving systems are the focus of this project's design and implementation of a generic metamorphic testing platform. The suggested method allows for the systematic creation and assessment of test cases without the need for explicit test oracles by introducing a model-agnostic metamorphic testing template that abstracts common metamorphic features. Regression models for real-world autonomous driving and proof-of-concept tests are used to validate the framework, which is constructed as a modular Python-based system. A web-based interface is also created to improve usability and accessibility for practitioners and researchers.
The suggested platform seeks to enhance the robustness, dependability, and real-world preparedness of autonomous driving regression models by decreasing manual testing effort and expanding test coverage. This will help make the deployment of autonomous car technology safer.

List of Acronyms

ADAS	Advanced Driver Assistance Systems
AI	Artificial Intelligence
API	Application Programming Interface
Euro NCAP	European New Car Assessment Programme
GPR	Gaussian Process Regression
IEEE	Institute of Electrical and Electronics Engineers
LKAS	Lane Keeping Assist System
ML	Machine Learning
MPC	Model Predictive Control
MR	Metamorphic Relation
MT	Metamorphic Testing
SAE	Society of Automotive Engineers
SGPR	Sparse Gaussian Process Regression
WEF	World Economic Forum

Contents

List of Figures

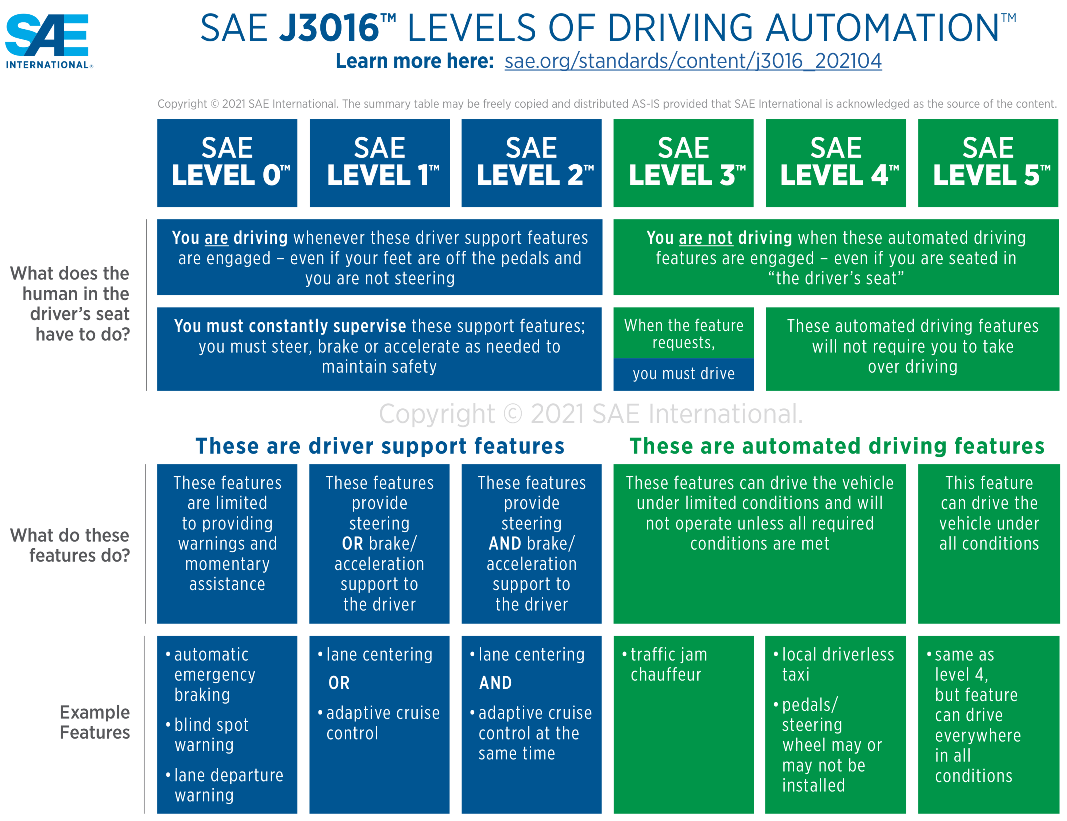
List of Tables

Chapter 1

Introduction

With growing use in early autonomous vehicle platforms and commercial driver assistance systems, autonomous driving has become one of the most significant uses of artificial intelligence. The World Economic Forum states that although widespread full autonomy is still limited by technical, legal, and safety issues, vehicle automation technologies are already present on public roads, especially at Levels 2 and 2+, and are anticipated to gradually expand toward higher levels of autonomy over the next ten years [7]. Current autonomous driving systems still have trouble operating dependably in a variety of real-world scenarios, despite tremendous advancements in perception, prediction, and control models [8] [9].

1.1 Background


SAE J3016™ LEVELS OF DRIVING AUTOMATION™

Learn more here: sae.org/standards/content/j3016_202104

	SAE LEVEL 0™	SAE LEVEL 1™	SAE LEVEL 2™	SAE LEVEL 3™	SAE LEVEL 4™	SAE LEVEL 5™
What does the human in the driver's seat have to do?	**You <u>are</u> driving** whenever these driver support features are engaged – even if your feet are off the pedals and you are not steering			**You <u>are not</u> driving** when these automated driving features are engaged – even if you are seated in "the driver's seat"		
	You must constantly supervise these support features; you must steer, brake or accelerate as needed to maintain safety			When the feature requests, you must drive	These automated driving features will not require you to take over driving	
	These are driver support features			These are automated driving features		
What do these features do?	These features are limited to providing warnings and momentary assistance	These features provide steering **OR** brake/ acceleration support to the driver	These features provide steering **AND** brake/ acceleration support to the driver	These features can drive the vehicle under limited conditions and will not operate unless all required conditions are met		This feature can drive the vehicle under all conditions
Example Features	• automatic emergency braking • blind spot warning • lane departure warning	• lane centering **OR** • adaptive cruise control	• lane centering **AND** • adaptive cruise control at the same time	• traffic jam chauffeur	• local driverless taxi • pedals/ steering wheel may or may not be installed	• same as level 4, but feature can drive everywhere in all conditions

Figure 1.1: SAE International Levels of Driving Automation (J3016 standard), illustrating the progression from no automation (Level 0) to full autonomy (Level 5) [2].

