sustainability

IMPACT FACTOR 3.3

CITESCORE 6.8

Article

Performance Evaluation of Lane Detection and Tracking Algorithm Based on Learning-Based Approach for Autonomous Vehicle

Swapnil Waykole, Nirajan Shiwakoti and Peter Stasinopoulos

Special Issue

The Potential Diffusion and Impacts of Autonomous Electric Vehicles: Evidence and Modelling Approaches

Edited by

Dr. Chengxiang Zhuge

MDPI

https://doi.org/10.3390/su141912100

sustainability

MDPI

Article

Performance Evaluation of Lane Detection and Tracking Algorithm Based on Learning-Based Approach for Autonomous Vehicle

Swapnil Waykole, Nirajan Shiwakoti * and Peter Stasinopoulos

School of Engineering, RMIT University, Melbourne, VIC 3000, Australia
* Correspondence: nirajan.shiwakoti@rmit.edu.au

Abstract: Disruptive technology, especially autonomous vehicles, is predicted to provide higher safety and reduce road traffic emissions. Lane detection and tracking are critical building blocks for developing autonomous or intelligent vehicles. This study presents a lane detecting algorithm for autonomous vehicles on different road pavements (structured and unstructured roads) to overcome challenges such as the low detection accuracy of lane detection and tracking. First, datasets for performance evaluation were created using an interpolation method. Second, a learning-based approach was used to create an algorithm using the steering angle, yaw angle, and sideslip angle as inputs for the adaptive controller. Finally, simulation tests for the lane recognition method were carried out by utilising a road driving video in Melbourne, Australia, and the BDD100K dataset created by the Berkeley DeepDrive Industrial Consortium. The mean detection accuracy ranges from 97% to 99%, and the detection time ranges from 20 to 22 ms under various road conditions with our proposed algorithm. This lane detection algorithm outperformed conventional techniques in terms of accuracy and processing time, as well as efficiency in lane detection and overcoming road interferences. The proposed algorithm will contribute to advancing the lane detection and tracking of intelligent-vehicle driving assistance and help further improve intelligent vehicle driving safety.

Keywords: MPC controller; lane detection and tracking; autonomous vehicle; learning-based-approach algorithm

Citation: Waykole, S.; Shiwakoti, N.; Stasinopoulos, P. Performance Evaluation of Lane Detection and Tracking Algorithm Based on Learning-Based Approach for Autonomous Vehicle. *Sustainability* **2022**, *14*, 12100. https://doi.org/10.3390/su141912100

Academic Editor: Chengxiang Zhuge

Received: 23 August 2022
Accepted: 21 September 2022
Published: 24 September 2022

1. Introduction

Traffic safety and transport emissions have significantly influenced city development due to the yearly increase in automobile ownership. To an extent, traffic crashes are caused by human factors associated with motorists, such as drunkenness, fatigue, and improper driving maneuvers. To some degree, automated vehicles can reduce these human errors [1,2]. The development of automated vehicles has increasingly sparked the interest of researchers in relevant sectors around the globe. Smart vehicles are able to assist people in performing driving activities based on current traffic information, emphasizing the significance of such algorithms in increasing automobile driving safety and releasing people from the boredom of certain driving situations [3,4].

The US Department of Transportation's National Traffic Safety Administration (NHTSA) defines five different autonomous driving levels shown in Figure 1:

Level 0: No automation; the vehicles are controlled by a human.

Level 1: Except for a few functions such as steering and acceleration, most of the activities are performed by a human.

Level 2: Refers to partial driving automation, in which specific features such as cruise control and lane-centring are automated. The driver in the vehicle can take control of the system at any instance in time.

Level 3: The vehicle makes decisions based on the conditions in which it is operating. However, the human driver remains present in the vehicle; when needed, the

human controls the system. The main advantage of this system is the avoidance of continuous monitoring.

Level 4: This is an autonomous stage in which the vehicle can perform safety-critical tasks and monitor the environment of its operation. This autonomous level is suitable for a few scenarios.

