JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue
JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)
An International Scholarly Open Access, Peer-reviewed, Refereed Journal

Detection of Lane and Speed Breaker Warning System for Autonomous Vehicles using Machine Learning Algorithm

[1]Varad Sunil Salunke, [2]Pushpak Sunil Rane, [3]Hrishikesh Nandu Kanade, [4]Harshal Chunilal Patel

[5]Prof. S.R. Bhujbal

[1]Student, [2] Student, [3] Student, [4] Student, [5]Head of Department
B.E. (Department of Computer Engineering)
P K Technical Campus, Pune, India

Abstract : Autonomous vehicles rely on a multitude of sensors and intelligent systems to navigate safely and efficiently. This paper presents a comprehensive Lane Detection and Speed Breaker Warning System designed to enhance the capabilities of autonomous vehicles using advanced machine learning algorithms. The primary goal is to improve the vehicle's perception of the road environment, specifically focusing on accurate lane detection and timely recognition of speed breakers. The proposed system integrates a combination of computer vision techniques and machine learning algorithms to achieve robust performance in real-world scenarios. For lane detection, a convolutional neural network (CNN) is employed to analyze camera inputs and identify lane boundaries. This enables the vehicle to precisely follow the road markings, ensuring safe navigation within lanes. To address the challenge of speed breaker detection, a machine learning model is trained on a diverse dataset containing images of roads with varying types and conditions of speed breakers. The model is designed to classify road segments and predict the presence of speed breakers ahead. When a speed breaker is detected, the system activates a warning mechanism to alert the autonomous vehicle, allowing it to adjust its speed and suspension settings accordingly. The effectiveness of the proposed system is evaluated through extensive simulations and real-world testing scenarios. The results demonstrate a significant improvement in lane-keeping accuracy and the ability to anticipate and respond to speed breakers proactively. The Lane and Speed Breaker Warning System contributes to the overall safety and reliability of autonomous vehicles, making them better equipped to handle diverse road conditions.

Keywords: **Autonomous Vehicles, Lane Detection, Speed Breaker Warning, Machine Learning, Convolutional Neural Network, Computer Vision.**

I. Introduction

In recent years, the advancement of autonomous vehicle technology has been at the forefront of innovation in the automotive industry. One of the critical challenges in the development of autonomous vehicles is the creation of reliable and robust perception systems. The ability of autonomous vehicles to accurately detect and respond to dynamic elements in their environment, such as lanes and speed breakers, is paramount to ensuring the safety and efficiency of these vehicles on the road. This research focuses on the implementation of a Lane Detection and Speed Breaker Warning System for autonomous vehicles, leveraging the power of machine learning algorithms. The primary objective is to enhance the perception capabilities of autonomous vehicles, enabling them to navigate through complex road scenarios with a high level of accuracy and safety.

II. Motivation of the Project

Autonomous vehicles become more prevalent, ensuring their ability to navigate diverse and challenging road conditions becomes crucial. Lane departure and the presence of speed breakers are common scenarios that demand swift and accurate responses from autonomous vehicles. By developing a robust detection and warning system, we aim to address these challenges and contribute to the overall reliability and safety of autonomous driving technology.

III. Objective

To develop a robust Lane Detection and Speed Breaker Warning System for Autonomous Vehicles through the implementation of advanced Machine Learning algorithms. This system aims to enhance the safety and efficiency of autonomous vehicles by accurately identifying and tracking lanes on roadways, as well as detecting and providing timely warnings for speed breakers. The project will leverage cutting-edge computer vision techniques and deep learning models to enable real-time decision-making, contributing to the seamless integration of autonomous vehicles into diverse road environments.

IV. Analysis Model

DLC Models stands for Software Development Life Cycle Models. In this article, we explore the most widely used SDLC methodologies such as Agile. Each software development life cycle model starts with the analysis, in which the Also, here are defined the technologies used in the project, team load. One of the basic notions of the software development process is SDLC models which stands for Software Development Life Cycle models. SDLC – is a continuous process, which starts from the moment, when it's made a decision to launch the project, and it ends at the moment of its full remove from the exploitation. There is no one single SDLC model. They are divided into main groups, each with its features and weaknesses.

1.Requirement Analysis - Requirement Analysis is the most important and necessary stage in SDLC. The senior members of the team perform it with inputs from all the stakeholders and domain experts or SMEs in the industry. Planning for the quality assurance requirements and identifications of the risks associated with the projects is also done at this stage. Business analyst and Project organizer set up a meeting with the client to gather all the data like what the customer wants to build, who will be the end user, what is the objective of the product. Before creating a product, a core understanding or knowledge of the product is very necessary.