The SAE J3016 standard specifies six levels of driving automation, from no automation (Level 0) to complete automation (Level 5), as seen in Figure 1.1. Currently, the majority of commercially deployed systems function at Levels 2 and 2+, requiring constant human supervision and only limited automation. This highlights how crucial sound testing procedures are to guaranteeing dependability and safety in practical implementation.

Insufficient and unrealistic testing is a significant barrier limiting the robustness of autonomous driving models. Autonomous systems function in extremely dynamic, complicated contexts where it is not feasible to list and verify every scenario before to deployment. For machine learning-based systems that provide continuous outputs like trajectories, steering angles, or acceleration values, this problem is commonly known as the oracle problem, where it is challenging or impossible to discern the proper output for a given test input. As previously mentioned, the lack of trustworthy test oracles is a common problem for machine learning applications, such as autonomous driving and advanced driver assistance systems (ADAS) [10], [8].

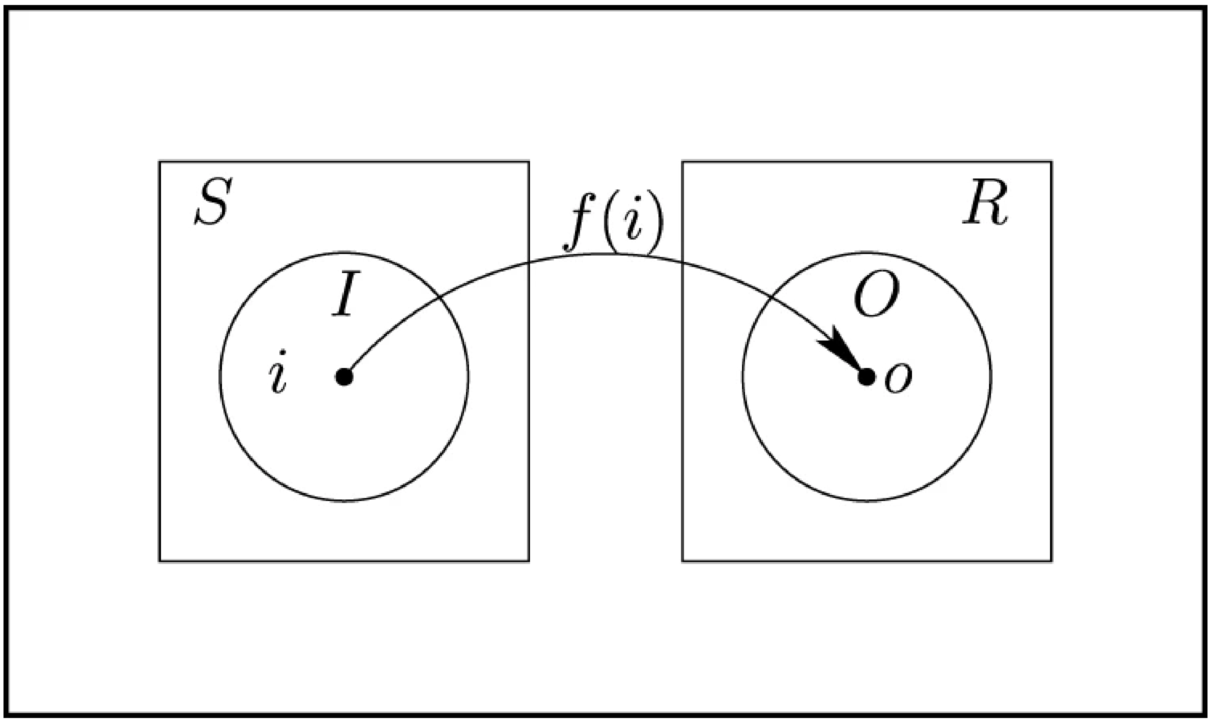


Figure 1.2: Illustration of the test oracle problem in software testing, highlighting the challenge of determining correct outputs without explicit expected results [3].

Figure 1.2 illustrates the challenge presented by the oracle problem and shows how difficult it is to determine proper outputs when clear expected results are not given. Alternative testing methods like metamorphic testing are necessary because this problem is especially common in machine learning-based autonomous driving systems that produce continuous outputs.

Chen first proposed Metamorphic Testing (MT) in 1998, and it offers a practical solution to the oracle problem. MT determines whether a system satisfies required qualities, called metamorphic relations (MRs), spanning several executions with related inputs rather than requiring explicit expected outcomes. Even when the correct output is unknown, errors can be identified if these relationships are broken. Numerous fields, such as navigation software, search engines, cybersecurity, machine learning systems, and autonomous driving applications, have effectively used MT [11], [9]. Geometric-transformation-based metamorphic relations have proven particularly successful in exposing discrepancies under perspective, spatial, or environmental alterations in the context of autonomous systems.

1.2 Problem Statement

However, using metamorphic testing in practice is still difficult despite its benefits. For each model and situation, current methods usually require developers to manually create metamorphic relations and apply unique test scripts. Due to irregular test design or inadequate domain coverage, this method is laborious, prone to errors, and may still overlook important edge situations. Because of this, metamorphic testing is currently neither readily available or methodically incorporated into the process of developing autonomous driving regression models.

