TRAFFIC SIGN DETECTION FROM VIDEO FOR AUTONOMOUS VEHICLES

By

WONG SONG WANG

A REPORT

SUBMITTED TO

Universiti Tunku Abdul Rahman

in partial fulfillment of the requirements

for the degree of

BACHELOR OF COMPUTER SCIENCE (HONOURS)

Faculty of Information and Communication Technology
(Kampar Campus)

FEB 2025

COPYRIGHT STATEMENT

This Final Year Project report is submitted in partial fulfillment of the requirements for the degree of Bachelor of Computer Science (Honours) at Universiti Tunku Abdul Rahman (UTAR). This Final Year Project report represents the work of the author, except where due acknowledgment has been made in the text. No part of this Final Year Project report may be reproduced, stored, or transmitted in any form or by any means, whether electronic, mechanical, photocopying, recording, or otherwise, without the prior written permission of the author or UTAR, in accordance with UTAR's Intellectual Property Policy.

ACKNOWLEDGEMENTS

I would like to express my sincere thanks and appreciation to my supervisors, Prof. Dr Leung Kar Hang who has given me this bright opportunity to engage in an image processing project. It is my first step to establish a career in image processing field. A million thanks to you.

Finally, I must say thanks to my parents and my family for their love, support, and continuous encouragement throughout the course.

ABSTRACT

Traffic sign detection from video plays a vital role in enhancing the safety and decision-making capabilities of autonomous vehicles and Advanced Driver Assistance Systems (ADAS). This project focuses on the development of a robust deep learning-based detection system utilizing the latest YOLO11 model to identify and classify traffic signs from recorded video feeds. The system was trained using a carefully prepared dataset consisting of 21,688 images across 18 traffic sign classes, collected under various real-world conditions such as illumination changes and occlusions.

The YOLO11 model was fine-tuned through data augmentation and hyperparameter optimization to maximize detection accuracy and model generalization. The final model demonstrated strong performance, achieving a precision of 96.8%, recall of 97.3%, mAP@50 of 98.7%, and mAP@50–95 of 90.8%.

The project concludes with the successful implementation of an efficient and scalable traffic sign detection framework that supports high reliability. The findings contribute to the field of computer vision and intelligent transportation by demonstrating the effectiveness of the YOLO11 model in detecting traffic signs under challenging conditions. This work serves as a foundation for further enhancements in autonomous navigation and real-world deployment of intelligent perception systems.

Area of Study: Computer Vision

Keywords: Deep Learning, YOLO11, Traffic Sign Detection, ADAS, Object Detection

Table of Contents

List of Figures

LIST OF TABLES

List of Abbreviations

ADAS	Advanced Driver Assistance Systems
YOLO11	You Only Look Once 11
SVMs	Support Vector Machines
k-NN	k-Nearest Neighbour
TSR	Traffic Sign Recognition
TSDR	Traffic Sign Detection and Recognition
HOG	Histogram of Oriented Gradients
LSS	Local Self-Similarity
PHOG	Pyramid Histogram of Oriented Gradients
CNN	Convolutional Neural Network
GTSRB	German Traffic Sign Recognition Benchmark
DPM	Deformable Part Models
FPS	Frames per Second
SSD	Single Shot MultiBox Detector
mAP	Mean Average Precision
OpenCV	Open Source Computer Vision Library
IoU	Intersection Over Union
RIOs	Regions of Interest
DFL	Distribution Focal Loss

CHAPTER 1 – INTRODUCTION

1.1 Background Information

Traffic signs are essential visual indicators that communicate regulatory, warning, and guidance information to road users, forming a critical part of traffic control systems worldwide. They are designed with standardized shapes, colours, and symbols to ensure immediate recognition and interpretation by drivers under varying conditions. In the context of autonomous vehicles and Advanced Driver Assistance Systems (ADAS), the reliable detection and classification of traffic signs is a prerequisite for safe and intelligent navigation. The ability to perceive and interpret these signs correctly enables the vehicle to make context-aware decisions, such as adjusting speed, yielding at intersections, or avoiding prohibited turns.

Traditional approaches to traffic sign detection have relied heavily on colour-based segmentation and shape-based filtering techniques. While these methods are computationally lightweight and intuitive, they are often sensitive to environmental variables such as changes in illumination, weather conditions, and partial occlusions. These limitations restrict their applicability in real-world scenarios, especially in dynamic and unstructured environments.

Recent advancements in deep learning have transformed object detection methodologies, offering powerful alternatives through convolutional neural networks (CNNs) capable of learning spatial hierarchies of features directly from raw input data. Among these, the "You Only Look Once" (YOLO) family of models has emerged as a leading solution for real-time object detection, combining speed and accuracy in a single-stage architecture. YOLO11, the latest iteration in this series, introduces architectural optimizations that enhance detection performance while reducing computational complexity. It processes images holistically, allowing it to retain global context and achieve high mean average precision (mAP) even in dense or complex scenes.

In this project, the YOLO11 model is leveraged to build a video-based traffic sign detection system. The model is trained on a large, labelled dataset of traffic sign images encompassing 18 classes, and then applied to analyse individual frames from video feeds. This approach ensures not only accurate sign recognition under challenging

conditions but also paves the way for future integration into intelligent transportation systems, supporting the broader vision of fully autonomous driving.

1.2 Problem Statement and Motivation

With the rapid increase in the number of vehicles on the road, the risk of accidents due to human error has also grown. Manually driving cars poses several challenges, as drivers may overlook critical traffic signs due to factors like fatigue, distractions, or environmental conditions. These oversights can lead to dangerous situations, including speeding or missing stop signs, resulting in accidents that can cause serious injuries or fatalities.

In response to these challenges, the development of Advanced Driver Assistance Systems (ADAS), which includes features like traffic sign detection, has become essential. However, many existing traffic sign detection systems often fail under varying environmental conditions such as low visibility, poor lighting, or worn-out signs. These failures can lead to incorrect decisions by both human drivers and autonomous systems, further increasing the risk of accidents. Therefore, there is an urgent need for a reliable traffic sign detection system that can operate effectively across a wide range of conditions to ensure the safety of all road users and support the development of autonomous driving technologies.

This project aims to address these challenges by developing a robust traffic sign detection system that improves detection accuracy under adverse conditions, thereby reducing the rate of traffic-related accidents and enhancing the functionality of ADAS.

1.3 Project Objectives

The primary objective of this project is to develop a robust and efficient traffic sign detection system using the YOLO11 deep learning model, optimized for processing video feeds in the context of autonomous vehicle environments. The system is designed to accurately identify and localize a wide range of traffic signs—such as regulatory, warning, and advisory types—within video frames, ensuring high detection accuracy and resilience under diverse real-world conditions.

To achieve this overarching goal, several specific objectives have been defined:

- To curate and preprocess a high-quality image dataset comprising 18 traffic sign classes under varied environmental conditions, including different lighting, occlusions, and backgrounds.
- To train and fine-tune the YOLO11 model using the prepared dataset, optimizing for detection performance through data augmentation, hyperparameter tuning, and iterative model validation.
- To evaluate the trained model using industry-standard performance metrics, including precision, recall, mean average precision (mAP), F1-score, and Intersection over Union (IoU), ensuring the system's reliability and robustness across different sign categories.
- To apply the trained model to frame-by-frame detection tasks on recorded video feeds, analysing system behaviour in realistic driving scenarios and assessing generalization capability in dynamic visual environments.

Through these objectives, the project aims to deliver a scalable, image-trained traffic sign detection system capable of supporting future research and practical deployment within intelligent transportation frameworks and ADAS modules.

1.4 Project Scope

This project is dedicated to the development of a robust, deep learning-based image processing system for detecting and recognizing traffic signs from video feeds, with a specific emphasis on applications in autonomous vehicles and intelligent driver assistance. The system is exclusively built upon the YOLO11 object detection framework, selected for its state-of-the-art performance, architectural efficiency, and proven scalability in complex environments. The central objective is to design and implement a complete detection pipeline that utilizes a YOLO11 model trained on annotated traffic sign images, which is then deployed to perform inference on individual frames extracted from recorded videos.

The project encompasses the full system development lifecycle, including dataset acquisition and preparation, model training, hyperparameter tuning, validation, performance benchmarking, and final application to traffic sign detection in video streams. A real-world dataset containing 18 distinct traffic sign classes was curated for the training phase, featuring diverse environmental conditions such as variable lighting,

partial occlusions, and multiple sign orientations. The model was trained solely on static images, while its detection capabilities were validated on pre-recorded video clips through frame-by-frame analysis.

The project scope excludes real-time implementation, deployment on embedded vehicular platforms, and the integration of system outputs with vehicular control modules. Furthermore, it does not aim for comprehensive global sign generalization but instead concentrates on widely encountered regulatory and warning signs within the dataset's context. Assumptions include the availability of video and image data of sufficient resolution and quality, as well as the presence of traffic signs adhering to standard design conventions. These boundaries are designed to ensure technical focus while delivering a scalable and reliable foundation for future integration into Advanced Driver Assistance Systems (ADAS) and autonomous navigation technologies.

1.5 Contribution

The development of a reliable traffic sign detection system holds considerable significance in the evolution of autonomous vehicles and Advanced Driver Assistance Systems (ADAS), where safety, responsiveness, and environmental awareness are paramount. In this context, the project contributes to the field of intelligent transportation by presenting a complete deep learning-based solution capable of detecting traffic signs from video feeds with high accuracy and efficiency.

The use of the YOLO11 model as the core of the system introduces substantial improvements over conventional detection methods, owing to its high detection precision, and adaptability across diverse environmental conditions. By training the model exclusively on a curated image dataset and applying it to analyse video frames, the project demonstrates a scalable and modular pipeline that bridges the gap between offline learning and real-world visual inference.

The project's significance also lies in its practical orientation—emphasizing real-world deployment constraints, such as variable lighting, occlusions, and dynamic backgrounds—while maintaining a focus on computational performance. Through rigorous evaluation using standard detection metrics and qualitative analysis on recorded video scenarios, the system sets a performance benchmark for future enhancements in traffic sign detection.

Key contributions include the creation of a well-labelled traffic sign dataset comprising 18 distinct classes, the successful fine-tuning and application of the YOLO11 model for video-based detection, and the delivery of a fully tested pipeline that can be extended or integrated into future ADAS frameworks. Ultimately, this work supports broader goals in autonomous driving by providing a reliable visual perception module capable of improving vehicle intelligence, operational safety, and traffic law compliance.

1.6 Report Organization

This report is structured into seven chapters. Chapter 1 introduces the research background, objectives, and scope. Chapter 2 reviews existing traffic sign detection methods and justifies the adopted approach. Chapter 3 outlines the system methodology and model architecture. Chapter 4 presents the detailed system design, including dataset preparation, model training, and video inference. Chapter 5 discusses implementation setup, operational workflow, and encountered challenges. Chapter 6 evaluates system performance using quantitative and qualitative metrics. Chapter 7 concludes the report with a summary of findings and recommendations for future enhancements.

CHAPTER 2 – LITERATURE REVIEW

2.1 Traffic Sign Detection and Recognition Based on Random Forests

2.1.1 Introduction to TSDR and ADAS

Traffic Sign Detection and Recognition (TSDR) is a key module in Advanced Driver Assistance Systems. This greatly works toward ensuring road safety and allows the vehicle to detect and recognize traffic signs on the go. It helps the driver to follow the rules of the road and navigate through the roads efficiently. The paper named "Traffic Sign Detection and Recognition Based on Random Forests" by Ellahyani et al. [8] suggests a new approach to TSDR based on colour segmentation, shape classification using invariant geometric moments, and a recognition process in which Histogram of Oriented Gradients (HOG) combined with Local Self-Similarity (LSS) features are used. With Random Forests as the main classifier, a strong and efficient framework can be developed for traffic sign recognition.

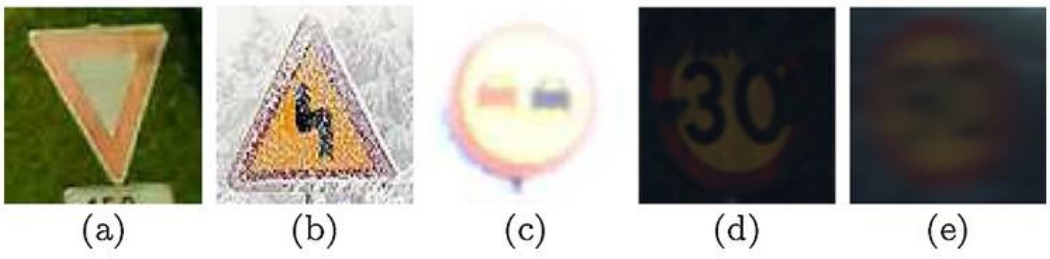


Figure 2.1.1.1 Examples of Difficulties Facing Traffic Sign Recognition

2.1.2 Colour Segmentation and Shape Classification Techniques

Several methodologies have been proposed within the literature to tackle the challenges presented by TSDR. In most approaches, colour segmentation has been a common place to begin working from to isolate possible traffic signs from their environment. Many early studies used the RGB colour space, but it is very sensitive to lighting changes and therefore not so reliable when operating outdoors. To overcome this, more recent approaches have shifted to using alternative colour spaces, such as YUV and HSI, that are less affected by the changes in illumination. For example, Saadna and Behloul [9] used specific relations between the RGB components for segmenting the traffic signs, while Zaklouta and Stanciulescu [10] did an enhancement of colour channels in the RGB space to detect signs effectively. The proposed method by Ellahyani et al. [8] goes a step ahead and does segmentation in the HSI colour space, which is more robust to

lighting variations, thus making the process of detection robust.

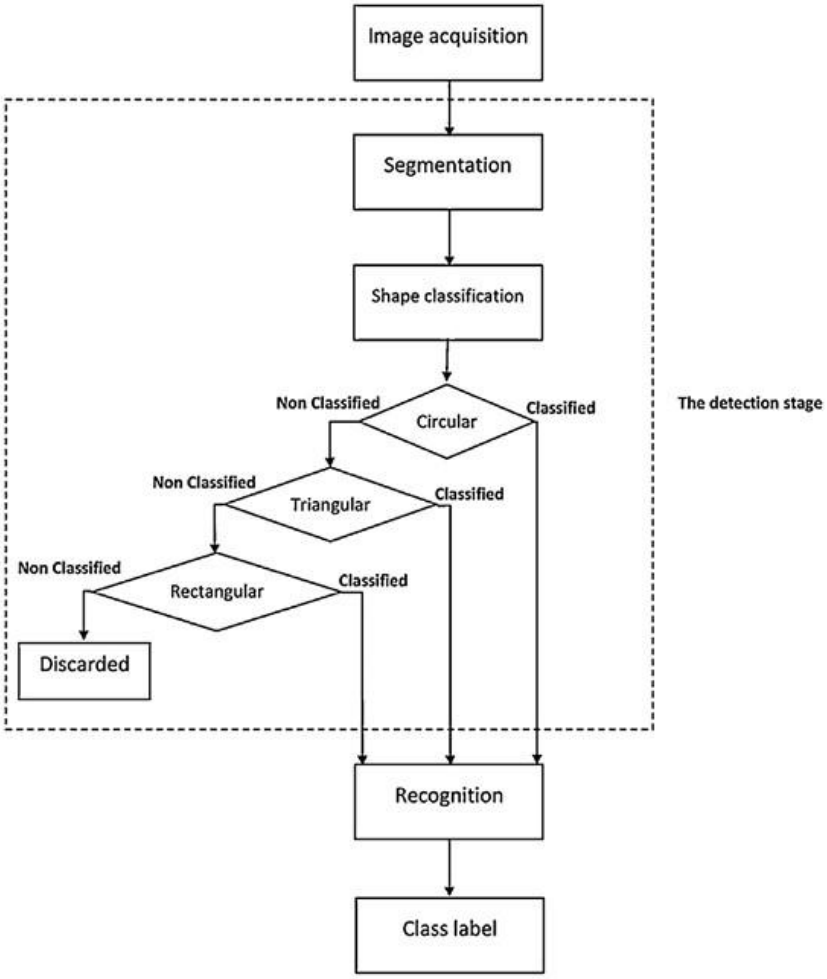


Figure 2.1.2.1 Algorithm Scheme

Once the colour segmentation has detected ROIs, then only shape classification will take place. The most general shape characteristics are circular, triangular, and rectangular, using which the traffic signs can be distinguished and classified. Traditional methods for shape classification include corner detection and the Hough transform, which are directly applied to scene images or post-segmentation. Ellahyani et al. [8] introduce invariant geometric moments as a shape classification tool that significantly reduces computational complexity and increases accuracy. This reduces the computational complexity dramatically and is considered more accurate than conventional machine learning classifiers such as Support Vector Machines (SVMs), which require large and long learning processes.