Level 5: This is a fully autonomous stage in which the vehicle performs all functions autonomously without a human driver in every driving scenario.

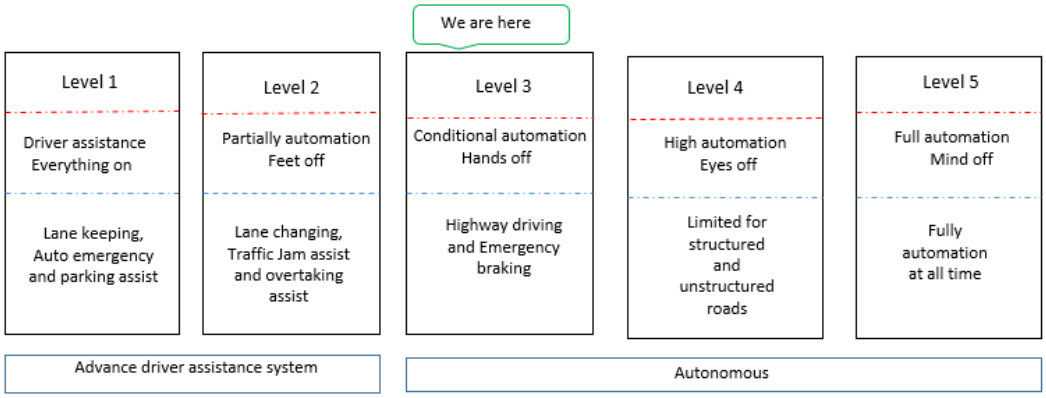

Figure 1. Different levels of automation.

Lane detection is a crucial foundation in the development of intelligent vehicles that has a direct effect on the functioning of driving behaviours. Based on the driving lane, a smart car can determine an optimal driving direction and provide an exact location of the vehicle in the lane. Due to this, it enhances the efficiency and safety of automated vehicles [5,6]. As a result, detailed investigation is required.

Currently, lane detection mostly relies on visual sensors. Visual sensors, which collect road surroundings in front of cars using cameras, have effectively become the eyes of smart cars. Such sensors can constantly operate for extended periods of time while being very adaptable. Simultaneously, a broad-spectrum is available for visual sensors and can detect invisible rays to the eye. Such property significantly expands human vision space [7–10]. However, in everyday natural circumstances, obstacles such as vehicle obstruction, inadequate light, various road twists and curves, and diverse surroundings on both sides have made reliable lane detection challenging [11,12]. Due to the quick development of computers' computational speed, many researchers have concentrated on lane detection. Such lane detection can be classified as either feature- and model-based lane detection or learning-based lane detection. The feature-based lane recognition algorithm identifies the lane around the surrounding road scene considering the lane's edge and colour characteristics. Zheng et al. [13] adopted an adaptive double threshold approach to extract the relevant characteristics of the lane in the road environment after converting the original RGB image to (Commission Internationale de l'Elcairage) CIE colour space. Haselhoff et al. [14] developed a two-dimensional linear filter technique for detecting both left and right lanes that can remove interference noise during the detection stage while retaining the intrinsic properties of the lane as much as possible in the distant view. Son et al. [15] investigated lane detection using light fluctuation effects. Yellow and white lanes were chosen as candidate lanes based on their lighting invariance. By using the clustering approach from parent lanes, the lanes were detected. To forecast the fading point of the lane, Amini et al. [16] adopted the Gabor filter. The lane boundary was the line formed below the fading point when the trust function was at its greatest confidence level.

Some researchers have concentrated on unstructured road lane detection and kerb detection. Using a vanishing point approach, Kong et al. [17] performed reverse-detected road boundaries and produced excellent results in lane detection for the unstructured road. Hervieu et al. [18] detected road boundaries by using the Kalman filter model. It was further shown that this approach successfully detected road boundaries with obstructions.

Road boundary markings were established by Hata et al. [19] using properties and changes in elevation.