1.3 Objectives and Scope

Designing and implementing a metamorphic testing environment specifically for autonomous driving regression models is the aim of this project. The goal of the suggested platform is to make it easier to define, implement, and assess metamorphic relations so that model developers can test both new and old autonomous driving models more successfully. This study aims to improve model robustness, dependability, and real-world readiness by streamlining the testing procedure and lowering dependency on manually written scripts. In the end, the project closes a significant gap between model creation and safe practical deployment by enhancing software quality assurance procedures for autonomous driving systems.

Chapter 2

Literature Review

2.1 Regression-Based Models in Autonomous Driving Systems

For autonomous driving systems to operate safely and effectively, key features like obstacle detection, decision-making, lane centerline management, and overtaking are essential. Because of their powerful representation learning capabilities, deep neural network-based methods have been increasingly used in recent years to tackle these issues. However, due to their interpretability, uncertainty estimates, and robust performance on structured, continuous data, regression-based models especially Gaussian Process-based approaches remain very important for autonomous driving applications [8]. Among these, Sparse Gaussian Process Regression (SGPR) has drawn interest as a scalable substitute for conventional Gaussian Processes that preserves prediction accuracy while permitting real-time inference.

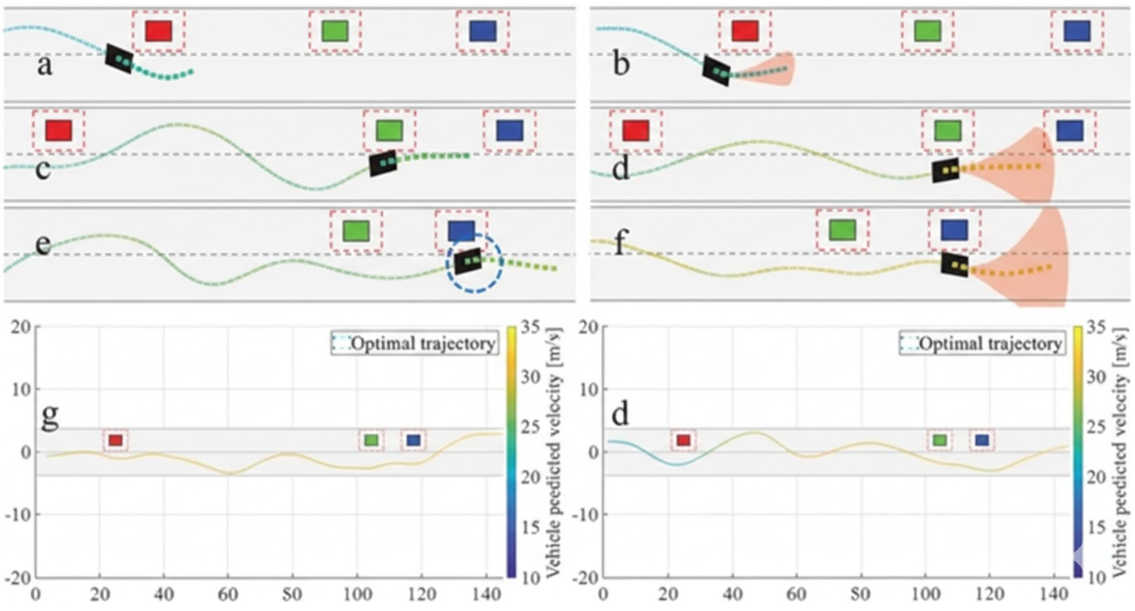


Figure 2.1: Example of Gaussian Process-based trajectory prediction in autonomous driving, showing predicted trajectories with uncertainty bounds for overtaking maneuvers [4].

An example of Gaussian Process-based trajectory prediction in an autonomous driving setting is shown in Figure 2.1, which shows expected vehicle trajectories and related uncertainty bounds. Because they can give uncertainty estimates, which are essential for making safety-aware decisions in dynamic driving conditions, these regression-based models are very useful.

Model Predictive Control (MPC) frameworks for autonomous driving tasks, such as overtaking maneuvers, have effectively incorporated regression models like SGPR. For instance, SGPR is used by Gaussian Process-based MPC techniques to simulate vehicle dynamics and surrounding vehicle behavior in a variety of traffic scenarios. Experimental results show enhanced trajectory prediction accuracy and control stability across various driving circumstances, and these models are assessed using specified testing datasets [9]. Similarly, SGPR-based trajectory prediction models can outperform deep learning approaches in terms of inference time, interpretability, and adaptability, especially when combined with geometric transformations like translation and rotation equivariance, according to recent IEEE Intelligent Vehicles Symposium research [4].

2.2 Limitations of Conventional Testing Approaches in Autonomous Driving

The testing and validation procedures used in many current autonomous driving projects, however, are frequently constrained to a small number of predetermined scenarios and datasets, despite their proven efficacy. It is challenging to fully capture the complexity and diversity of real-world driving conditions using conventional testing techniques because they include unforeseen impediments, uncommon edge cases, and extremely dynamic traffic interactions. As a result, even regression-based autonomous driving models that function effectively in carefully regulated experimental settings could behave erratically when used in novel or unusual circumstances. There is an increasing demand for either significantly larger and more varied test datasets or more generalized and systematic testing approaches that may assess model performance outside of predetermined situations in order to increase robustness and facilitate efficient fine-tuning of such models.

2.3 Metamorphic Testing: Previous Work

Metamorphic testing (MT) was first introduced by Chen (1998) as a systematic software testing technique designed to address the oracle problem, a common challenge in domains where it is difficult or impossible to determine the correct output for a given input in advance. Metamorphic testing examines the links between numerous executions of the same program under altered inputs to verify program correctness rather than depending on explicit predicted results. These anticipated interactions, referred to as metamorphic relations (MRs), reflect essential characteristics that the system being tested needs to meet. Even without a conventional test oracle, a problem is exposed if an MR is broken. MT greatly decreases reliance on explicit oracles by reorienting the attention from output correctness to relational consistency. It has been especially successful for complex, data-driven systems when traditional testing methods are inadequate [11].