Quite a few various approaches have also been developed for the recognition stage,

finding the content of detected signs. The HOG features had been adopted by researchers due to their strength in describing local gradients and being less sensitive to variations in lighting and scale. To do that, the paper under review suggests extending HOG features to the HSI colour space, therefore producing the HSI-HOG descriptor. Then, this descriptor was further combined with LSS features; it will give more holistic methods in bringing colour and texture information for higher accuracy in recognition. In this setup, Random Forests as the classifier are especially well-suited because it is robust to overfitting and good at working in noisy data.

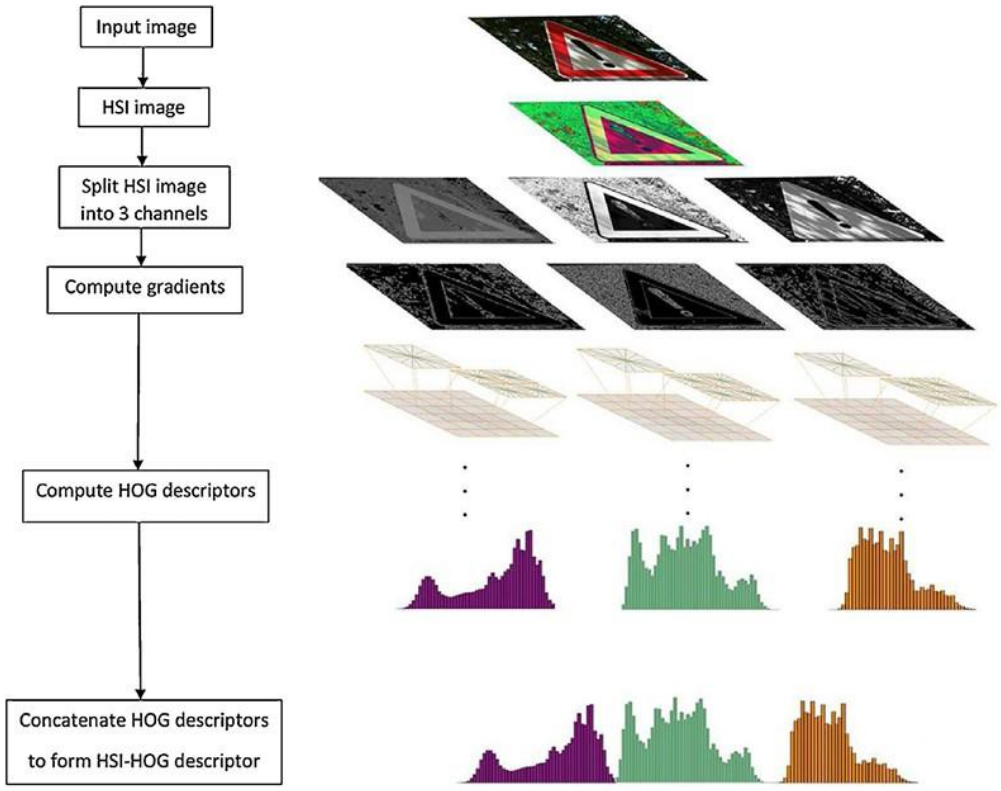


Figure 2.1.2.2 Example of HSI-HOG Feature Computation

2.1.3 Strengths and Limitations of the Proposed Method

Despite several strengths of the proposed method, there still lie a few limitations. The main challenges are the system's sensitivity to environmental conditions, which entails heavy shadows or a faded sign colour that leads to a failure in detection and one of the sources of false positives. Some of these problems could be reduced by the HSI colour space, but extreme conditions still pose to be too challenging. Besides, the combined feature extraction of HSI-HOG with LSS proves to be a boost in effective results, which in turn increases the computational complexity of the process of recognition to an extent

that may hamper real-time performances on platforms under constraints in resources. It also heavily relies on empirically derived thresholds for colour segmentation and shape classification, which are not really generalized among different datasets and varied environmental conditions.

To address these limitations, several improvements could be implemented. Increasing further the robustness of colour segmentation and shape classification could be done with the use of adaptive thresholding techniques, when dealing with varying lighting conditions. Another way to enhance the performance is that signs detected over the frames can further be tracked using temporal data from video sequences to reduce false positives and increase detection rates for more challenging scenarios. On the other hand, feature selection techniques can be implemented in selecting an optimal size of descriptors with respect to which the computational load will be reduced while keeping recognition accuracy high. This optimization can be combined with other classifiers like Convolutional Neural Networks, which show extremely higher efficiency in a TSDR task besides being computationally expensive.

2.1.4 Conclusion

In conclusion, the literature on traffic sign detection and recognition is quite diverse and consists of many ways of tackling this problem. This offers a good solution following the work of Ellahyani et al. [8], who used the HSI colour space for segmentation, invariant geometric moments for shape classification, and a good combination of HSI-HOG and LSS features for recognition. However, there are still several avenues open for improvements, regarding how to render the system more robust in face of environmental variations and computationally efficient for real-time applications. Such an advancement would ensure that TSDR systems would perform in a reliable manner under a broad spectrum of conditions, hence ensuring overall safety and efficacy in the process of driving.

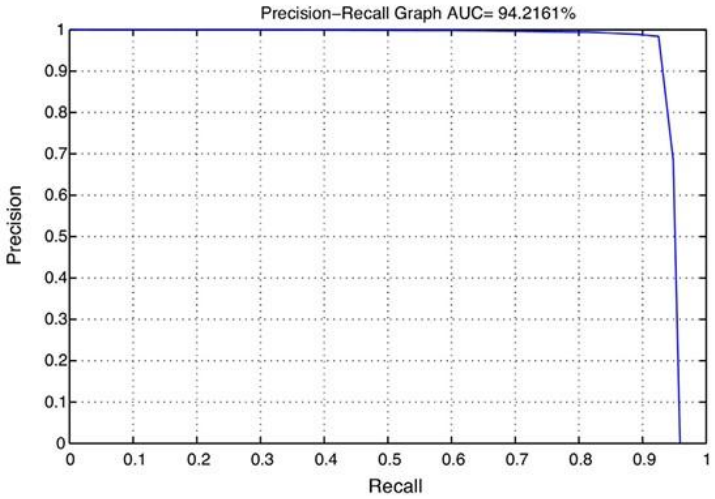


Figure 2.1.4.1 Precision-Recall Curves of the Proposed Detection and Recognition Method

2.2 A novel traffic sign detection method via colour segmentation and robust shape matching

2.2.1 Existing Techniques

During the past few decades, researchers have proposed many methods to improve the robustness and accuracy of traffic sign detection under different real-world conditions, such as variations due to weather, shadows, and occlusions. Traditional techniques for detecting traffic signs rely mostly on colour and shape features because traffic signs typically have distinct and regular visual characteristics, such as high-contrast colours like red, yellow, and blue, and regular shapes like circles, triangles, and diamonds. For example, Ganesan et al.[11] modelled colour pixels with a Gaussian model in CIE Lab space to mitigate the effects of varying illumination. Shape-based methods like the Hough transform have been applied to detect specific shapes, such as circular speed limit signs. However, these methods often struggle with computational complexity and are sensitive to environmental factors like lighting and background clutter.

Researchers have increasingly combined colour-based and shape-based methods to improve detection accuracy. For instance, they used a corner detection algorithm after segmenting red colour regions to identify triangular borders. Similarly, they employed Gabor features and K-means clustering in CIE Lab space to distinguish traffic sign shapes. While these combined features enhance the robustness of traffic sign detection systems, challenges remain, particularly in cluttered environments and under adverse

weather conditions.

2.2.2 Strengths and Limitations

The main strength of these combined approaches is their ability to leverage the complementary nature of colour and shape information, which generally leads to better detection rates compared to methods that rely solely on one type of feature. However, these approaches are not without limitations. Colour-based methods are particularly susceptible to variations in illumination and weather conditions, leading to inaccurate segmentation of signs. Similarly, shape-based methods, while more robust to colour variations, often suffer from high computational costs and sensitivity to occlusions and background noise.

2.2.3 Improvements Proposed

This paper proposes an innovative approach that integrates colour invariants-based image segmentation with Pyramid Histogram of Oriented Gradients (PHOG) features for shape matching. The method enhances the robustness of traffic sign detection against varying environmental conditions by introducing chromatic-edge enhancement to improve contour detection, thereby reducing the noise sensitivity associated with traditional PHOG features. The combination of colour invariants and PHOG, along with a support vector machine (SVM) for classification, offers a more discriminative and computationally efficient solution for traffic sign detection.

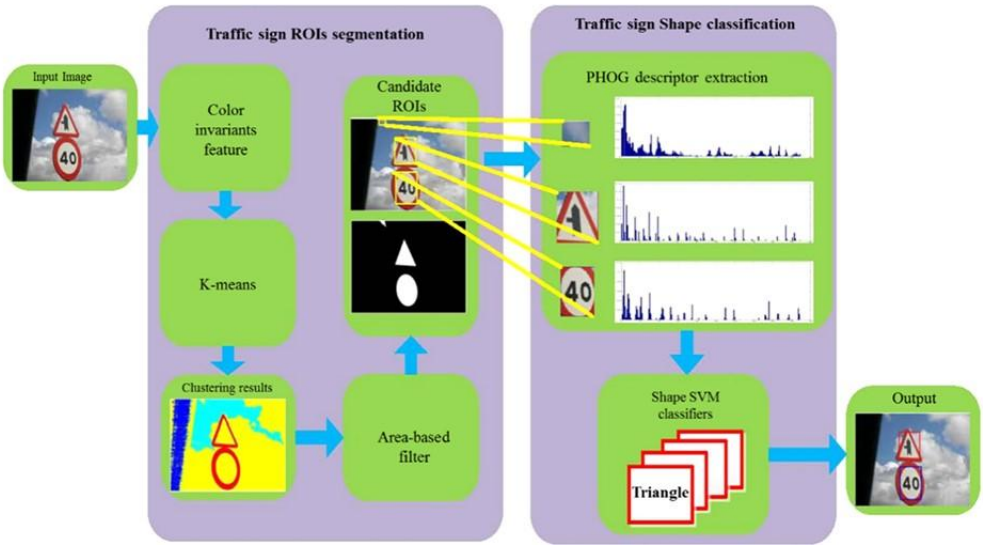


Figure 2.2.3.1 The outline of the proposed traffic sign detection system

The proposed method shows significant improvements in detection accuracy,

particularly under challenging conditions such as shadows, occlusions, and complex backgrounds. Chromatic-edge enhancement effectively addresses the weaknesses of traditional PHOG features, making the system more robust to noise and clutter. However, the method's reliance on predefined colour models and the need for extensive training data for the SVM classifiers could be seen as limitations. The approach might also face challenges in scenarios with severely degraded or non-standard traffic signs, where colour and shape information alone might not be sufficient.

2.2.4 Conclusion

In conclusion, the integration of colour invariants and enhanced PHOG features represents a promising direction for robust traffic sign detection. Future work could focus on overcoming the limitations related to the reliance on predefined models and exploring more adaptive techniques that can handle a wider range of real-world variations in traffic sign appearance.

2.3 Incremental Framework for Video-Based Traffic Sign Detection, Tracking, and Recognition

The paper by Yuan et al. [12]presents a new approach to traffic sign recognition with a unified framework that incorporates detection, tracking, and recognition using video data from a camera mounted on the vehicle. This approach addresses several limitations of existing TSR systems.

2.3.1 Existing Practices and Strengths

Many of the conventional ways in the field of TSR are predominantly based on colour segmentation and shape-based detection, which work suitably well under controlled situations but fail with the variability observed in actual scenarios. Deep learning methods, especially Convolutional Neural Network (CNNs), have made a significant leap in accuracy by learning complex features from large data sets, getting results on benchmarks like German Traffic Sign Recognition Benchmark (GTSRB) where performance is close to human. Techniques such as tracking maintain the consistency of detection across frames, thereby leading to a reduction in false positive rates and improved localization accuracy.

2.3.2 Limitations of Current Solutions

More significantly, deep learning-based models are very computationally demanding

and need high-end hardware; hence, the models are less applicable to real-time embedded systems. Moreover, many TSR systems mainly focus on single-image processing and neglect the temporal information during video processing, which causes inconsistency in dynamic environments. Simplistic motion models applied for tracking may not work with sudden changes in vehicle motion.

Figure 2.3.2.1 Shows the various scenarios in which traffic signs may appear

Figure 2.3.2.2 Shows the appearance changes of traffic signs caused by occlusion and illumination

2.3.3 Contributions of the Paper

The proposed framework can address the limitations raised with the combination of detection, tracking, and recognition in a single system. It is based on an incremental learning approach that operates in real time and can adapt to the changes in the environment online without any requirement for time-consuming retraining. For

accuracy improvement of detections, spatial distribution priors based on typical traffic sign locations are employed. For the increase of the localization accuracy and improvement of robustness of the system in a non-stationary environment, Kalman filtering together with online sample collection is used.

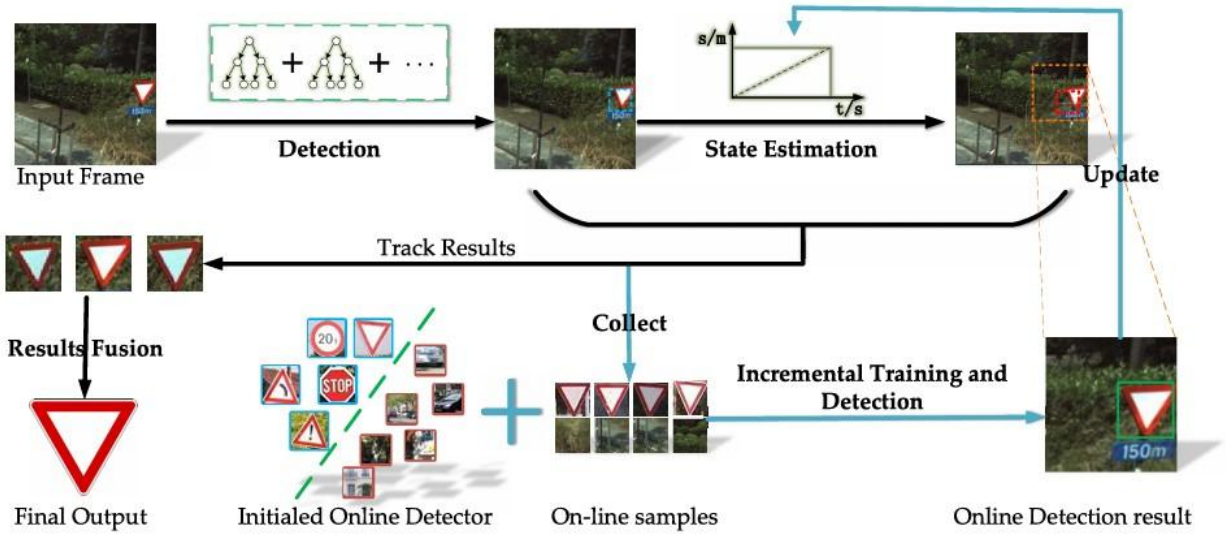


Figure 2.3.3.1 Overview of the components of TSR framework

Although effective, it is consequently true that to catch indications from unusual locations, like those presented during complex manoeuvres, flexibility is limited in relying on spatial priors. The framework is also computationally intensive, albeit much more efficient than deep learning models, thus it still needs optimization to work in real-time with dense traffic scenarios.