2. System Design - The next phase is about to bring down all the knowledge of requirements, analysis, and design of the software project. This phase is the product of the last two, like inputs from the customer and requirement gathering.

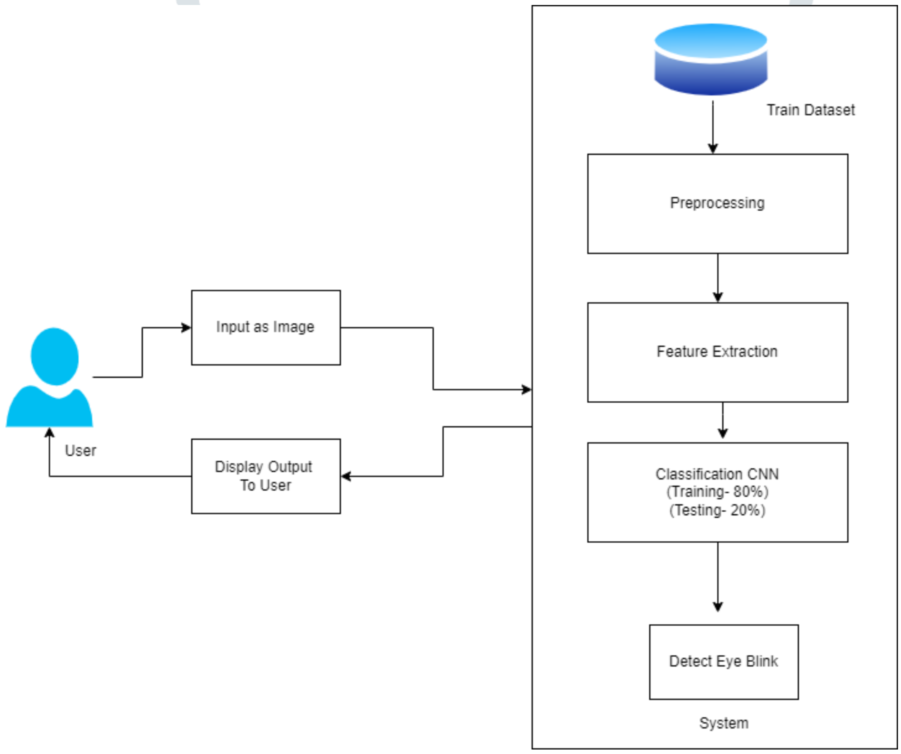
3. Implementation - In this phase of SDLC, the actual development begins, and the programming is built. The implementation of design begins concerning writing code. Developers have to follow the coding guidelines described by their management and programming tools like compilers, interpreters, debuggers, etc. are used to develop and implement the code.

4. Testing - After the code is generated, it is tested against the requirements to make sure that the products are solving the needs addressed and gathered during the requirements stage. During this stage, unit testing, integration testing, system testing, acceptance testing are done.

5. Deployment - Once the software is certified, and no bugs or errors are stated, then it is deployed. Then based on the assessment, the software may be released as it is or with suggested enhancement in the object segment. After the software is deployed, then its maintenance begins.

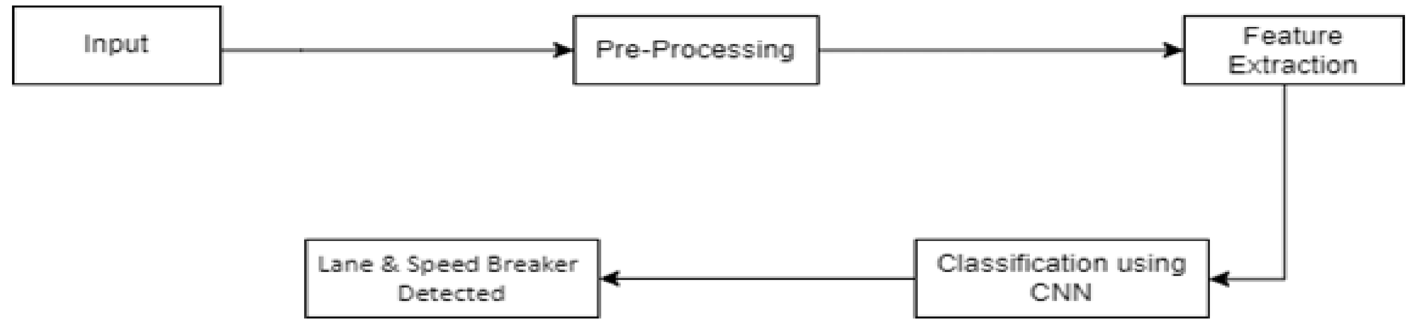
6. Maintenance - Once when the client starts using the developed systems, then the real issues come up and requirements to be solved from time to time. This procedure where the care is taken for the developed product is known as maintenance.