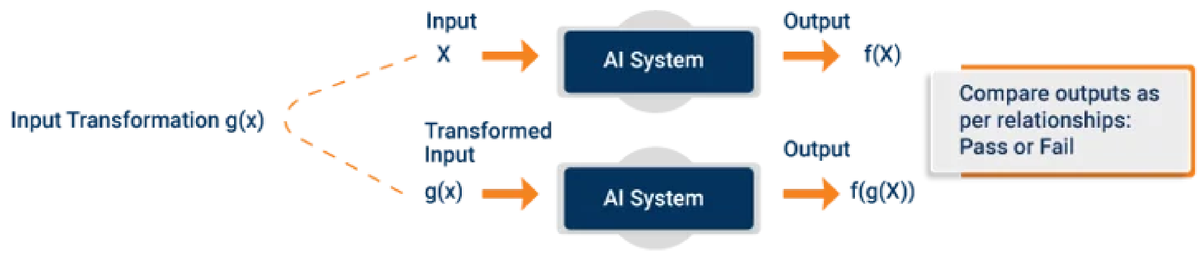

Input Transformation g(x)
Input
X
AI System
Output
f(X)
Transformed Input
g(x)
AI System
Output
f(g(X))
Compare outputs as per relationships: Pass or Fail

Figure 2.2: Conceptual overview of metamorphic testing, illustrating the generation of follow-up test cases from source test cases using metamorphic relations to validate system behavior without an explicit test oracle [5].

Instead of depending on explicit expected outputs, metamorphic testing uses preset metamorphic relations to generate follow-up test cases from an initial source test case, as shown in Figure 2.2. The graphic illustrates how the oracle problem in complex systems, where accurate outputs are hard to predict ahead of time, is addressed by using linkages between many executions to check system behavior.

2.4 Metamorphic Testing in Machine Learning and Autonomous Driving Systems

In order to overcome the shortcomings of conventional testing methods, metamorphic testing (MT) is being used more and more in machine learning-based systems and autonomous driving applications. Previous studies show that MT is especially useful for Advanced Driver Assistance Systems (ADAS), because complex settings and continuous system outputs make it challenging to define exact test oracles. For instance, Lane Keeping Assist Systems (LKAS) under Euro NCAP driving scenarios have been verified and validated using geometric-transformation-based metamorphic relations (MRs) in recent studies.

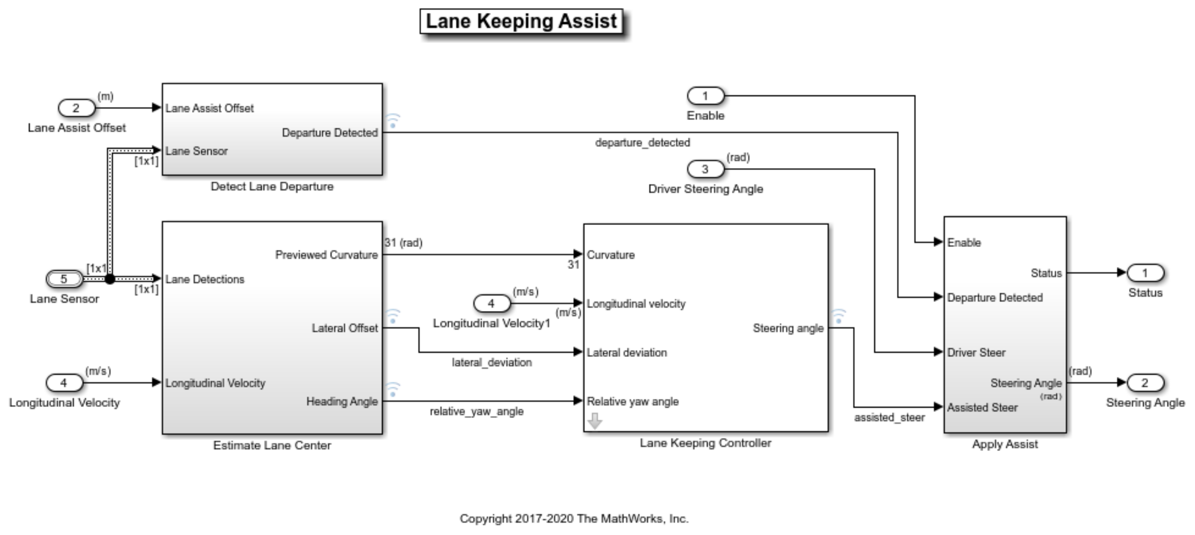


Figure 2.3: Diagram of a Lane Keeping Assist System (LKAS) in autonomous driving, illustrating lane detection, departure warning, and corrective steering [6].

A Lane Keeping Assist System (LKAS) usually includes of lane detection, vehicle position assessment, and corrective steering control components, as shown in Figure 2.3. Because geometric transformations and symmetry-based metamorphic relations can be used to verify system consistency under various driving conditions, LKAS's modular structure makes it a good choice for metamorphic testing.

Researchers were able to successfully identify important system faults that were missed by standard testing alone by employing symmetry- and rotation-based MRs to methodically create a large number of follow-up test cases from existing situations. Without depending on explicit intended outputs, these methods use relationships between numerous executions—such as invariant vehicle behavior under scene rotations—to indirectly assess system correctness [11] [9]. In addition to geometric transformations, other metamorphic relations explored in the literature include translation invariance, trajectory-preserving transformations, environmental condition variations, and sensor symmetry relations, all of which are well-suited for testing perception and regression-based control models in autonomous systems. However, despite their proven efficacy, the majority of current MT implementations in autonomous driving are still closely linked to particular case studies, simulation platforms, or manually created MRs, which restricts their scalability and reuse across many models and development pipelines.