2.3.4 Conclusion

This paper reports a significant advance in TSR as it discusses an embedded approach for detection, tracking, and recognition in a single adaptive framework. Incremental learning of the spatial priors is the developed innovation in the increase of real-time performance. However, for much of this work, there is a need for increased spatial flexibility and computational efficiency to further the applicability of frameworks such as these.

2.4 Vision-Based Traffic Sign Detection and Recognition Systems

This paper [13] presents an in-depth survey of existing methodologies for vision-based traffic sign detection and recognition systems, summarizing the strengths and weaknesses of each.

2.4.1 Existing Practices and Solutions

Colour-based methods are based on the inherent colours of traffic signs, like red, blue, and white, to locate regions of interest in images. The key features of signboards are their geometry: shape-based methods focus on geometric characteristics of the signs through such techniques as Hough transformation and edge detection to identify a given shape. These methods are robust to illumination variation but are computationally intensive and do not work well under conditions where the signs are partially occluded or deformed.

Hybrid methods combine colour and shape-based features that generally provide more accuracy and reliability in the detection part. A hybrid approach is considered more accurate due to the usage of colour in restricting the search space for the detection of shapes in complicated environments.

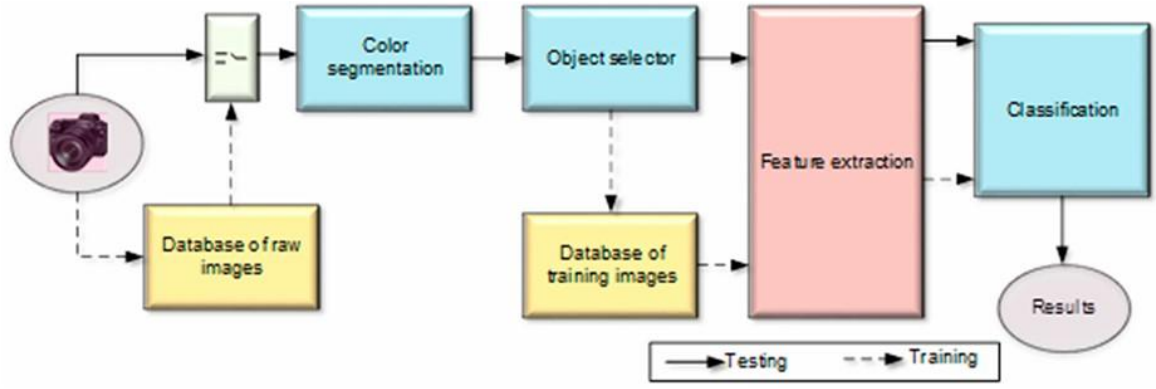


Figure 2.4.1 Block diagram of the traffic sign recognition system

2.4.2 Strengths and Weaknesses

Colour-based methods are efficient and fast and, therefore, suitable for real-time detection but not reliable in varying lighting or when signs are damaged. Shape-based methods are usually robust in challenging lighting; however, they are computationally intensive and usually do not work very well if the signs are not perfect. Hybrid methods may offer some improvement in accuracy but are more difficult to tune and can still have significant difficulty when working in dynamic real-world conditions.

2.4.3 Addressing Limitations

It will be necessary for research in the future to focus on advanced machine learning techniques to investigate deep learning to enhance the adaptability and accuracy of the

TSDR system. In principle, a deep-learning technique should handle a larger variety of appearances, even in the case when a sign is occluded or damaged. Introducing contextual information, like the expected location of a sign, will probably reduce false positives and increase reliability in detection.

In a nutshell, there is a requirement for more research to come up with a way of overcoming some of the challenges in existence and enhancing the accuracy and reliability of TSDR systems for safer driver assistance.

2.5 Machine Vision Based Traffic Sign Detection Methods

This paper [1] presents a fair review of the methodologies used in Traffic Sign Detection (TSD) and Traffic Sign Recognition (TSR). The current review classifies existing approaches into colour-based approaches, shape-based approaches, machine learning-based approaches, and LIDAR-based approaches found to be effective in handling challenges associated with TSD. These mainly include alterations in lighting conditions, signs very small in size, and complicated driving environments.

2.5.1 Traffic Sign Detection Methods

Colour-based methods use only the different colour features of traffic signs, such as red, blue, and yellow, to segment the regions containing the signs from their backgrounds. These methods are widely popular owing to their simplicity and speed, making them suitable for real-time applications. They are highly sensitive to lighting variations and, in fact, require precise threshold adjustments that may be hurdles to generalization over a wide number of settings. Another approach is the use of shape-based methods to detect geometric shapes common in traffic signs, with techniques ranging from Hough transforms to Fourier descriptors. These methods are relatively robust toward colour variations but quite weak when it comes to small or partially occluded signs; moreover, the strong dependence on edge detection makes them computationally expensive and sensitive to noise.

Machine learning, and particularly deep learning, has revolutionized the state-of-the-art development of TSD. Techniques, such as AdaBoost, Support Vector Machines (SVM), and Convolutional Neural Networks (CNN), have been implemented to improve the accuracy of the sign detection and classification system. Among these techniques, CNNs exhibit better performance in learning complex features directly from the data; however, they rely on large datasets for training and are computationally

expensive, which may compromise real-time applications. In this sense, LIDAR-based methods provide a strong solution for robustness against occlusions and variable lighting conditions, based on the 3D structure of point cloud data and reflective properties. However, such methods often call for accurate recognition, leading to integration with camera data, and are currently limited by the scarcity of publicly available LIDAR datasets.

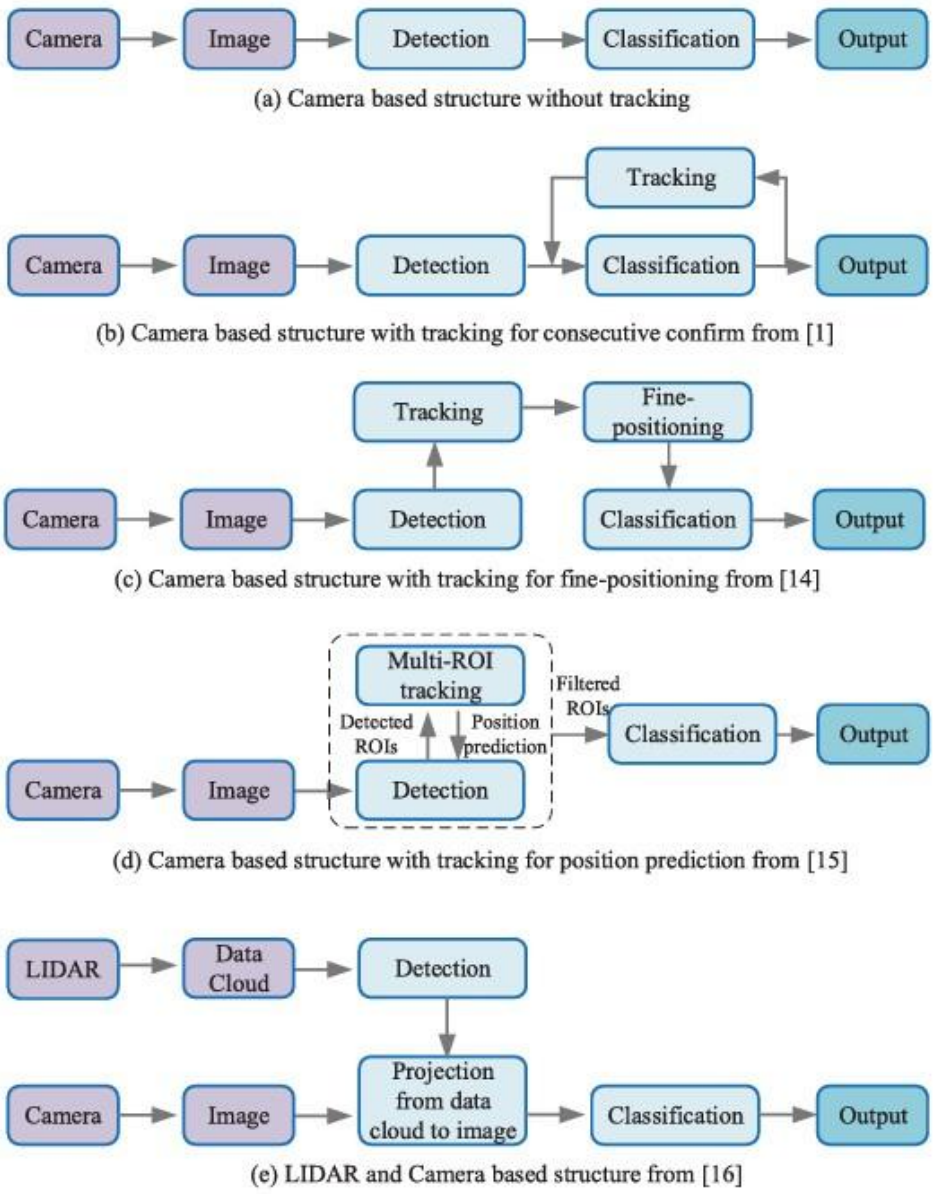


Figure 2.5.1.1 Different structures of traffic sign recognition systems

2.5.2 Strengths and Weaknesses

Each of these methods has its strengths and weaknesses. Most of the colour-based

methods are quick and simple but poorly performing under a different variety of conditions with respect to lightness. While this shape-based method works well in detection of standard shapes, it becomes computationally intense and not so efficient with small or unclear signs. Machine learning-based methods are accurate but costly in resources, depending on large, annotated datasets. LIDAR-based methods offer excellent point detection and are very good at noticing 3D space, although data integration and standardization are challenging. Adaptive colour thresholding techniques, hybrid methods for shape detection using some form of machine learning, optimization of ML models for real-time use, and broadening the LIDAR dataset to include more generalization and better benchmarking could be the focus of future research to take care of these limitations.

2.5.3 Conclusion

The above summary reveals that, despite immense progress in the field of TSD, there are still many challenges pertaining to performance optimization across diverse environmental conditions and ensuring computational efficiency for real-time applications. The future most likely will be the result of hybrid methods using the best aspects from different approaches and the development of better comprehensive data sets in support of the continued evolution of these technologies.

2.6 You Only Look Once: Unified, Real-Time Object Detection

YOLO (You Only Look Once) represents a groundbreaking approach to object detection by reframing it as a regression problem that predicts bounding boxes and class probabilities simultaneously. Unlike traditional systems such as Deformable Part Models (DPM) and region-based frameworks like R-CNN, which rely on multi-stage pipelines involving sliding windows or region proposals, YOLO unifies the entire detection process into a single convolutional neural network. This innovation significantly improves efficiency, enabling YOLO to achieve real-time performance of up to 155 frames per second (FPS) while maintaining competitive accuracy[6]. Furthermore, YOLO's ability to process entire images during training and testing allows it to capture global contextual information, making it less prone to false positives on background regions compared to methods like Fast R-CNN[6].

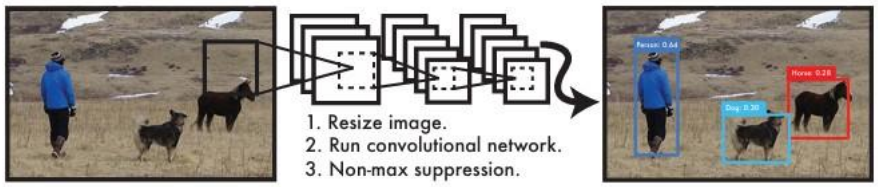


Figure 2.6.1 The YOLO Detection System

2.6.1 Strengths and Weaknesses

One of YOLO's key strengths lies in its speed, which makes it ideal for applications like autonomous driving where low latency is crucial. Its unified architecture streamlines object detection, reducing the complexity of training and optimization seen in multi-stage systems[6]. YOLO also excels in generalizing across domains, performing well even on datasets that differ significantly from its training data, such as artwork or abstract imagery[6]. However, despite these advantages, YOLO has notable limitations. It struggles with precise localization, particularly for small objects, due to its reliance on coarse feature maps and strong spatial constraints. Each grid cell in YOLO's framework predicts only two bounding boxes and one class, which limits its ability to detect multiple objects in close proximity[6]. Additionally, its loss function, which uses sum-squared error, does not adequately prioritize small bounding box errors, reducing its effectiveness in detecting smaller objects[6].

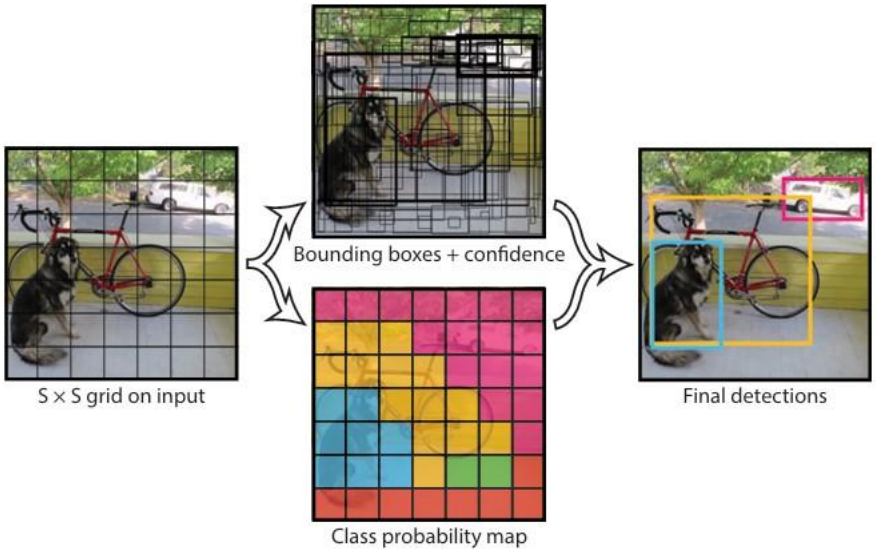


Figure 2.6.1.1 The Model

2.6.2Addressing Limitations

To address these limitations, enhancements in feature extraction and model design are necessary. Incorporating multi-scale feature maps, as seen in SSD (Single Shot MultiBox Detector), could improve YOLO's ability to detect small and overlapping objects. Adjusting the loss function to weigh localization errors more effectively based on object size would further enhance its performance. Additionally, redesigning the grid cell structure to allow more flexible predictions could help in scenarios with dense object arrangements. Hybrid models that combine YOLO's speed and contextual reasoning with the precise localization capabilities of methods like Fast R-CNN have already shown promise, with experiments demonstrating significant performance boosts in mean average precision (mAP)[6].