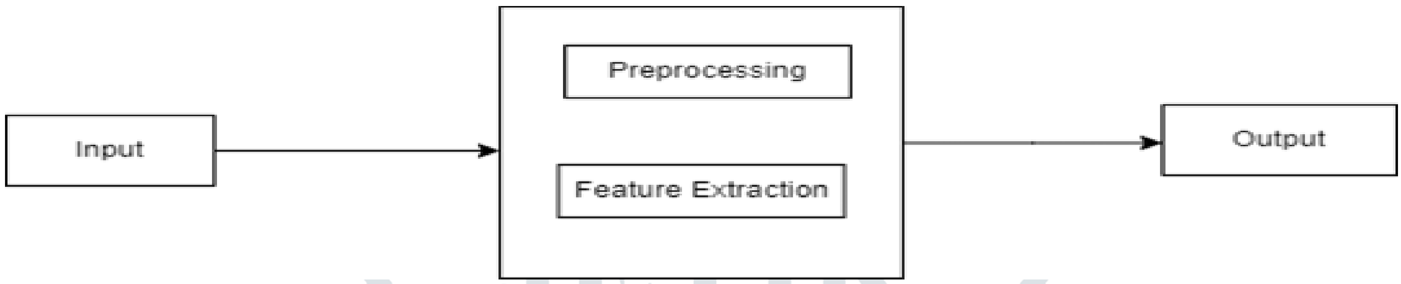
V. System Architecture



VI. Dataflow Diagrams

In Data Flow Diagram, we Show that flow of data in our system in DFD0 we show that base DFD in which rectangle present input as well as output and circle show our system, In DFD1 we show actual input and actual output of system input of our system is text or image and output is rumor detected like wise in DFD 2 we present operation of user as well as admin.





VII. System Testing

1. Introduction

Software testing, depending on the testing method employed, can be implemented at any time in the development process. However, most of the test effort occurs after the requirements have been defined and the coding process has been completed. As such, the methodology of the test is governed by the software development methodology adopted. Different software development models will focus the test effort at different points in the development process. Newer development models, such as Agile, often employ test driven development and place an increased portion of the testing in the hands of the developer, before it reaches a formal team of testers. In a more traditional model, most of the test execution occurs after the requirements have been defined and the coding process has been completed.

2. Types of Testing Used

- **Unit testing**

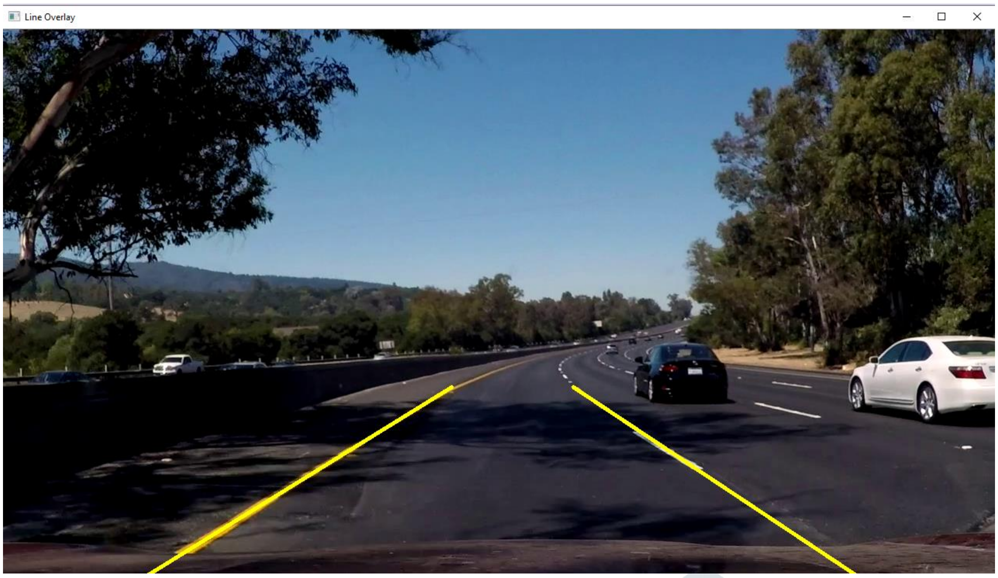
It is the testing of individual software units of the application. It is done after the complexion of an individual unit before integration. Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

- **Integration testing**

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

- **System testing**

To test this application we are going with proper sequencing of testing like unit, integration, validation, GUI, Low level and High level test cases, major scenarios likewise. We will go with the GUI testing first and then integration testing. After integration testing performs the high level test cases and major scenarios which can affect the working on the application. We will perform the testing on the data transmitted using the various inputs and outputs and validate the results.