Model-based lane detection considers the lane as a matching geometric prototype and then accommodates the lane after obtaining the model parameters. Geiger et al. [20] utilised a Bayesian classifier to generate a probabilistic model based on road surfaces' pixel locations and presented a probability function integrated with vehicle path and fading point features. The comparison divergence parameters were used to develop lane feature recognition. Liang et al. [21] developed an adaptive model to examine the relationship between time and road visual ahead. In this way, the lane geometry could be recognized using the model. Bosaghzadeh et al. [22] used a principal component analysis (PCA) approach to address the complexity of determining the front image's view angle. Lane detection was performed using the rotation matrix model, and its robustness was excellent.

Learning-based approaches have also recently been applied to lane identification. He et al. [23] introduced a lane detection technique using a convolution neural network that transformed the detection picture into an aerial image. Although the accuracy for detection was excellent, it had a low speed for processing and therefore was slow to perform. SegNet, a classification network that considered pixels, was proposed by Badrinarayanan et al. [24]. It can detect lane lines through scene segmentation and categorise and distinguish people, plants, and buildings. On the other hand, the classification network has a complex structure, high computational burden, and inadequate real-time performance. Pan et al. [25] proposed a novel neural network model based on the graphical geometry band that successfully handled detection issues when the lane was obstructed. These findings reveal that several research approaches can enhance the effective detection accuracy of lane detection. However, these algorithms can be limited by different situations [26,27]. The feature-based approach for lane detection is relevant to real-world road landscapes with defined boundaries for lanes and uncomplicated road conditions. Detection accuracy was dramatically lowered when the lane was broken or had inadequate visibility. The accuracy of road boundary detection was good for unstructured road detection and kerb detection. However, when the road information was too complex, or there was interference from obstacles, the computational speed was greatly lowered, which easily led to erroneous detection [28–31]. Model-based lane detection was only appropriate for circumstances in which the current model was compatible with the detection images. Additionally, this approach was found to entail high complexity and a substantial amount of work. Real-time performance was also lower in realistic road scenarios [32–34]. Learning-based approaches have the potential to significantly increase the accuracy and robustness of lane detection. However, relevant algorithms require more technology, and the training model structures are too complex. Thus, there are still some limitations [35–38]. As a result, more improvements to the lane detecting system are required.

Model Predictive Control (MPC) has been used in the literature to control the vehicle dynamic for lane detection and tracking [39]. MPC is a sophisticated control approach that has been utilised for process control since the 1980s. As microprocessor processing power has increased, the usage of MPC has moved to real-time embedded applications, which are often utilised in the automobile and aerospace sectors. MPC can handle multi-input multi-output (MIMO) systems with linked input–output channels, which simplifies control loop construction. It would be difficult to design a controller for such a system using PID (proportional–integral–derivative) controllers, for example, since the control loops and related responses would be entangled. Furthermore, MPC can manage limitations on inputs, outputs, and states, which is crucial since real-world systems must adhere to physical limits. MPC requires several design decisions that affect the complexity of the underlying optimisation issue and, as a result, how quickly the controller can execute. Choosing the proper MPC type, structure, and model complexity depending on plant dynamics and control goals, for example, or selecting the suitable MPC settings based on hardware/software constraints, are common problems [40].

Objective and Scope of the Study

Much research is being conducted in the autonomous vehicle area to develop and produce an enhanced Advanced Driver Assistance System (ADAS) capable of providing safety to drivers. Lane detection and tracking has been a crucial component of ADAS for safe driving maneuvers and avoiding accidents. The literature review shows that there are limited studies on lane detection and tracking on unstructured roads because of the complexity of the unstructured roads in suburban areas and the unavailability of datasets. Some techniques, such as feature-based and model-based models, have previously been used for lane detection and tracking on straight structured roads, but the learning-based approach has yet to be integrated with MPC for structured and unstructured roads. Therefore, to address this knowledge gap, this study develops and evaluates the performance of a lane detection and tracking algorithm based on a learning-based approach integrated with MPC for structured and unstructured roads in low (<30 km/h), medium (30–50 km/h), and moderate speed environments (51–60 km/h).