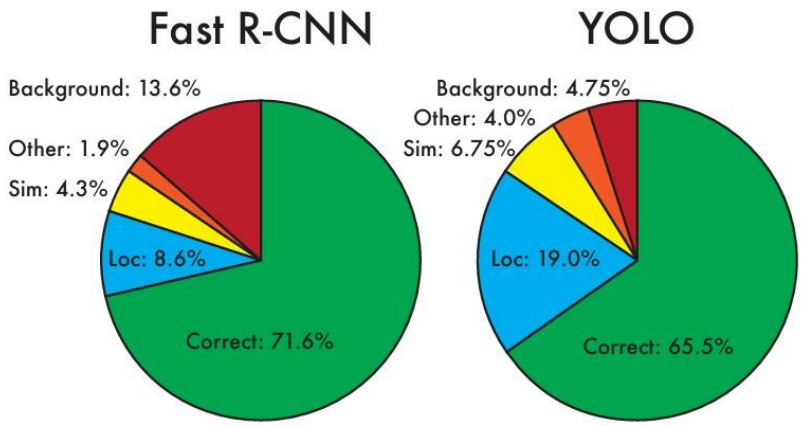


Figure 2.6.2.1 ErrorAnalysis: FastR-CNNvs. YOLO

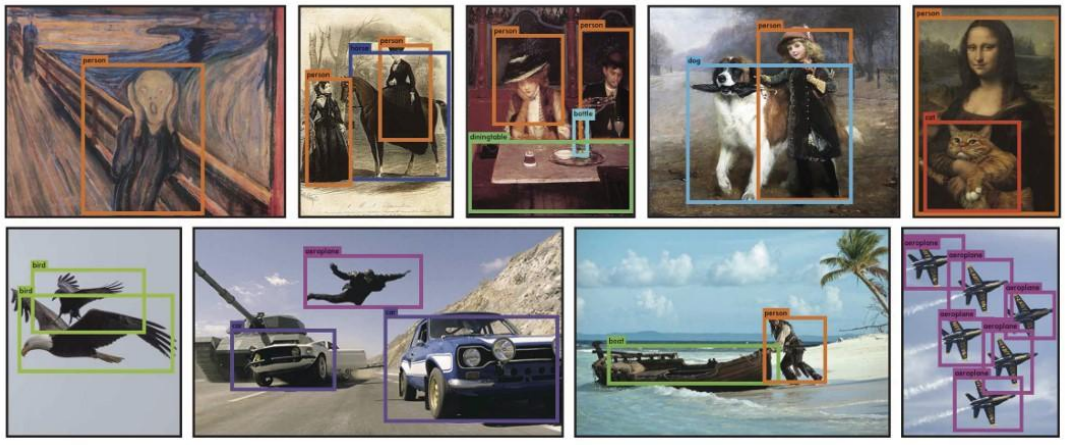


Figure 2.6.3.1 Qualitative Results

2.6.3 Conclusion

In conclusion, YOLO is a significant milestone in object detection, offering unmatched real-time performance and simplicity. However, addressing its challenges related to

small object detection, localization precision, and overlapping objects will make it even more robust, especially for safety-critical applications like traffic detection in intelligent transportation systems. These improvements could further solidify YOLO's position as a cornerstone in real-time object detection research and deployment[6].

2.7 Critical Remarks of Previous Works

In this section, the strengths and weaknesses of previous works reviewed in this chapter against our proposed approach for traffic sign detection and segmentation were examined critically.

Random Forest-Based TSDR (Section 2.1)

Strengths:

- Utilizes Random Forests with invariant geometric moments for shape classification.
- Effective against lighting variations.

Weaknesses:

- Sensitive to extreme environmental conditions like heavy shadows or faded signs.
- High computational complexity hinders real-time performance.

Comparison:

- The proposed solution integrates YOLO11 with traditional methods for improved robustness and efficiency.

Colour Segmentation + PHOG (Section2.2)

Strengths:

- Enhances detection accuracy under challenging conditions by combining colour invariants with PHOG.

Weaknesses:

- Relies on predefined colour models and requires extensive SVM training.
- Limited effectiveness for non-standard traffic signs.

Comparison:

- YOLO11's advanced generalization capabilities address limitations in static models.

Incremental Framework for Video-Based TSR (Section 2.3)

Strengths:

- Integrates detection, tracking, and recognition into a real-time adaptable system.
- Incorporates robust spatial priors and Kalman filtering.

Weaknesses:

- Computationally intensive.
- Limited flexibility with unusual sign locations.

Comparison:

- The hybrid approach optimizes computational demands while ensuring detection accuracy.

Vision-Based Detection (Section 2.4)

Strengths:

- Hybrid methods combining colour and shape features demonstrate high accuracy in controlled environments.

Weaknesses:

- Struggles with occlusions and deformations.
- Hybrid methods are challenging to tune.

Comparison:

- Refined preprocessing and YOLO11 integration improve adaptability and accuracy.

Machine Vision-Based Methods (Section 2.5)

Strengths:

- Robust across varied conditions using diverse techniques like colour, shape, machine learning, and LIDAR.

Weaknesses:

- Colour methods are sensitive to lighting; shape methods are computationally expensive.
- Machine learning methods rely heavily on large datasets.

Comparison:

- The proposed system mitigates lighting sensitivity and computational inefficiency with a hybrid approach.

YOLO (Section 2.6)

Strengths:

- Real-time performance with high speed and generalization across domains.

Weaknesses:

- Struggles with small or overlapping objects due to coarse feature maps.

Comparison:

- Incorporates preprocessing to address small object detection challenges.

2.7.1 Conclusion

A comprehensive review of prior works in traffic sign detection reveals a rich progression from classical image processing methods to modern deep learning-based frameworks. Traditional approaches, including colour thresholding, shape analysis, and handcrafted feature extraction, offered foundational insights into sign localization and classification, yet struggled under non-ideal conditions such as low illumination, occlusion, and background clutter. Furthermore, these methods typically lacked the scalability and real-time performance required for deployment in dynamic driving environments.

With the emergence of deep learning architectures, particularly those in the YOLO (You Only Look Once) family, a paradigm shift occurred in how object detection is addressed. These models, especially the latest iterations such as YOLOv5 and beyond, demonstrated superior generalization, faster inference speeds, and robust spatial localization capabilities, enabling more accurate and reliable detection even in complex traffic scenes. Among these, YOLO11 has emerged as a highly efficient and scalable object detection model due to its streamlined backbone, enhanced attention modules, and optimized anchor-based prediction strategies.

The insights gathered from earlier literature were instrumental in shaping the direction of this project, which fully embraced the YOLO-based methodology. The limitations of traditional approaches underscored the necessity of adopting a model capable of learning directly from data with minimal manual intervention, thus allowing for better adaptation to varying road conditions, sign deformations, and environmental inconsistencies. By leveraging a deep convolutional network trained on a curated and diverse dataset, the proposed system addresses the shortcomings observed in prior

works, particularly those related to detection robustness and computational overhead.

In conclusion, the transition toward deep learning-based object detection—epitomized by the YOLO11 model—marks a pivotal advancement in the field of traffic sign detection. This project aligns with and builds upon this evolution, offering a modernized solution capable of delivering high-performance detection in real-time, video-based scenarios. The review affirms the obsolescence of earlier methods in favour of more scalable, adaptable, and intelligent systems, establishing a strong foundation for future development within autonomous driving and intelligent transportation applications.

CHAPTER 3 – System Methodology/Approach

3.1 System Design Overview

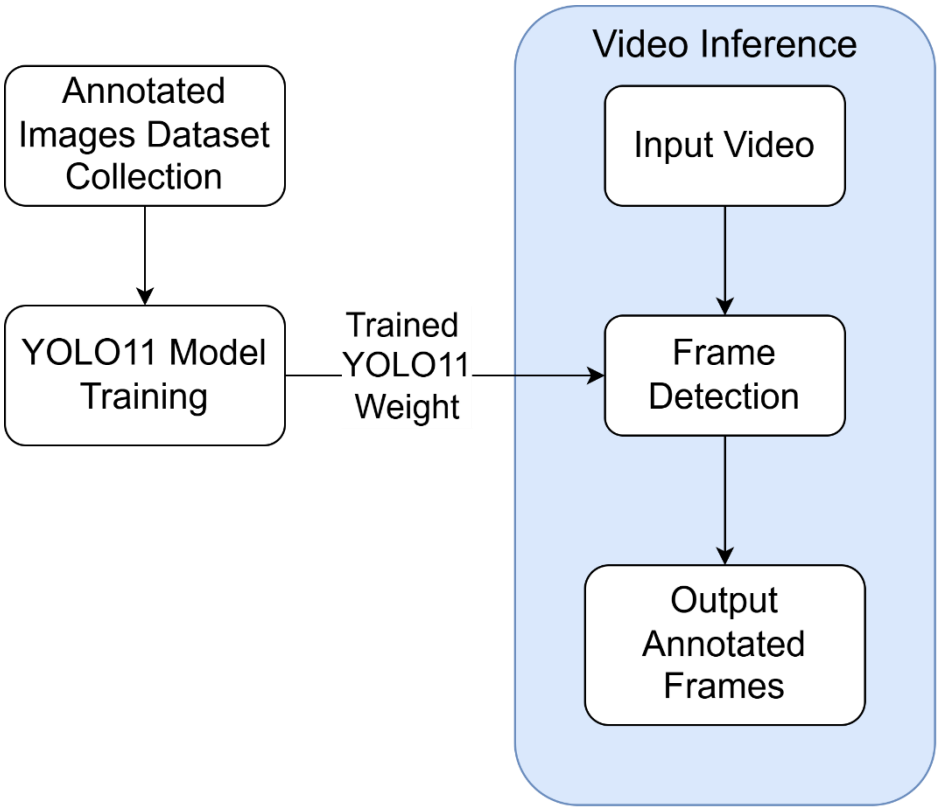


Figure 3.1 System Overview

The process commences with the Annotated Images Dataset Collection, where an extensive volume of traffic sign images is sourced and meticulously labelled using bounding box annotations compliant with YOLO-format specifications. These images, covering a wide array of environmental conditions—including varying illumination, occlusion patterns, and diverse road contexts—form the empirical substrate required to teach the deep learning model to generalize effectively beyond controlled conditions.

Following dataset compilation, the system advances to the YOLO11 Model Training phase, wherein the annotated data is ingested by the YOLO11 architecture. During this stage, the model undergoes supervised learning to optimize its capacity to predict object classes and precise spatial locations. Once the training converges, the Trained YOLO11 Weights are preserved as the learned intelligence that encapsulates the detection model's representational knowledge.

In the operational inference stage, the system accepts an Input Video, which is

segmented into individual image frames for further analysis. The Frame Detection module leverages the pretrained YOLO11 weights to execute detection on each frame independently. For every input frame, the YOLO11 model conducts inference, identifying traffic signs by drawing bounding boxes and assigning corresponding class labels and confidence scores. This single-stage detection paradigm ensures high throughput without compromising on precision, making it ideal for safety-critical autonomous driving applications.

The final component in the workflow is the Output Annotated Frames, where the processed video frames—now embedded with visual detection results—are reassembled into a coherent annotated video stream. These outputs provide intuitive visualization for performance verification, post-processing analysis, and system debugging.

Altogether, this system design leverages the expressive power of deep convolutional neural networks through YOLO11, structured into a logical progression from data-driven model development to high-fidelity detection in video feeds. Its modular decomposition not only facilitates isolated optimization and troubleshooting of each subsystem but also lays a foundational framework for future enhancements, including real-time deployment, integration with ADAS modules, and multi-sensor fusion.

3.2 Model Architecture

The YOLO11 architecture is a state-of-the-art, single-stage object detection model designed specifically for real-time detection tasks critical to autonomous driving. By integrating advanced convolutional neural networks (CNN) and attention mechanisms, YOLO11 efficiently detects and localizes traffic signs from dynamic video streams captured by vehicle-mounted cameras.

At its core, YOLO11 uses a deep CNN backbone enhanced with attention modules, which effectively amplify relevant spatial features and suppress irrelevant background details. This selective attention capability significantly improves the model's accuracy in identifying small, distant, or partially obstructed traffic signs, which are common challenges encountered in realistic driving conditions.

Figure 3.2.1 below illustrates the YOLO11 model architecture, emphasizing its

backbone structure, attention mechanisms, and multi-scale feature integration process:

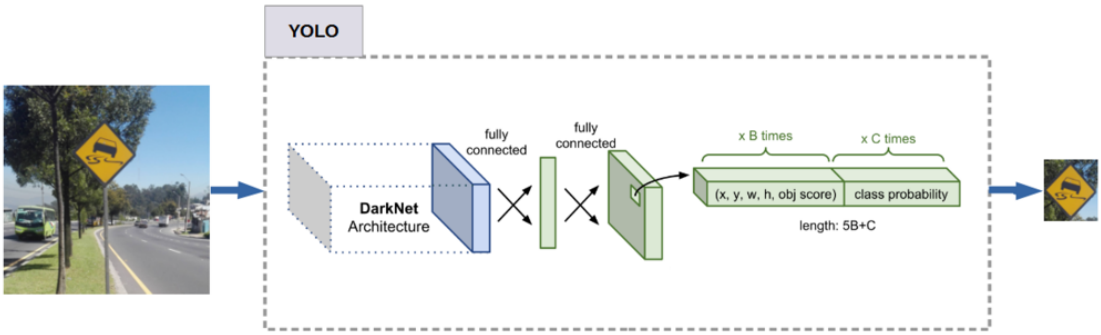


Figure 3.2.1: YOLO11 Architecture Diagram

YOLO11 divides each input frame into a grid of cells, where each cell predicts multiple bounding boxes alongside confidence scores and class probabilities. These predictions rely on adaptively optimized anchor boxes, predefined templates determined through clustering methods tailored specifically to typical traffic sign dimensions. This strategic optimization enhances the precision of bounding box regression, resulting in accurate localization and classification of detected signs.

To manage overlapping and redundant predictions, YOLO11 incorporates an enhanced Non-Maximum Suppression (NMS) algorithm. The improved NMS effectively retains the most accurate bounding boxes by considering the confidence scores and intersection-over-union (IoU) thresholds, generating clear and actionable detections suitable for real-time vehicle decision-making.

A crucial advancement in YOLO11 is its implementation of a Bidirectional Feature Pyramid Network (BiFPN). This sophisticated network fuses detailed spatial information from early convolutional layers with high-level semantic information from deeper layers, thus effectively detecting signs across various sizes and environmental complexities.

Practical results of YOLO11 detection performance in real traffic scenarios are presented in Figure 3.2.2, highlighting the model's ability to robustly identify multiple classes of traffic signs under varying conditions, including different scales and partial occlusions:

Figure 3.2.2: Examples of YOLO Traffic Sign Detection Results

In summary, the YOLO11 architecture provides advanced attention-driven feature extraction, optimized anchor-box prediction, enhanced NMS strategies, and sophisticated multi-scale feature integration. These combined innovations make YOLO11 particularly effective for accurate and rapid detection of traffic signs, directly contributing to improved safety and operational efficiency in autonomous vehicle navigation systems.

Chapter 4 - System Design

4.1 System Module Specifications

4.1.1 Annotated Images Dataset Collection

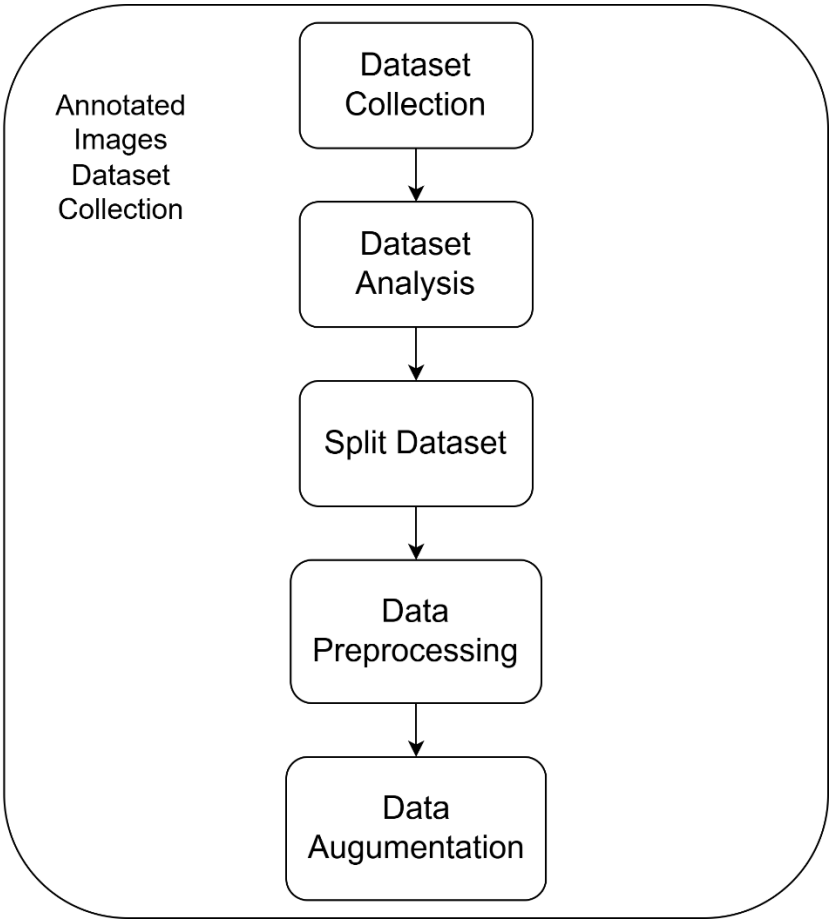


Figure 4.1.1 Dataset Collection Module

In this project, the dataset was sourced and curated through the Roboflow platform, employing an iterative refinement strategy that emphasizes data quality, representational diversity, and contextual relevance. Rather than relying on a monolithic source, the dataset was meticulously assembled by cloning only the most relevant and contextually valid images from multiple existing datasets, each of which contributes distinct scenarios, sign types, and environmental conditions. This selective cloning mechanism ensured the exclusion of noisy, redundant, or semantically ambiguous samples while preserving only those instances that meet stringent criteria for annotation completeness, label clarity, and visual interpretability.