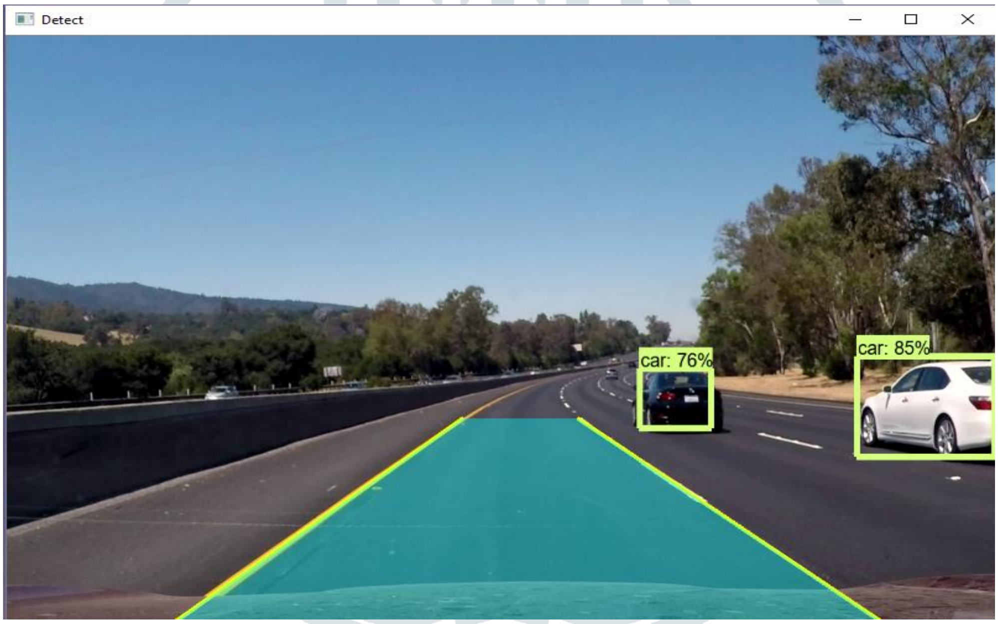


3. White Box Testing

Software testing methods are traditionally divided into white- and black-box testing. These two approaches are used to describe the point of view that a test engineer takes when designing test cases. In white-box testing an internal perspective of the system, as well as programming skills, are used to design test cases.

4. Black Box Testing

Black-box testing treats the software as a quot;black boxquot;, examining functionality without any knowledge of internal implementation. The testers are only aware of what the software is supposed to do, not how it does it.



VIII. OUTPUTS

1. Lane Detection

2. Speed Breaker Detection

IX. REFERENCES

[1] Andersen, J., Sutcliffe, S. (2000). Intelligent Transport Systems (ITS) - An Overview. IFAC Proceedings Volumes, 33(18), 99-106.

[2] Devapriya, W., Nelson Kennedy Babu, C. and Srihari, T. (2016) Real Time Speed Bump Detection Using Gaussian Filtering and Connected Component Approach. Circuits and Systems, 7, 2168-2175.

[3]. Accessed: Jun 15, 2020: kill-they-cause-30-crashes-and-9-deaths-aday/articleshow/59209813.cms.

[4] Varona, B., Monteserin, A. Teysseyre, A. A deep learning approach to automatic road surface monitoring and pothole detection. Pers Ubiquit Comput 24, 519–534 (2020).

[5] VSKPVarmaa,AdarshSa,KIRamachandranb,Binoy BNaira"RealTimeDetectionofSpeedHump/Bump and Distance Estimation with Deep Learning using GPU and ZED Stereo Camera" , 8th International Conference on Advances in Computing and Communication (ICACC-2018) , pp. 988 -997, 2018.

[6] Blue, S.T. and Brindha, M., 2019, July. Edge detection based boundary box construction algorithm for improving the precision of object detection in YOLOv3. In 2019 10th International Conference on Computing, Communication and Networking Technologies (ICCCNT) (pp. 1-5). IEEE.

[7] Zaidel D., Hakkert A.S., And Pistiner A.H., "The Use Of Road Breakers For Moderating Speeds On Urban Streets.," Accident Analysis And Prevention, Vol. 24

[8] Afrin, M., Mahmud, M. R., and Razzaque, M. A. (2015, December). "Real time detection of speed breakers and warning system for on-road drivers", In 2015 IEEE International WIE Conference on Electrical and Computer Engineering (WIECONECE) (pp. 495-498). IEEE