2.5 Gap Identification: Limitations of Existing Metamorphic Testing Approaches

Although the fact that metamorphic testing (MT) has been demonstrated to be a useful tool for resolving the oracle problem and exposing hidden flaws in autonomous driving and machine learning systems, its practical use is still restricted because of a number of methodological and structural issues. The majority of current research uses ad hoc metamorphic test scripts, in which metamorphic relations (MRs) are created by hand and closely linked to particular models, simulation systems, or experimental setups [11] [9]. Geometric-transformation-based MRs like rotation and symmetry can effectively reveal critical system failures, as shown in earlier work on ADAS and Lane Keeping Assist Systems (LKAS). Nevertheless, these MRs are usually implemented in a case-specific and non-reusable manner, requiring significant manual effort for each new system or platform.

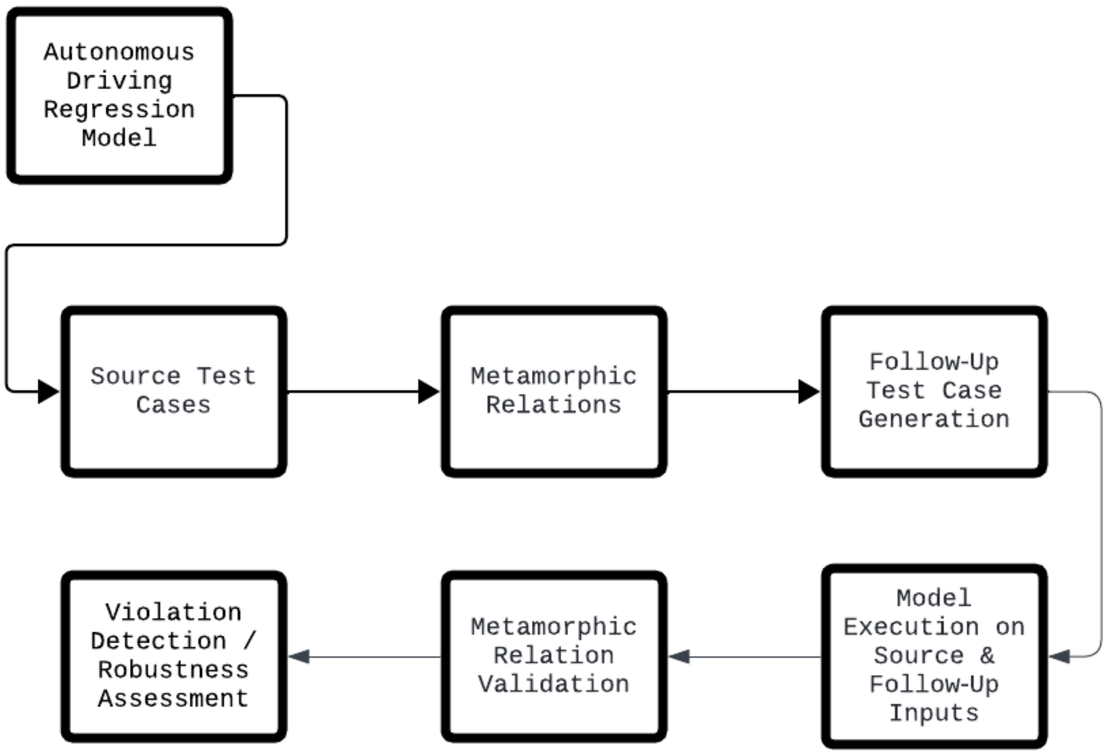
Additionally, the lack of abstraction and automation restricts MT's scalability, even though it has been used in machine learning models and autonomous driving systems using transformations including rotation, translation, scaling, and symmetry-based relations [9] [8]. As demonstrated by simulation-based ADAS testing environments, current MT implementations frequently need developers to have a thorough understanding of both the system being tested and the underlying simulation tools, which results in considerable development and maintenance overhead. This restriction is especially noticeable for regression-based autonomous driving models, where the design and implementation of MRs are made more difficult by continuous outputs, intricate environmental interactions, and dynamic scenarios.

The LKAS and Euro NCAP case studies demonstrate that MT can find errors that are missed by traditional testing; however, wider adoption across models and development pipelines is hindered by the lack of a generic, user-friendly metamorphic testing platform. Thus, a systematic, reusable, and model-agnostic metamorphic testing approach specifically designed for autonomous driving regression models is clearly lacking. Reducing manual labor, increasing test coverage, and facilitating scalable and reliable validation all depend on closing this gap, which immediately motivates the suggested strategy and logically leads to the methodology described in the following section.

Chapter 3

Methodology

To develop a generalized and reusable metamorphic testing framework applicable to a wide range of regression models in autonomous driving, this work follows a structured, multi-stage methodology as outlined below.



Autonomous Driving Regression Model
Source Test Cases
Metamorphic Relations
Follow-Up Test Case Generation
Model Execution on Source & Follow-Up Inputs
Metamorphic Relation Validation
Violation Detection / Robustness Assessment

Figure 3.1: Overall workflow of the proposed metamorphic testing framework for autonomous driving regression models.

An overview of the suggested metamorphic testing framework workflow is given in Figure 3.1, which shows the steps from regression model analysis to the execution and evaluation of metamorphic tests. The workflow demonstrates how metamorphic relations are used to change source test cases, which are then assessed to find violations without the need for conventional test oracles.

3.1 Comprehensive Analysis of Regression Models and Metamorphic Properties

The first stage involves a comprehensive study of regression models commonly employed in autonomous driving and related cyber-physical systems. These include, but are not limited to, linear and nonlinear regression models, Gaussian Process Regression (GPR), Sparse Gaussian Process Regression (SGPR), neural-network-based regression models, and hybrid learning-based regression approaches. For each category, existing metamorphic testing strategies and applicable metamorphic relations (MRs) such as geometric transformations, symmetry relations, monotonicity, invariance, and consistency relations are systematically analyzed. This analysis ensures that the proposed framework remains model-agnostic and is capable of supporting diverse regression behaviors and output characteristics typical of autonomous driving systems.