Upon collection, the data pipeline progresses to the dataset analysis stage, where the curated images undergo visual inspection and statistical verification. Each image is annotated with precise bounding boxes and class labels according to YOLO specifications, forming the structured basis for supervised learning. Special attention is given to maintaining inter-class balance to mitigate the model's tendency toward bias and class imbalance—a common pitfall in object detection tasks involving traffic signs with inherently skewed occurrence rates.

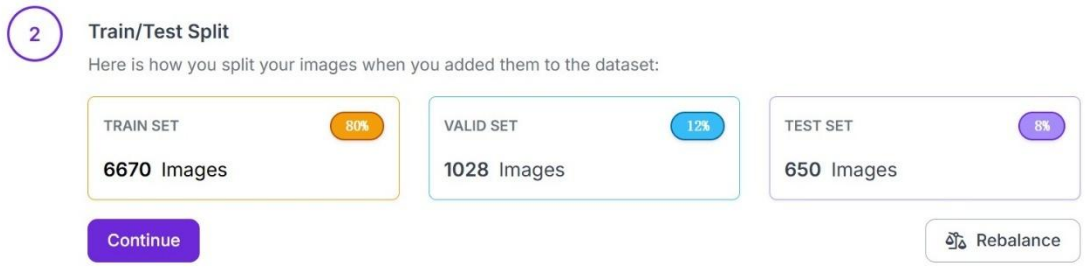


Figure 4.1.1.1 Splitting Dataset

Following validation, the images are partitioned through a controlled dataset splitting process. As configured within Roboflow and verified in Figure 4.1.1.1, the dataset is divided into a training set comprising 6670 images (80%), a validation set of 1028 images (12%), and a test set of 650 images (8%). This proportional division ensures that the model receives an ample volume of diverse training data while preserving enough unseen samples for reliable validation and performance benchmarking. The splitting algorithm used ensures stratified distribution, such that the frequency of each traffic sign class is proportionally maintained across all three subsets.

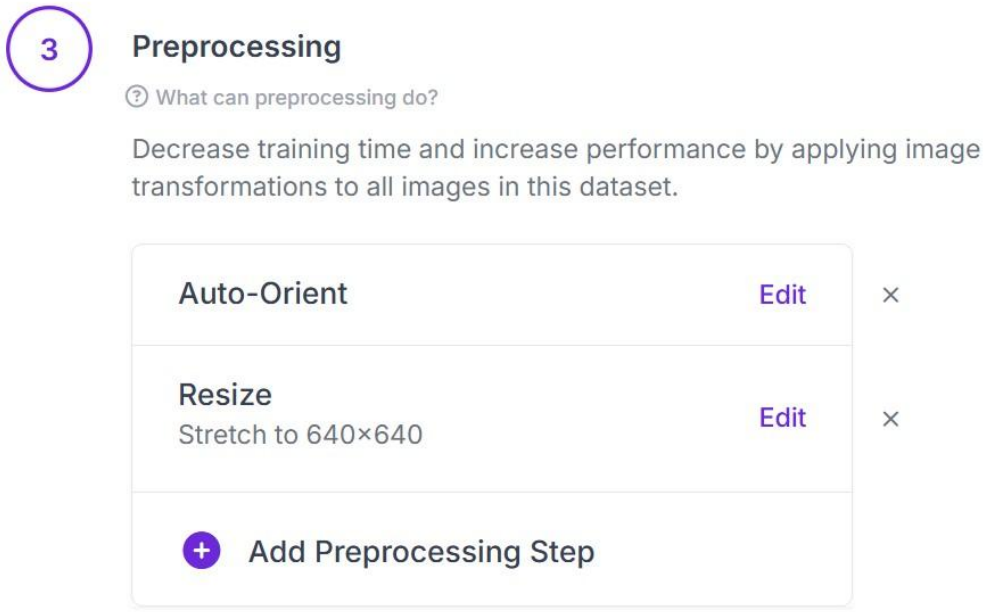


Figure 4.1.1.2 Dataset Preprocessing

Subsequently, the dataset undergoes data preprocessing, which is designed to standardize input dimensions and image orientation, thereby facilitating stable model convergence during training. Each image is subjected to auto-orientation correction followed by resizing to a fixed resolution of 640×640 pixels, as shown in Figure 4.1.1.2. This resolution was chosen to align with the input requirements of the YOLO11 architecture, striking an optimal balance between feature resolution and computational efficiency.

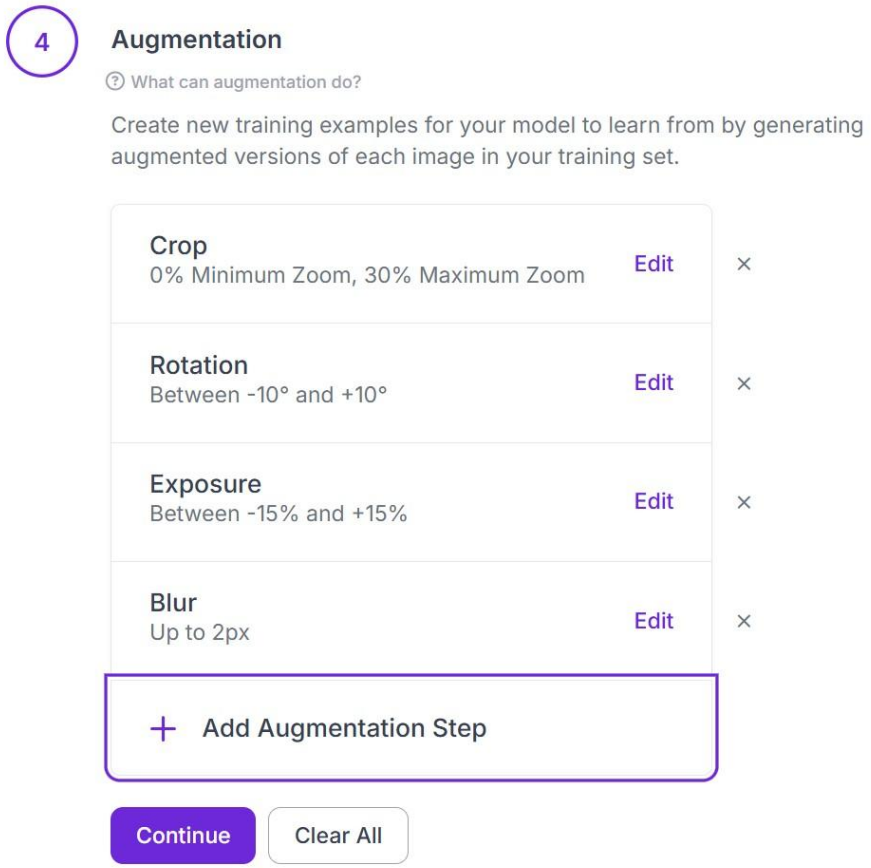


Figure 4.1.1.3 Dataset Augmentation

To further enhance model generalization and resilience against real-world perturbations, data augmentation techniques are applied to the training set. As detailed in Figure 4.1.1.3, the augmentation pipeline includes controlled cropping (0% minimum zoom, 30% maximum zoom), rotation within the range of –10° to +10°, exposure adjustments from –15% to +15%, and a blur filter up to 2 pixels. These augmentations are algorithmically applied to synthetically diversify the training data, simulating a wide array of real-world conditions such as motion blur, lighting variation, and angular displacement. The intention is not merely to increase data volume, but to cultivate a model that is resilient, adaptable, and highly tolerant to noise and distortion, as typically encountered in in-the-wild driving environments.

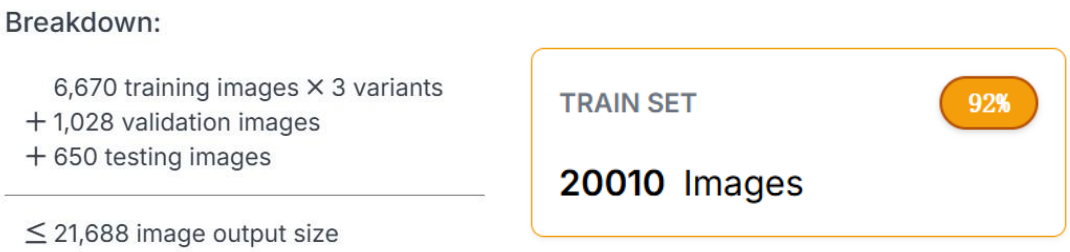


Figure 4.1.1.4 Final Daset Size Calculation

As a direct result of this augmentation pipeline, the total dataset size expanded substantially. Specifically, the original 6670 training images were transformed into three augmented variants per image, resulting in an effective training subset of approximately 20,010 samples. When combined with the untouched validation set (1028 images) and test set (650 images), the final post-augmentation dataset comprises up to 21,688 image instances. This enlarged dataset plays a critical role in regularizing the model, enriching its training exposure, and fostering high generalization accuracy across varying spatial, temporal, and lighting conditions.

4.1.2 YOLO11 Model Training Module

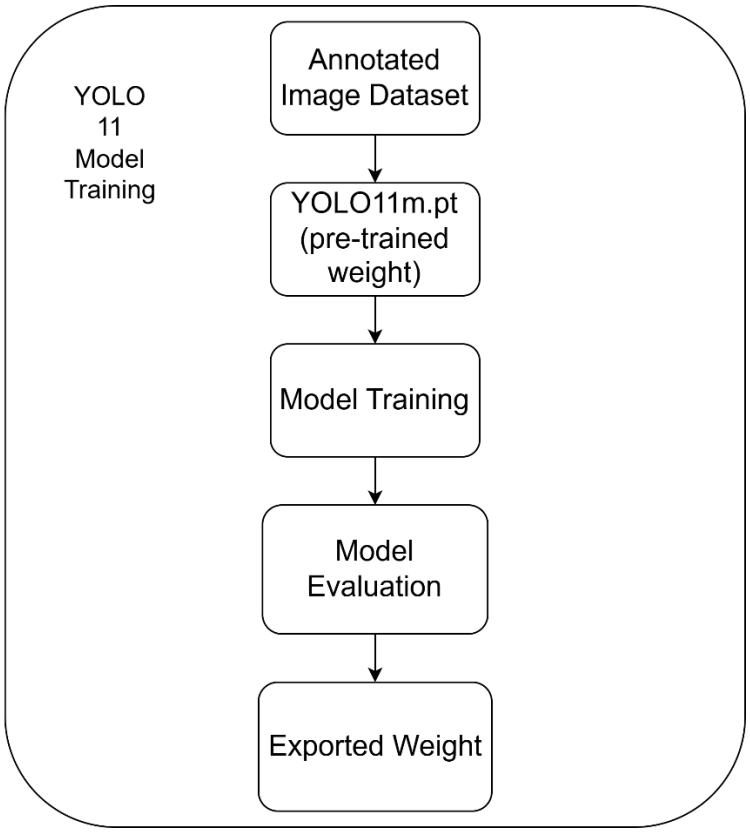


Figure 4.1.2 YOLO11 Model Training Module

```
!pip install roboflow

from roboflow import Roboflow
rf = Roboflow(api_key="RwI139D9dtYOsVUHcZOM")
project = rf.workspace("new-fyp-cidcn").project("fyp-combined-dataset")
version = project.version(2)
dataset = version.download("yolov11")
```

Figure 4.1.2.1 Download Dataset from Roboflow

The training phase initiates with the downloading of a combined traffic sign dataset hosted on the Roboflow platform. The dataset is programmatically accessed through the Roboflow API, specifying version control and compatibility with the YOLOv11 format. This dataset, which has undergone extensive preprocessing, class balancing, and augmentation as detailed in Section 4.1.1, is used to train the model from a foundational checkpoint based on yolo11m.pt—a medium-sized variant of YOLO11 pre-trained on a general-purpose object detection corpus.

```
!pip install ultralytics
import ultralytics
from ultralytics import YOLO
from IPython.display import Image, Video

!yolo task=detect mode=train data={dataset.location}/data.yaml model="yolo11m.pt" epochs=40 imgsz=640 batch=16
```

Figure 4.1.2.2 Training YOLO model

Model training is conducted using the YOLO CLI interface with the following key hyperparameters: epochs=40, imgsz=640, batch=16, and the model="yolo11m.pt" flag to specify the initialization weights. The training command is executed in Colab using GPU acceleration, and the training results—including best-performing weights, loss curves, and evaluation metrics—are saved under the project directory /runs/detect/train. During training, the model iteratively optimizes its parameters using a default AdamW optimizer, minimizing a compound loss function comprising bounding box regression loss, object confidence loss, and classification loss.

The YOLO11 model's optimization routine is guided by backpropagation and stochastic gradient-based updates, iterating over mini-batches of size 16. Each batch feeds forward through the convolutional network, produces predictions for bounding boxes and class probabilities, computes error signals based on ground truth annotations, and backpropagates gradients to update network weights accordingly.

Over 40 epochs, the model converges to a local minimum where its performance on both training and validation sets is maximized.

Throughout the training process, model checkpoints are periodically saved, and performance metrics are monitored on the validation set. Upon the completion of the training routine, the system identifies the optimal model state—corresponding to the lowest validation loss or highest mAP—and exports this configuration as best.pt. This file encapsulates the learned representations of traffic sign characteristics and serves as the final output of the training module.