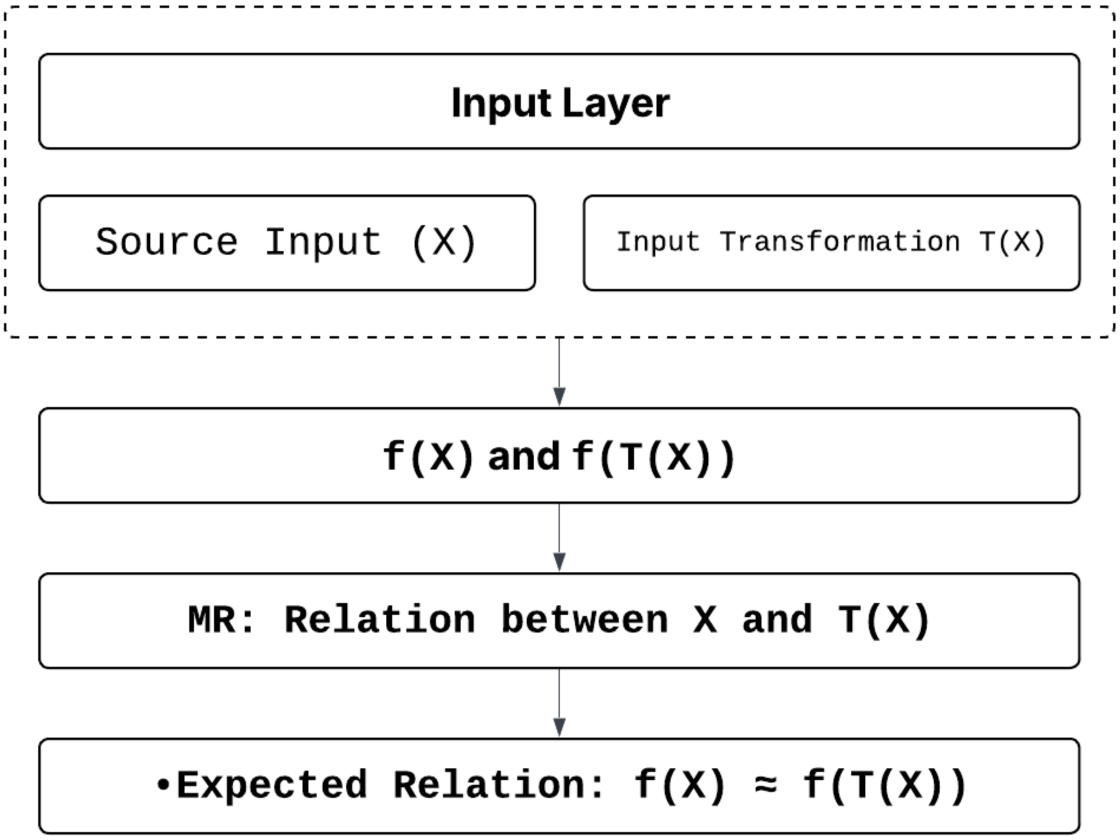

Input Layer
Source Input (X)
Input Transformation T(X)
f(X) and f(T(X))
MR: Relation between X and T(X)
•Expected Relation: f(X) ≈ f(T(X))

Figure 3.2: Structure of the generalized metamorphic testing template illustrating source inputs, transformations, and relational output constraints.

As shown in Figure 3.2, the generalized metamorphic testing template defines the relationship between source inputs, transformed follow-up inputs, and expected relational constraints on regression outputs. This abstraction enables the framework to remain independent of specific regression algorithms while supporting diverse model types and autonomous driving applications.

3.2 Design of a Generalized Metamorphic Testing Template

Based on the insights obtained from the regression model analysis, a generalized mathematical and conceptual metamorphic testing template is designed. The template formally defines:

- The structure of source and follow-up test inputs
- A standardized representation of metamorphic relations
- Expected relational constraints between regression outputs

The proposed template is independent of any specific regression algorithm. Instead, it allows instantiation for different regression models by specifying appropriate metamorphic relations and input transformation rules, thereby enabling reuse across multiple model types and applications.

3.3 Proof of Concept: Demonstration on a Representative Regression Model

The suggested metamorphic testing template is created and used to a typical regression model used in autonomous driving as a proof of concept. To illustrate the viability and accuracy of the method, a rudimentary regression-based model such as a steering angle prediction or trajectory estimate model is chosen.

In this proof-of-concept study, simulated driving scenarios are used to build a collection of source test inputs. Well-defined metamorphic relations, such as geometric transformations (e.g., translation or rotation of trajectories), symmetry relations, and invariance properties, are then used to construct subsequent test cases. Both source and follow-up inputs are used to run the regression model, and the results are compared against the expected metamorphic constraints.

Observed violations of metamorphic relations indicate potential robustness or correctness issues in the regression model, while satisfied relations provide increased confidence in model behavior. This proof-of-concept implementation demonstrates that the generalized metamorphic testing template can be practically applied to real regression models without requiring model-specific test oracles, thereby validating the core idea of the proposed framework.

3.4 Validation Using Real-World Autonomous Driving Regression Models

Following the proof-of-concept demonstration, real-world regression models used in autonomous driving tasks like lane keeping, trajectory prediction, steering angle estimation, and vehicle control are used to further validate the suggested metamorphic testing template. Based on the specified relations, metamorphic test cases are automatically created. The outputs are then examined to find any deviations from the expected metamorphic qualities. This validation stage shows how the methodology may evaluate regression models' accuracy and resilience under realistic and methodically altered driving conditions.