4.1.3 Video-Based Inference Module

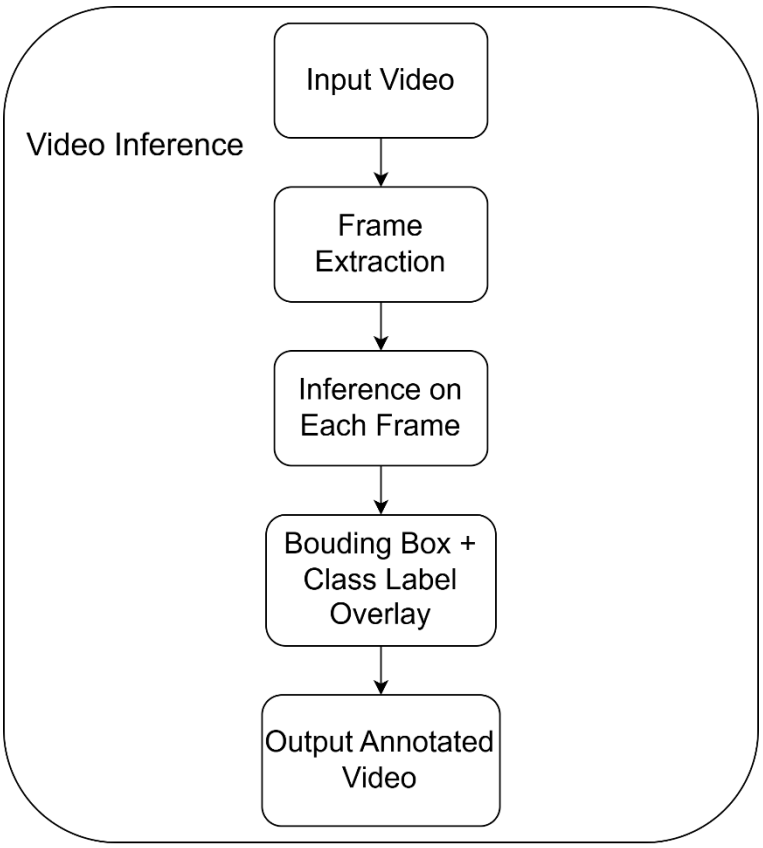


Figure 4.1.3 Video-Based Inference Module

The video-based inference module constitutes the operational deployment layer of the traffic sign detection system, wherein the trained YOLO11 model (best.pt) is utilized to perform object detection on traffic signs from video input. This module is designed to transform continuous video streams into frame-wise visual insights, enabling the model to interpret, localize, and annotate traffic signs encountered in dynamic environments. It is the culmination of the prior data preparation and model training stages and serves as the interface through which detection results are rendered for both analysis and demonstration.

The inference process begins with the acquisition of a pre-recorded video stream that simulates a vehicular driving scenario. The video, once loaded into the system, is decomposed into a sequential set of static image frames. Each of these frames is treated as an individual inference unit, decoupling temporal continuity in favour of spatial object recognition. This design decision simplifies implementation while ensuring that each frame is independently processed, preserving inference stability even in the presence of motion blur or sudden scene transitions.

```
video 1/1 (frame 45699/45709) /content/drive/MyDrive/FYPRelated/InferenceVideos/GermanTest.mp4: 384x640 (no detections), 13.0ms
video 1/1 (frame 45700/45709) /content/drive/MyDrive/FYPRelated/InferenceVideos/GermanTest.mp4: 384x640 (no detections), 13.3ms
video 1/1 (frame 45701/45709) /content/drive/MyDrive/FYPRelated/InferenceVideos/GermanTest.mp4: 384x640 (no detections), 13.0ms
video 1/1 (frame 45702/45709) /content/drive/MyDrive/FYPRelated/InferenceVideos/GermanTest.mp4: 384x640 (no detections), 12.9ms
video 1/1 (frame 45703/45709) /content/drive/MyDrive/FYPRelated/InferenceVideos/GermanTest.mp4: 384x640 (no detections), 13.5ms
video 1/1 (frame 45704/45709) /content/drive/MyDrive/FYPRelated/InferenceVideos/GermanTest.mp4: 384x640 (no detections), 12.9ms
video 1/1 (frame 45705/45709) /content/drive/MyDrive/FYPRelated/InferenceVideos/GermanTest.mp4: 384x640 (no detections), 13.8ms
video 1/1 (frame 45706/45709) /content/drive/MyDrive/FYPRelated/InferenceVideos/GermanTest.mp4: 384x640 (no detections), 13.2ms
video 1/1 (frame 45707/45709) /content/drive/MyDrive/FYPRelated/InferenceVideos/GermanTest.mp4: 384x640 (no detections), 13.1ms
video 1/1 (frame 45708/45709) /content/drive/MyDrive/FYPRelated/InferenceVideos/GermanTest.mp4: 384x640 (no detections), 13.2ms
video 1/1 (frame 45709/45709) /content/drive/MyDrive/FYPRelated/InferenceVideos/GermanTest.mp4: 384x640 (no detections), 13.2ms
Speed: 2.4ms preprocess, 13.2ms inference, 0.8ms postprocess per image at shape (1, 3, 384, 640)
Results saved to runs/detect/predict3
```

Figure 4.1.3.1 Inference Process

The core of the module is the YOLO11 detection engine, which accepts each video frame as input and performs bounding box regression, objectness estimation, and class probability assignment based on the parameters encoded in the trained weights. The detection model infers the presence and category of traffic signs in the frame, returning coordinates for each detected object along with a corresponding confidence score. To reduce visual noise and false positives, the outputs are filtered using non-maximum suppression (NMS), which retains only the most confident detection for overlapping bounding boxes.

Figure 4.1.3.2 Example of Annotated Frames

Once processed, the inference results are visually overlaid onto the original frames. Each detected traffic sign is enclosed in a bounding box, labelled with its class name and confidence score. The annotated frames are then recompiled into a coherent video stream that mirrors the original input but is enriched with semantic interpretation. The final output video is encoded using standard codecs (e.g., H.264 via FFmpeg) and stored in a compressed format for evaluation, visualization, or downstream application in intelligent transportation systems.

```
from IPython.display import HTML
from base64 import b64encode
import os

# Input video path
save_path = '/content/runs/detect/predict3/GermanTest.avi'

# Compressed video path
compressed_path = "/content/drive/MyDrive/FYPRelated/PredictedVideos/GTWith19C.mp4"

!ffmpeg -i {save_path} -vcodec libx264 {compressed_path}

import os
print(os.getcwd())

# List files in the current directory and the target directory to check if the file exists
print(os.listdir('.'))
print(os.listdir('/content/drive/MyDrive/FYPRelated/PredictedVideos/'))

# Show video
mp4 = open(compressed_path,'rb').read()
data_url = "data:video/mp4;base64," + b64encode(mp4).decode()
HTML("""
<video width=400 controls>
      <source src="%s" type="video/mp4">
</video>
""" % data_url)
```

Figure 4.1.3.3 Code to Compressed Inference Video

CHAPTER 5 – System Implementation

5.1 Hardware Setup

The hardware used for the project is a laptop with high-performance specifications suitable for intensive image processing tasks. The detailed specifications of the hardware used are listed below:

Table 5.1 Laptop Specifications

Brand	Predator PHN16-71
Operating System	Microsoft Windows 11 Home Single Language (x64)
Processor (CPU)	13th Gen Intel(R) Core (TM) i7-13700HX @ 2.10 GHz
Graphics (GPU)	NVIDIA GeForce RTX 4060 Laptop (AD107M/GN21-X4)
Memory (RAM)	2x16GB DDR5 SDRAM 2400MHz
Storage	2x1TB SSD WD PC SN810 SDCQNRY-1T00-1014

5.2 Software Setup

The software setup for this project is critical to ensure that the system can efficiently handle large datasets and accurately detect and segment traffic signs. The project is developed in a Python environment, leveraging open-source libraries specifically designed for computer vision tasks. The development was conducted using Google Colab, a cloud-based Jupyter Notebook environment, which provided an interactive environment ideal for iterative testing, visualization, and refinement of the algorithms with the access to GPU and TPU resources, making it ideal for machine learning and deep learning tasks.

Below is a detailed table summarizing the software tools and libraries used in the system implementation:

Table 5.2 Software Specifications

Category	Description	Version
Operating System	Microsoft Windows 11 Home Single Language (x64)	22H2 (Build: 22621.2134)
Programming Language	Python 3.10	3.10.12
Development Environment	Google Colab with Jupyter Notebook	6.5.5 (notebook version)
Libraries	- **OpenCV**: For image processing tasks	4.10.0
	- **NumPy**: For numerical computations and matrix operations	1.26.4
	- **Matplotlib**: For visualizing images and bounding boxes	3.8.0
Libraries	- **ultralytics**: For implementing with the YOLO model	8.3.40
Model Architecture	YOLO11	YOLO11m
Dataset	fyp-combined-dataset (Managed via Roboflow)	v2

Libraries

OpenCV (Open Source Computer Vision Library) is a powerful and popular open-source library for computer vision and image processing tasks. It contains a comprehensive set of functions for image manipulation, filtering and more, making it the best choice for developing our project. Especially in colour space conversions, edge detection algorithms (like Canny), contour analysis, and bounding box generation will be utilised in our project.

Development Environment

Google Colab is a free cloud-based platform that allows users to write and execute Python code in a Jupyter Notebook environment. It provides access to powerful hardware accelerators like GPUs and TPUs, which can significantly speed up computations, especially for deep learning tasks. Colab integrates seamlessly with Google Drive, making it easy to manage and share files. It also supports collaboration, allowing multiple users to work on the same notebook simultaneously. With pre-installed libraries and an easy-to-use interface, Google Colab is an excellent tool for data scientists, researchers, and developers looking to leverage the power of cloud computing for their projects.

Dataset Management

Roboflow is a cloud-based computer vision platform designed to streamline the end-to-end workflow of dataset management, model training, and deployment for object detection, classification, and segmentation tasks. In the context of this project, Roboflow plays a pivotal role in the curation, preprocessing, augmentation, and formatting of the annotated image dataset used to train the YOLO11 detection model. It provides an intuitive interface and API support for importing diverse image sources, applying consistent annotations, and automatically generating YOLO-compatible dataset structures. Through its built-in tools, the platform enables users to perform critical preprocessing operations such as image resizing and auto-orientation, as well as advanced augmentation techniques like rotation, cropping, exposure correction, and blur simulation. Moreover, Roboflow facilitates seamless dataset version control and automated splitting into training, validation, and test sets, ensuring class-balanced partitions essential for supervised learning. Overall, Roboflow serves as the data backbone of this project, enabling a scalable and reproducible pipeline for high-quality dataset preparation in support of deep learning-based traffic sign detection from video.

5.3 Setting and Configuration

The training process was conducted on the Google Colab Pro platform, leveraging a high-performance NVIDIA A100 GPU backend, which enabled efficient parallelized training using mixed-precision arithmetic. The YOLO11 model architecture was instantiated using the Ultralytics implementation and initialized with the pretrained weight file yolo11m.pt—a medium-complexity model offering a balance between speed and accuracy. The annotated image dataset, prepared and augmented via the

Roboflow platform, was downloaded in YOLOv11-compatible format and included 18 distinct traffic sign classes, pre-stratified into training, validation, and testing subsets.

The training hyperparameters were configured as follows:

Table 5.3 Model training parameters

Parameter	Value
imgsize	640x640
Batch	16
epochs	40
lr0	0.01

The default AdamW optimizer was employed, which utilizes adaptive learning rates in conjunction with decoupled weight decay regularization, ensuring stable and efficient convergence during backpropagation. The model was trained using YOLO's CLI interface in task=detect mode, with full logging enabled for performance tracking.

Inference was conducted on the same platform, using the best-trained model checkpoint (best.pt) derived from the training phase. During inference, video files were decomposed into frames and passed sequentially through the YOLO11 model using a confidence threshold of 0.25, with non-maximum suppression (NMS) activated to eliminate redundant bounding boxes. The system generated annotated video outputs with detected traffic signs clearly enclosed in labelled bounding boxes, preserving detection fidelity across frames.

All system paths, including dataset directory references and export locations for weights and videos, were configured relative to the Colab working directory and integrated with Google Drive for persistent storage. This configuration not only ensured reproducibility across training sessions but also facilitated streamlined transfer of outputs between local and cloud-based environments.

5.4 System Operation

The operational execution of the traffic sign detection system is conducted in a structured sequence that transitions the trained YOLO11 model from a passive weight file (best.pt) into an active inference engine capable of processing and interpreting

visual input from video streams. This system operation was carried out within the Google Colab environment, taking advantage of its integrated GPU acceleration, seamless directory management, and support for media rendering. The following subsections delineate the procedural steps undertaken during system execution, supported by appropriate visual documentation.

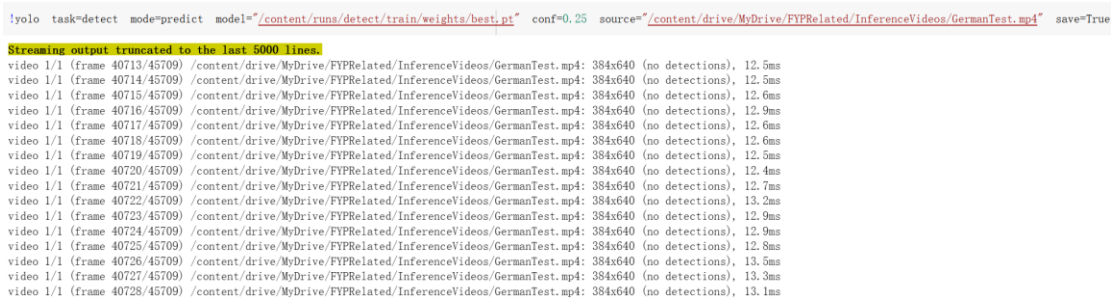


Figure 5.4.1 Colab execution cell running YOLOv11 prediction command

Figure 5.4.2 Console output showing path to output video

The process initiates with the mounting of Google Drive within the Colab workspace to access the trained model weights and the inference video files. After initializing the YOLO environment via the Ultralytics library, the system loads the best.pt checkpoint and configures the detection task. A pre-recorded video—representing a simulated driving scenario—is provided as input. The video is then segmented into individual frames, which are passed sequentially through the YOLO11 detection pipeline. Each frame is evaluated independently, allowing for spatially localized detections without temporal dependency.

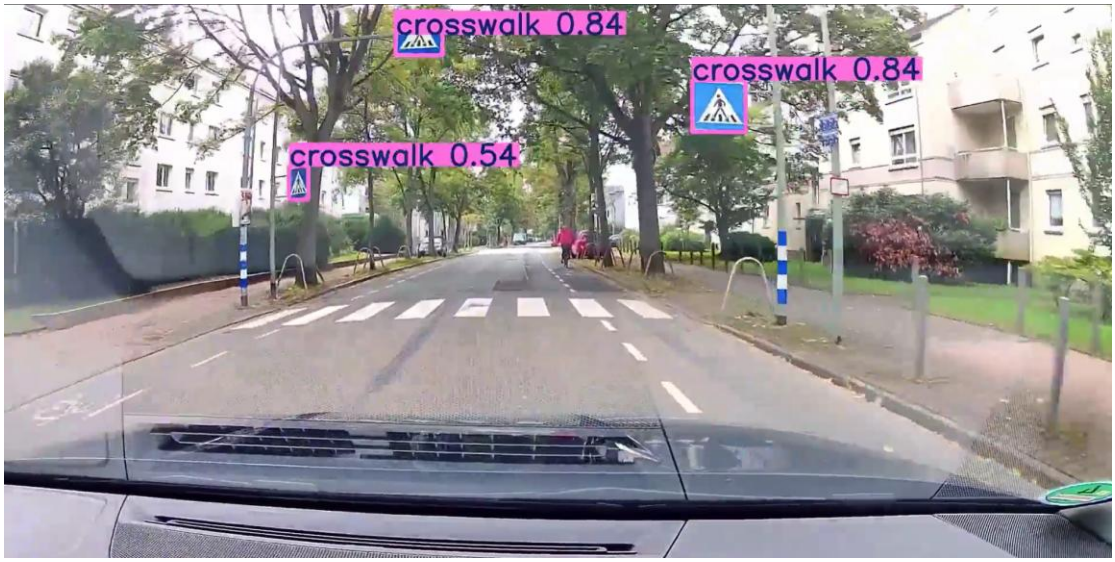


Figure 5.4.3 Sample frame from the output video showing bounding boxes on detected traffic sign