3.5 Implementation of a Python-Based Metamorphic Testing Framework

The generalized metamorphic testing template is developed as a modular and expandable Python-based framework following validation. The implementation offers:

- Application Programming Interfaces (APIs) for integrating arbitrary regression models
- Configurable definitions of metamorphic relations
- Automated test case generation and execution
- Output comparison and metamorphic violation reporting

This framework greatly reduces manual labor and increases repeatability by allowing researchers and practitioners to apply metamorphic testing to regression models without creating ad hoc testing scripts.

3.6 System Architecture of the Proposed Framework

The overall system architecture is designed in a modular manner and consists of the following core components:

- Regression Model Interface
- Metamorphic Relation Engine
- Test Case Generator
- Execution and Evaluation Module
- Reporting and Visualization Module

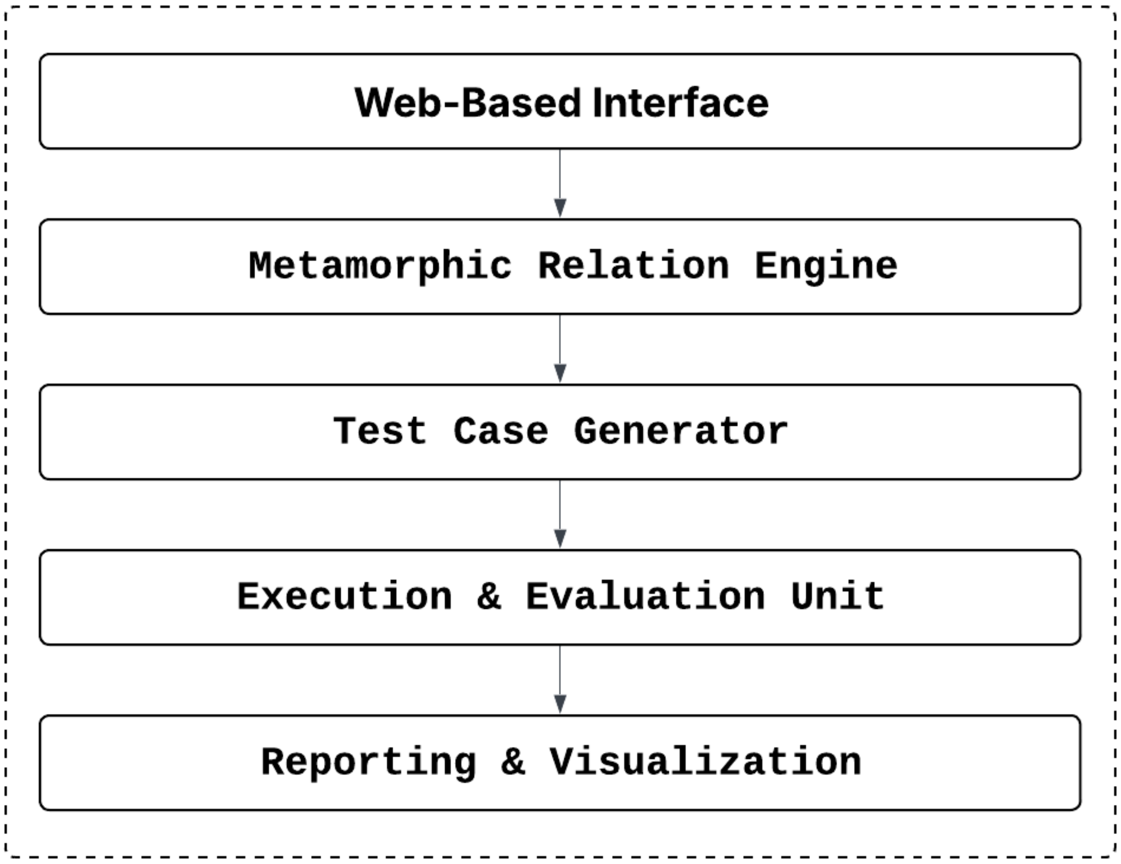

Web-Based Interface
Metamorphic Relation Engine
Test Case Generator
Execution & Evaluation Unit
Reporting & Visualization

Figure 3.3: System architecture of the proposed Python-based metamorphic testing framework for autonomous driving regression models.

The system architecture of the suggested Python-based metamorphic testing framework is shown in Figure 3.3. Scalability, reusability, and ease of expansion are ensured by the architecture's modular components, which include the regression model interface, metamorphic relation engine, test case generator, execution and evaluation module, and reporting and visualization module.

This architecture ensures scalability, reusability, and ease of extension to new regression models and autonomous driving scenarios.

3.7 Web-Based Platform for Accessibility and Public Use

To promote usability and broader adoption, a web-based interface is developed on top of the Python framework. The platform enables users to:

- Upload or connect regression models
- Select predefined or custom metamorphic relations
- Execute metamorphic tests through a graphical interface
- Visualize testing results and detected violations

Chapter 4

Timeline

The project has been divided into 18 sequential milestones, each of which corresponds to the planned project activities, in order to guarantee its effective completion within the allotted time period of 27 weeks (two semesters).

1. Identification of Metamorphic Properties and Relations

    - Identify suitable metamorphic properties for regression models.
    - Based on expected outputs and input transformations, define metamorphic relations.
    - Validate identified relations through theoretical analysis.

2. Design of Generalized Metamorphic Testing Template

    - Create a template for universal and reusable metamorphic testing.
    - Describe the processes for creating, executing, and validating tests.
    - Make sure it can be expanded to accommodate other regression models.

3. Proof of Concept (PoC) Development

    - Conduct a proof of concept to show that the suggested strategy is feasible.
    - Use sample regression models to validate important metamorphic linkages.
    - Examine PoC results and make any necessary design adjustments.

4. Development of Python-Based Metamorphic Testing Framework

    - Develop the core Python-based metamorphic testing framework.
    - Implement modules for the creation and execution of automated test cases.

- Integrate metamorphic relations into the framework.