Once inference is complete for each frame, the system overlays the bounding boxes and class labels corresponding to the detected traffic signs. The predictions are visually encoded using color-coded annotations and confidence scores, thereby enabling rapid human interpretability of detection performance. After all frames are processed, the annotated outputs are recompiled into a video stream using FFmpeg and saved in .mp4 format for efficient compression and playback.

```
Output #0, mp4, to '/content/drive/MyDrive/FYPRelated/PredictedVideos/GTWith19C.mp4':
  Metadata:
    software        : Lavf59.27.100
    encoder         : Lavf58.76.100
  Stream #0:0: Video: h264 (avc1 / 0x31637661), yuvj420p(pc, bt470bg/unknown/unknown, progressive), 1920x1080, q=2-31, 25 fps, 12800 tbn
    Metadata:
      encoder         : Lavc58.134.100 libx264
    Side data:
      cpb: bitrate max/min/avg: 0/0/0 buffer size: 0 vbv_delay: N/A
frame=45709 fps= 38 q=-1.0 Lsize= 2529177kB time=00:30:28.24 bitrate=11332.8kbits/s speed=1.51x
video:2528623kB audio:0kB subtitle:0kB other streams:0kB global headers:0kB muxing overhead: 0.021903%
[libx264 @ 0x56c6ba835e00] frame I:187   Avg QP:21.33  size:338120
[libx264 @ 0x56c6ba835e00] frame P:14086 Avg QP:24.88  size:109155
[libx264 @ 0x56c6ba835e00] frame B:31436 Avg QP:28.03  size: 31446
[libx264 @ 0x56c6ba835e00] consecutive B-frames:  4.8%  7.9%  7.8% 79.6%
[libx264 @ 0x56c6ba835e00] mb I  I16..4:  7.0% 81.8% 11.2%
[libx264 @ 0x56c6ba835e00] mb P  I16..4:  2.8% 20.7%  3.0%  P16..4: 26.2% 12.1% 10.0%  0.0%  0.0%    skip:25.2%
[libx264 @ 0x56c6ba835e00] mb B  I16..4:  0.9%  4.1%  0.6%  B16..8: 36.1%  8.3%  3.8%  direct: 2.8%  skip:43.5%  L0:48.1% L1:40.2% BI:11.7%
[libx264 @ 0x56c6ba835e00] 8x8 transform intra:77.1% inter:58.6%
[libx264 @ 0x56c6ba835e00] coded y,uvDC,uvAC intra: 67.9% 56.3% 6.2% inter: 20.7% 10.2% 0.3%
[libx264 @ 0x56c6ba835e00] i16 v,h,dc,p: 26% 40% 17% 16%
[libx264 @ 0x56c6ba835e00] i8 v,h,dc,ddl,ddr,vr,hd,vl,hu: 18% 27% 32%  3%  3%  3%  6%  3%  6%
[libx264 @ 0x56c6ba835e00] i4 v,h,dc,ddl,ddr,vr,hd,vl,hu: 23% 26% 15%  5%  6%  5%  9%  4%  7%
[libx264 @ 0x56c6ba835e00] i8c dc,h,v,p: 49% 30% 17%  4%
[libx264 @ 0x56c6ba835e00] Weighted P-Frames: Y:1.0% UV:0.4%
[libx264 @ 0x56c6ba835e00] ref P L0: 64.5% 16.7% 13.8%  4.9%  0.0%
[libx264 @ 0x56c6ba835e00] ref B L0: 89.1%  9.2%  1.7%
[libx264 @ 0x56c6ba835e00] ref B L1: 95.3%  4.7%
[libx264 @ 0x56c6ba835e00] kb/s:11329.54
```

Figure 5.4.4 Compressed output video

The output video is automatically stored in the specified Google Drive directory (/PredictedVideos/) to allow for persistence beyond session expiration. To improve

accessibility and reduce storage footprint, the final annotated video is transcoded using the H.264 codec and compressed without loss of detection fidelity. The result is a fully processed visual demonstration of traffic sign detection applied to real-world video input.

5.5 Implementation Issues and Challenges

Despite the structured methodology and the systematic deployment of modern deep learning tools, the implementation of the traffic sign detection system was not devoid of technical intricacies and unforeseen impediments. As is characteristic of data-driven systems operating at the intersection of real-world perception and artificial intelligence, the challenges encountered during this project spanned across multiple phases of the development lifecycle—ranging from data acquisition and model training to inference stability and output fidelity.

A significant early-stage challenge arose during the dataset curation and annotation phase, particularly due to the diversity and inconsistency inherent in traffic sign datasets sourced from open repositories. While the Roboflow platform provided an invaluable interface for dataset assembly and augmentation, it was observed that many original datasets featured poorly annotated samples, ambiguous class definitions, or underrepresented categories. This necessitated a selective cloning and manual inspection process to ensure only semantically meaningful and structurally complete images were included. The problem of class imbalance further complicated the dataset design, as several rare traffic signs—though critical in real-world scenarios—were sparsely represented, thus threatening to introduce learning bias into the model.

During the training phase, the model initially demonstrated instability in convergence behaviour. Fluctuating validation loss and inconsistent mAP scores were observed across early epochs, which suggested the presence of noisy labels and overly aggressive augmentations that distorted key sign features beyond interpretability.

The inference module also presented non-trivial complexities, particularly in the context of frame-by-frame processing of high-resolution videos. Variations in frame lighting, occlusions, and motion blur frequently tested the robustness of the trained YOLO11 model, occasionally resulting in false negatives—especially for signs positioned at oblique angles or partially truncated. While the system performed

admirably under most conditions, these edge cases highlighted the need for future integration of temporal smoothing techniques or multi-frame consensus mechanisms to enhance detection reliability in video streams.

CHAPTER 6 – System Evaluation and Discussion

6.1 System Testing and Performance Metrics

To substantiate the efficacy and operational readiness of the developed traffic sign detection system, a rigorous system testing phase was conducted using the final trained YOLO11 model. The model was evaluated on a reserved validation set comprising unseen data, thereby offering an unbiased estimate of generalization capacity and detection fidelity under practical conditions. The evaluation relied on a suite of standard object detection metrics, including Precision, Recall, mean Average Precision at IoU threshold 0.50 (mAP@50), and mean Average Precision across IoU thresholds from 0.50 to 0.95 (mAP@50–95).

At the conclusion of training (Epoch 40), the system achieved a precision of 96.83%, a recall of 97.33%, and an exceptionally high mAP@50 of 98.67%, reflecting a strong ability to correctly localize and classify most traffic signs within the validation set. The more stringent mAP@50–95 score of 90.83% further affirmed the model's robustness across various IoU thresholds, indicating a high level of consistency in detecting signs of varying scales and positional offsets.

The loss components at the final epoch also converged effectively, with a bounding box loss of 0.4366, classification loss of 0.2236, and distribution focal loss (DFL) of 0.9504, suggesting that the model had reached an optimized state with no signs of overfitting. The performance progression over training epochs is illustrated in Figure 6.1.1, which shows the decreasing loss curves and upward trend in precision and mAP metrics across the 40-epoch timeline.

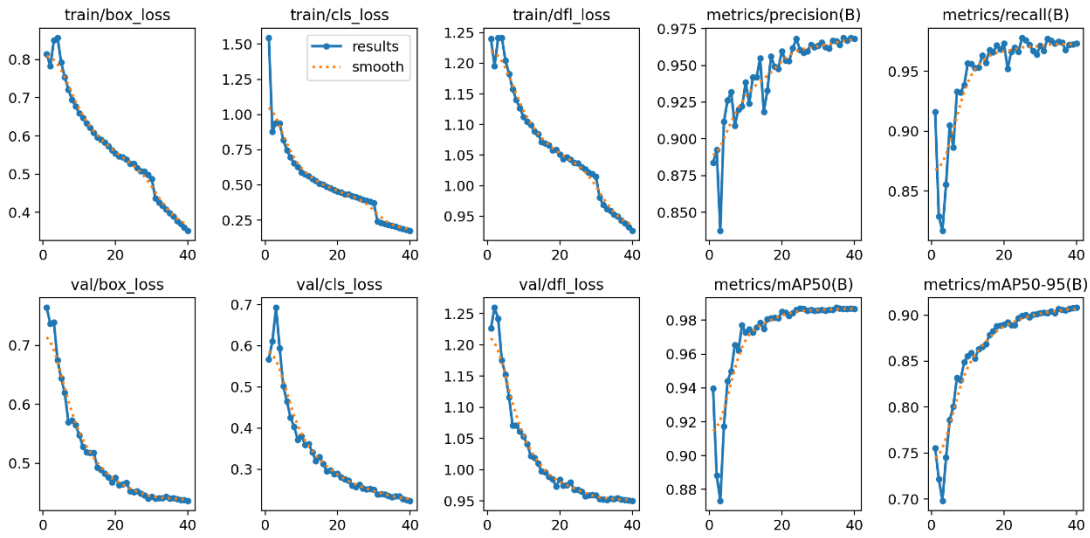


Figure6.1.1 YOLO11 training performance overview—loss convergence and mAP progression across epochs

To further contextualize model behaviour, the confusion matrix presented in Figure 6.1.2 offers insight into class-specific detection performance. It highlights strong diagonal dominance, indicating high precision across most classes, though minor confusion is noted between semantically and visually similar signs such as "No Entry" and "No U-Turn."

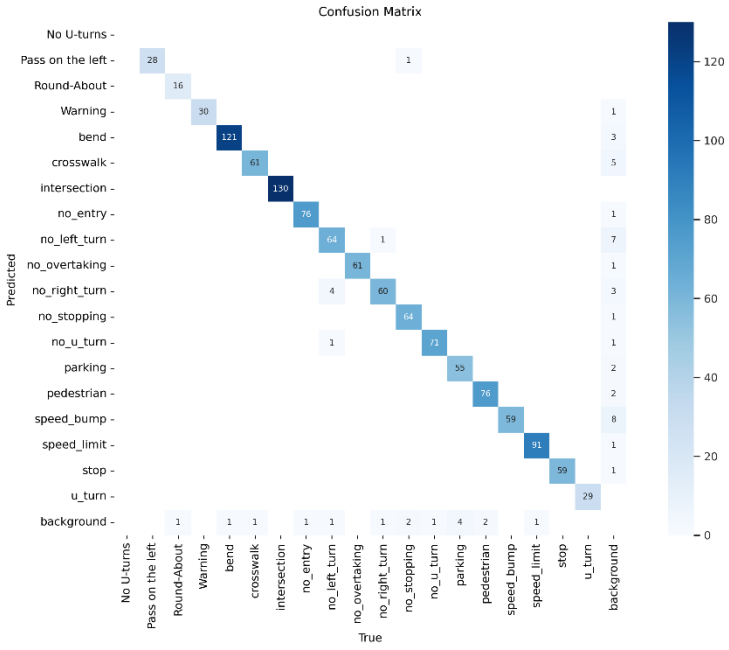


Figure 6.1.2 Confusion matrix showing class-wise detection accuracy and inter-class misclassifications

The normalized confusion matrix, shown in Figure 6.1.3, aids in interpreting relative performance by visualizing the proportion of correct predictions per class.

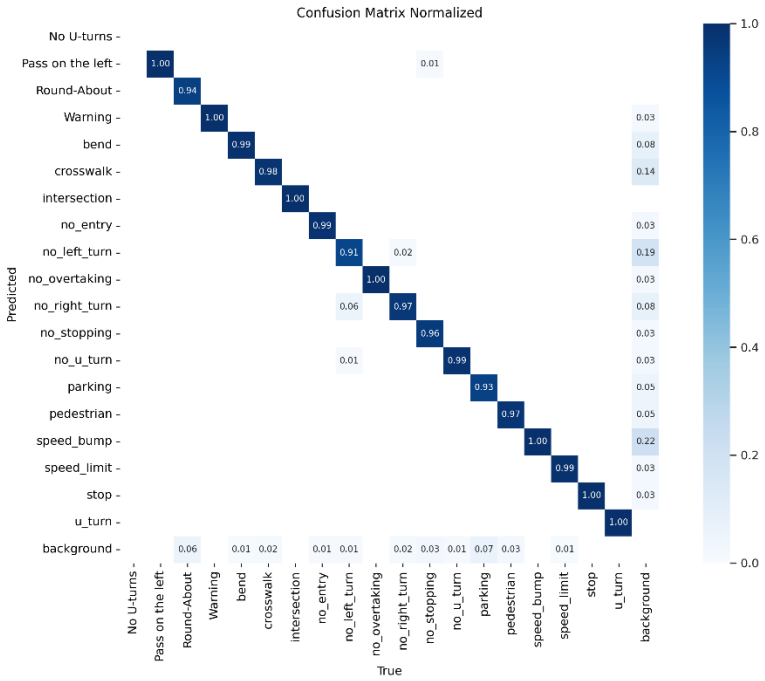


Figure 6.1.3 Normalized confusion matrix representing per-class detection proportions

In addition, the precision-recall (PR) curve in Figure 6.1.4 further demonstrates the model's discriminative capacity across traffic sign categories, with most classes achieving high area-under-curve (AUC) values, reinforcing the reliability of prediction confidence.

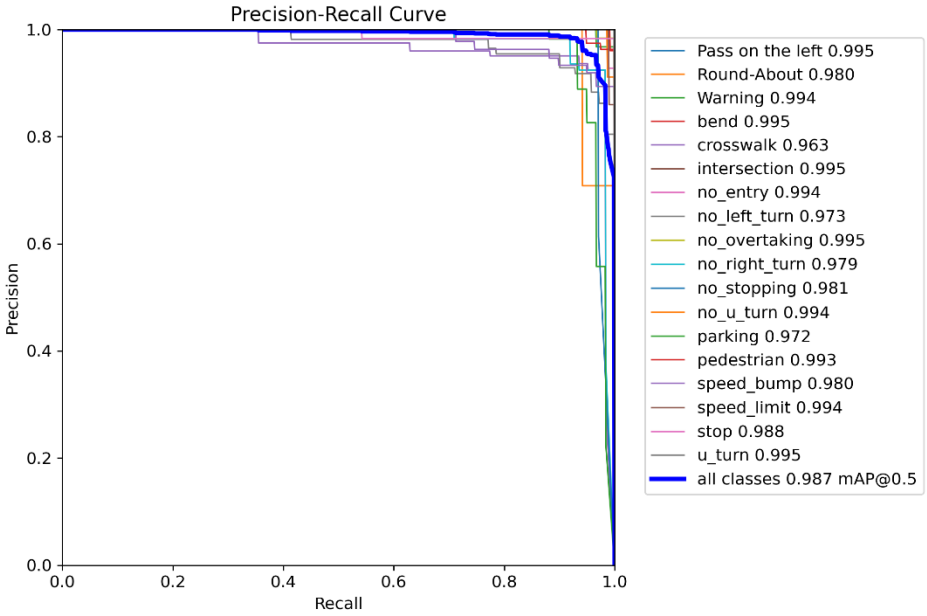


Figure 6.1.4 Precision–Recall (PR) curves across all detected classes

Complementary to the PR curve, the F1 score curve in Figure 6.1.5 confirms the model's balanced performance between precision and recall for each class, particularly emphasizing stable performance on critical regulatory signs.

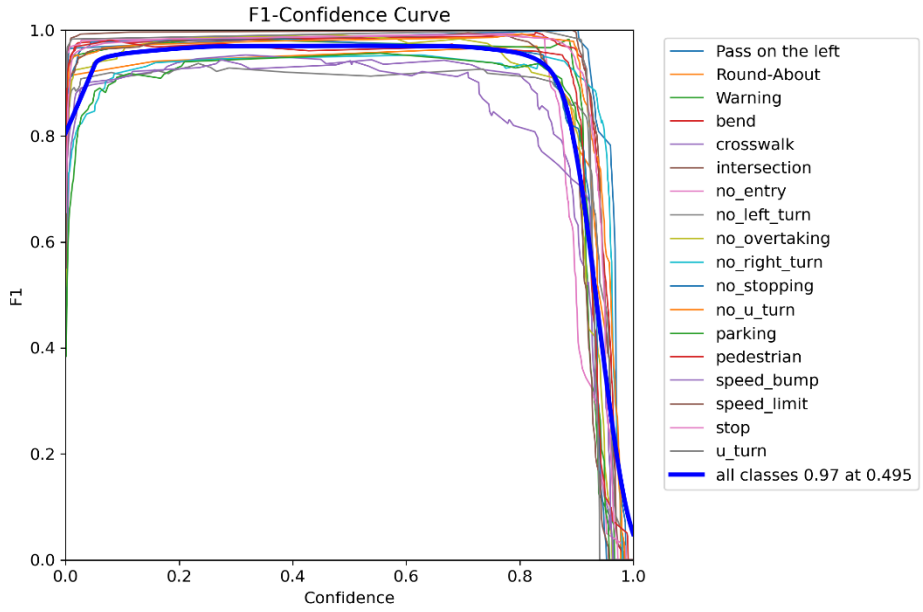


Figure 6.1.5 F1-score per class, reflecting the harmonic balance between precision and recall

To qualitatively assess detection integrity, a series of validation predictions were visualized. As shown in Figure 6.1.6, the system demonstrates consistent localization

of traffic signs across diverse scene contexts. Each bounding box is rendered with its predicted class and confidence score, aligning accurately with ground truth annotations and affirming spatial alignment.