5. Project Progress Presentation and VIVA
    - Prepare progress presentation highlighting completed work.
    - Present implementation progress to the evaluation panel.
    - Attend the progress VIVA and address feedback.

6. Submission of Project Progress Report
    - Prepare a detailed progress report documenting completed milestones.
    - Include implementation details and preliminary results.
    - Submit the progress report for assessment.

7. Experimental Evaluation
    - Create test configurations to assess the framework's efficacy.
    - Execute experiments using regression models.
    - Evaluate the performance and capability of fault detection by analyzing the data.

8. Web-Based Platform Development
    - Design and implement a web-based platform for the testing framework.
    - Enable user interaction for test execution and result visualization.
    - Integrate the web platform with the backend testing framework.

9. Documentation and Final Writing
    - Prepare comprehensive documentation of system design and implementation.
    - Write chapters of the final project report.
    - Verify adherence to the formatting requirements of the university.

10. Submission of Draft Final Report for Supervisor's Comments
    - Submit the draft final report to the supervisor for review.
    - Collect and analyze feedpback.

- Identify required revisions.

11. Submission of Final Project Report

    - Revise the final report based on supervisor feedback.
    - Complete the document by reviewing it.
    - Submit the completed final report.

12. Final Presentation and VIVA

    - Prepare final presentation summarizing the entire project.
    - Deliver the final presentation.
    - Attend the final VIVA and defend the project outcomes.

13. Final Demonstration

    - Demonstrate the complete metamorphic testing system.
    - Display the characteristics of the web-based platform and experimental findings.
    - Conclude the project with final evaluation and submission.

Table 4.1: Project Timeline and Milestones

PROJECT ACTIVITY	DURATION	PROJECT WEEK																										
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27
Identification of metamorphic properties & relations	3 weeks	■	■	■																								
Design of generalized metamorphic testing template	3 weeks			■	■	■																						
Proof of Concept (PoC)	4 weeks					■	■	■	■																			
Python-based MT framework	6 weeks								■	■	■	■	■	■														
Project Progress presentations and VIVA	1 week												■															
Submission of project progress report	1 week													■														
Experimental evaluation	4 weeks														■	■	■	■										
Web-based platform	3 weeks																		■	■	■							
Documentation & final writing	3 weeks																					■	■	■				
Submission of draft of the final report for supervisor's comments	1 week																								■			
Submission of Final report	1 week																									■		
Final presentation and VIVA	1 week																										■	
Final Demonstration	1 week																											■

Chapter 5

Conclusion

The growing popularity of autonomous driving systems has brought attention to the urgent need for dependable and methodical testing techniques that can guarantee robust and safe model behavior in a variety of real-world scenarios. Due to their continuous outputs and lack of trustworthy test oracles, regression-based machine learning models which are frequently employed in autonomous driving for tasks like trajectory prediction, steering control, and lane keeping present particular testing issues. The complexity and diversity of real-world driving situations are frequently not well captured by conventional testing methods, which reduces confidence in model deployment.

A generalized metamorphic testing platform created especially for regression models in autonomous driving systems is proposed in this study. Instead of depending on explicit expected outcomes, the suggested method uses metamorphic testing concepts to validate relational coherence across successive executions, so addressing the Oracle problem. Systematic, reusable, and scalable testing across a variety of regression models and autonomous driving applications is made possible by the development of a model-agnostic metamorphic testing framework.

The manual effort needed to define and run metamorphic tests is greatly decreased by the use of a modular Python-based framework and an intuitive web-based interface. The suggested platform shows its efficacy in detecting any robustness and consistency problems that might be missed by traditional testing techniques through proof-of-concept demonstrations and validation using real-world autonomous driving regression models.

All things considered, this work advances the robustness, dependability, and practicality of autonomous driving regression models. The suggested metamorphic testing platform facilitates the safer and more reliable deployment of autonomous driving technologies by bridging the gap between model development and systematic validation. It also offers a basis for further research and expansion in software testing for machine learning-based cyber-physical systems.

References

[1] H. Liu, F.-C. Kuo, D. Towey, and T. Y. Chen, "How effectively does metamorphic testing alleviate the oracle problem?" *IEEE Transactions on Software Engineering*, vol. 40, no. 1, pp. 4–22, 2014.

[2] SAE International. (2021) Sae j3016 levels of driving automation visual chart. [Online]. Available: https://www.sae.org/standards/content/j3016_202104/

[3] E. T. Barr *et al.*, "The oracle problem in software testing: A survey," *IEEE Transactions on Software Engineering*, 2015. [Online]. Available: https://ieeexplore.ieee.org/document/6963470

[4] H. Liu, K. Chen, and J. Ma, "Incremental learning-based real-time trajectory prediction for autonomous driving via sparse gaussian process regression," in *Proceedings of the IEEE Intelligent Vehicles Symposium (IV)*, 2024.

[5] QA Source. (2020) Metamorphic testing concept diagram. [Online]. Available: https://blog.qasource.com/what-is-metamorphic-testing

[6] MathWorks. (2024) Lane keeping assist with lane detection system diagram. [Online]. Available: https://www.mathworks.com/help/driving/ug/lane-keeping-assist-with-lane-detection.html

[7] World Economic Forum, "Autonomous vehicles: Timeline and roadmap ahead," 2025.

[8] X. Zhang *et al.*, "Machine learning testing: Survey, landscapes, and horizons," *IEEE Transactions on Software Engineering*, 2020.

[9] Z. Zhou *et al.*, "Metamorphic testing for autonomous driving systems," *IEEE Transactions on Software Engineering*, 2019 - 2021.

[10] E. T. Barr *et al.*, "The oracle problem in software testing," *IEEE Transactions on Software Engineering*, 2015.

[11] T. Y. Chen, "Metamorphic testing: A new approach for generating next test cases," in *Technical Report*, 1998.