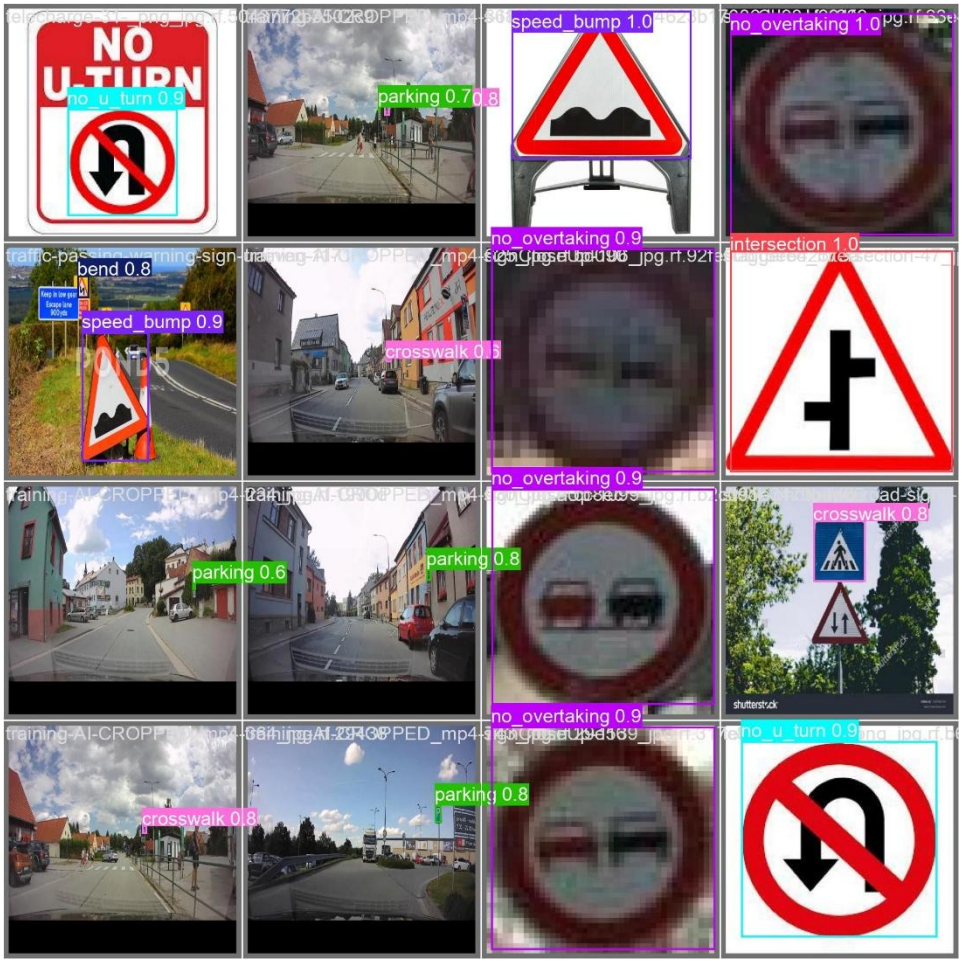


Figure 6.1.6 Sample validation predictions (val_batch0_pred.jpg)—bounding boxes and class labels rendered on test frames

Collectively, these results underscore the YOLO11 model's high detection precision, spatial sensitivity, and classification reliability in video-derived traffic sign detection. The synergy between quantitative metrics and visual evidence confirms the system's readiness for integration into intelligent perception modules within autonomous vehicle architectures, subject to future real-time testing and platform-specific optimization.

6.2 Objectives Evaluation

The first objective was to curate and preprocess a high-quality image dataset comprising 18 traffic sign classes under varied environmental conditions, including different lighting, occlusions, and backgrounds. This goal was systematically fulfilled

through the integration of Roboflow as the dataset management and augmentation platform. Traffic sign images were selectively cloned from multiple publicly available sources and then meticulously labelled to ensure semantic consistency and bounding box precision. The dataset was further enriched through controlled augmentation strategies—such as exposure adjustment, rotation, blur simulation, and cropping—to simulate environmental variability without compromising structural integrity. This preprocessing effort produced a robust, class-balanced dataset with sufficient variance to train a generalizable model, thereby achieving the intended data quality and diversity.

The second objective was to train and fine-tune the YOLO11 model using the prepared dataset, optimizing for detection performance through data augmentation, hyperparameter tuning, and iterative model validation. This was successfully realized through a well-calibrated training process conducted over 40 epochs using Google Colab's A100 GPU backend. The model was initialized with pretrained yolo11m.pt weights and configured with a batch size of 16 and image resolution of 640×640. Augmented data was incorporated directly into the training loop to improve robustness, while training parameters such as learning rate and optimizer settings were selected to ensure efficient convergence. The final model exhibited low loss values and consistent learning trends, confirming that both the training and fine-tuning objectives were rigorously met.

The third objective aimed to evaluate the trained model using industry-standard performance metrics, including precision, recall, mean average precision (mAP), F1-score, and Intersection over Union (IoU), ensuring the system's reliability and robustness across different sign categories. The model's performance was assessed using a dedicated validation set, yielding a precision of 96.83%, recall of 97.33%, mAP@50 of 98.67%, and mAP@50–95 of 90.83%, in addition to favourable class-specific F1-scores. These results, supported by detailed confusion matrices and PR/F1 curves, provided quantitative evidence of the model's capability to detect, localize, and classify traffic signs across diverse categories with exceptional accuracy. Thus, the system's evaluation phase fully satisfied the performance assessment criteria outlined in the objective.

Finally, the fourth objective was to apply the trained model to frame-by-frame detection

tasks on recorded video feeds, analysing system behaviour in realistic driving scenarios and assessing generalization capability in dynamic visual environments. This was achieved through the implementation of a video-based inference module that decomposed recorded driving scenes into individual frames and processed each using the YOLO11 detection engine. The model consistently rendered accurate bounding boxes and class labels, even under challenging conditions involving motion blur, occlusions, and non-uniform lighting. Output videos demonstrated high detection fidelity, validating the system's functionality in temporally disjointed, spatially dynamic contexts representative of real-world applications.

Chapter 7: Conclusion and Recommendation

7.1 Conclusion

This project commenced with the careful curation and refinement of a traffic sign dataset composed of 18 classes, derived from open-access sources and augmented using the Roboflow platform. The emphasis was placed on dataset fidelity, annotation precision, and visual variability, thus ensuring that the model would be trained under conditions reflective of actual road scenes. Preprocessing procedures including resizing, orientation correction, and normalization were coupled with augmentation techniques such as blur, exposure shifts, and rotational distortion to simulate realistic visual challenges while preserving the semantic core of each sign.

The core of the detection framework centred around the YOLO11 architecture, initialized with pretrained weights and fine-tuned over a structured training regime spanning 40 epochs. The training phase yielded a model capable of high-performance detection, achieving a precision of 96.83%, a recall of 97.33%, and a mean Average Precision (mAP@50) of 98.67%, with a generalization score (mAP@50–95) of 90.83%. These results, validated on an unseen test set, substantiate the model's ability to reliably detect and classify traffic signs even under visual variability and moderate occlusion. Furthermore, these metrics were complemented by a series of confusion matrices, PR and F1 curves, and bounding box overlays, all of which confirmed the operational integrity of the model.

The video-based inference module further demonstrated the functional feasibility of deploying the trained model in a frame-by-frame processing pipeline. Video sequences simulating real-world vehicular navigation were successfully analysed, with the YOLO11 model reliably identifying and labelling traffic signs in each frame. Output videos were generated with bounding boxes and class labels rendered on-screen, confirming both the spatial alignment and the semantic correctness of the model's predictions. These demonstrations serve as visual validation of the system's applicability in intelligent transportation ecosystems.

Throughout the implementation, several challenges were encountered—including limitations in dataset class balance, sensitivity to augmentation extremes, and constraints imposed by the cloud-based development environment. However, each of

these obstacles contributed to a deeper understanding of system parameters, guiding incremental refinements that ultimately led to a robust and scalable solution. The resulting system represents not only a successful technical artifact but also a blueprint for how deep learning techniques can be pragmatically applied to address one of the many perception problems in autonomous vehicle development.

In sum, the project has fulfilled all its core objectives, delivering a fully functional, quantitatively validated, and qualitatively demonstrated system for traffic sign detection from video. It offers practical insights into the synthesis of data preparation, model training, and deployment strategies within a constrained yet realistic research and development context. As such, it serves as both a proof of concept and a springboard for further innovations in autonomous visual perception systems.

7.2 Recommendation

While the system developed herein has demonstrated commendable levels of accuracy and operational robustness, several avenues for enhancement remain open and merit further exploration. These recommendations are offered not as remedies for failure, but as prospective optimizations to push the boundaries of what the system may achieve under more demanding, real-world conditions.

Firstly, the current system relies solely on frame-by-frame analysis, which, while effective, does not exploit temporal continuity—a key feature of video data. Future iterations could incorporate recurrent or transformer-based modules to enable temporal tracking, allowing the system to remember sign positions across multiple frames and enhance detection under occlusion or transient visual noise. This would be particularly beneficial for handling blurred or momentarily obstructed signs in fast-moving scenes.

Secondly, while the model was trained on an image dataset constructed from diverse sources, the geographic and regulatory scope of traffic signs remains limited to a controlled subset. Expanding the dataset to include region-specific variants (e.g., European triangular warnings, Japanese pictorial signs) and non-standard signs would enhance the system's generalizability and move it closer to real-world deployment readiness in global contexts. This expansion would also necessitate the integration of multilingual or symbol-sensitive classification layers capable of interpreting context-rich signage.

Another critical recommendation involves real-time deployment on embedded systems. While Google Colab offers an ideal environment for rapid prototyping, operational deployment in autonomous vehicles demands hardware-optimized models running on edge devices such as NVIDIA Jetson or Raspberry Pi with Coral TPU. Converting the YOLO11 model to TensorRT or ONNX format and benchmarking it under latency and power constraints would provide meaningful insights into the system's viability in real-time ADAS architectures.

In conclusion, while the project has reached a successful milestone, the broader journey of deploying robust, adaptive, and globally scalable traffic sign detection systems remains ongoing. The recommendations offered herein lay a roadmap for this continued advancement, reinforcing the role of computer vision as a cornerstone in the development of safer, smarter, and more autonomous vehicles.

References

[1] C. Liu, S. Li, F. Chang, and Y. Wang, 'Machine Vision Based Traffic Sign Detection Methods: Review, Analyses and Perspectives', IEEE Access, vol. 7, pp. 86578–86596, 2019.

[2] H. Li, F. Sun, L. Liu, and L. Wang, 'A novel traffic sign detection method via color segmentation and robust shape matching', Neurocomputing, vol. 169, pp. 77–88, Dec. 2015.

[3] E. Hamuda, B. Mc Ginley, M. Glavin, and E. Jones, 'Automatic crop detection under field conditions using the HSV colour space and morphological operations', Comput. Electron. Agric., vol. 133, pp. 97–107, Feb. 2017.

[4] F. Garcia-Lamont, J. Cervantes, A. López, and L. Rodriguez, 'Segmentation of images by color features: A survey', Neurocomputing, vol. 292, pp. 1–27, May 2018.

[5] X. R. Lim, C. P. Lee, K. M. Lim, T. S. Ong, A. Alqahtani, and M. Ali, 'Recent Advances in Traffic Sign Recognition: Approaches and Datasets', Sensors, vol. 23, no. 10, p. 4674–4690, May 2023.

[6] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, 'You Only Look Once: Unified, Real-Time Object Detection', in 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA: IEEE, vol. pp. 779–788, Jun. 2016.

[7] Y. Zhu and W. Q. Yan, 'Traffic sign recognition based on deep learning', Multimed. Tools Appl., vol. 81, no. 13, pp. 17779–17791, May 2022.

[8] A. Ellahyani, M. E. Ansari, and I. E. Jaafari, 'Traffic sign detection and recognition based on random forests', Appl. Soft Comput., vol. 46, pp. 805–815, Sep. 2016.

[9] Y. Saadna and A. Behloul, 'An overview of traffic sign detection and classification methods', Int. J. Multimed. Inf. Retr., vol. 6, no. 3, pp. 193–210, Sep. 2017

[10] F. Zaklouta and B. Stanciulescu, 'Real-time traffic sign recognition in three stages', Robot. Auton. Syst., vol. 62, no. 1, pp. 16–24, Jan. 2014.

[11] P. Ganesan, V. Rajini, and R. I. Rajkumar, 'Segmentation and edge detection of color images using CIELAB color space and edge detectors', in INTERACT-2010, vol , pp. 393–397, Dec. 2010.

[12] Y. Yuan, Z. Xiong, and Q. Wang, 'An Incremental Framework for Video-Based Traffic Sign Detection, Tracking, and Recognition', IEEE Trans. Intell. Transp. Syst., vol. 18, no. 7, pp. 1918–1929, Jul. 2017.

[13] S. B. Wali et al., 'Vision-Based Traffic Sign Detection and Recognition Systems: Current Trends and Challenges', Sensors, vol. 19, no. 9, Art. no. 9, Jan. 2019.

[14] H. Handoko, J. H. Pratama, and B. W. Yohanes, 'Traffic sign detection optimization using color and shape segmentation as pre-processing system', TELKOMNIKA Telecommun. Comput. Electron. Control, vol. 19, no. 1, p. 173, Feb. 2021.

APPENDIX A

POSTER

UTAR
UNIVERSITI TUNKU ABDUL RAHMAN

TRAFFIC SIGN DETECTION FROM VIDEO FOR AUTONOMOUS VEHICLES

Project Developer: Wong Song Wang
Project Supervisor: Prof. Dr Leung Kar Hang

01.

Introduction

A deep learning system using YOLO11 to accurately detect and classify traffic signs in video frames, enabling reliable perception for autonomous driving under diverse conditions.

02. **Proposed Method**

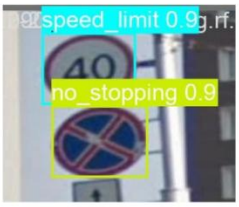
YOLO11 is trained on 18 traffic sign classes and applied frame-by-frame to video input for real-time detection and annotation.

03. **Result**

Dataset: 21,688 images, 18 classes
Performance:
Precision: 96.82%
Recall: 97.33%
mAP@50: 98.67%
mAP@50–95: 90.83%

04.

Conclusion



Key Achievements
Successfully implemented YOLO11 for high-accuracy traffic sign detection in video. Achieved strong performance across 18 classes, demonstrating robustness under varied visual conditions.

Challenges
Detection performance is limited for small, distant, or partially occluded signs. Class imbalance and limited dataset diversity introduced minor misclassification in rare sign categories.

Future Work
Enhance dataset diversity by including rare and region-specific signs. Optimize the model for real-time performance on embedded systems.

Impact
Establishes a scalable and accurate detection framework for autonomous driving. Contributes to safer, more reliable Advanced Driver Assistance Systems (ADAS).