The proposed algorithm uses the steering angle, yaw angle, and sideslip angle as inputs for the adaptive controller. MPC was utilised for planning difficulties in the predictive nature of the optimisation and the internal plant model utilised in the process. Instead of directly employing the optimisation problem solution for low-level control, MPC was applied to the internal plant model to provide reference output trajectories for another controller to track. Finally, simulation tests for the lane recognition method were carried out using a road driving video in Melbourne, Australia, under various road geometries and the BDD100K dataset by the Berkeley DeepDrive Industrial Consortium. Datasets for performance evaluation were created using an interpolation method. The performance of the proposed algorithm was evaluated by comparing experimental results with other algorithms.

The structure of the paper is organized as follows. Section 2 presents MPC's design for vehicle dynamics and control parameters, followed by Section 3, which explains and executes the proposed lane detection approach. Simulation tests based on obtained datasets are carried out along with algorithm performance comparisons. Finally, Section 4 describes the conclusions and recommends future work.

2. Design of the Model Predictive Controller (MPC)

In the next few sub-sections, the design of the MPC that includes an explanation of the basic principle of the MPC, the vehicle dynamic model, control parameters, plant model and prediction model, along with the implementation of the model in Simulink are explained.

2.1. Basic Principle of an MPC

An MPC algorithm is a process control method founded on a predictive model. The best control sequence at the present time is determined by solving the open loop optimisation problem in a finite horizon timeline based on the system state value (or an estimated value) acquired at each sample time. The first element is applied to the controlled body, and the procedure described above is repeated at the following sample time. The best value is progressively acquired as the horizon recedes ahead. Prediction, objective function, and control law are the components of the model predictive controller. The predictive model predicts the process's outcome by outlining the connection between the outputs and the inputs. The objective function determines the control rule by using various cost functions. The control rule's objective is to frame the output based on the reference signal, and the control signal provides the factors to be considered by the controller rule. The future output is determined by replacing the cost function in the selected model with a combination of input, output, and the future control signal. The model's optimised output is used to design the optimum solution. MPC parameters should be properly created and modified to provide good performance without producing unpleasant feelings. The MPC parameter contains the prediction horizon, the control horizon, and the performance index weights.

In this part, we demonstrate the parameter design and tuning method that was constructed from multiple closed-loop simulations and experiments [40].

2.2. Vehicle Dynamic Model

Different dynamic models can describe the vehicle's different dynamic responses. It is critical to choose an optimal vehicle model for vehicle motion control. Because only the lateral plane motion of vehicles is addressed, the additional motion of the vehicle's internal system may be neglected. As a result, the two-degree-of-freedom (2-DOF) vehicle-dynamics model, which combines lateral motion along the y-axis and yaw around the z-axis as input, was selected as the study object. The output is predicted by MPC based on the input. The 2-DOF vehicle-dynamics model is shown in Figure 2. The 2-DOF vehicle dynamics model is chosen as the research object, which includes lateral motion along the y-axis and yaw angle around the z-axis as input. Because the vehicle is a complex multiple-degree-of-freedom nonlinear time-varying system, it is extremely difficult to establish a highly accurate dynamic model that fully reflects all aspects of the vehicle's characteristics, which undoubtedly increases the system's complexity and computational burden and is incompatible with real-time vehicle control. As a result, in order to achieve adequate control accuracy and real-time performance, a sufficiently streamlined dynamic model must be established to further minimize calculation efforts while ensuring high precision. This paper focuses on the trajectory tracking control algorithm (lane detection and tracking) for autonomous vehicles while taking cornering characteristics into consideration so that the vehicle can still track the desired path quickly and smoothly even in challenging conditions, especially for the vehicle's longitudinal, lateral, and yaw movement.

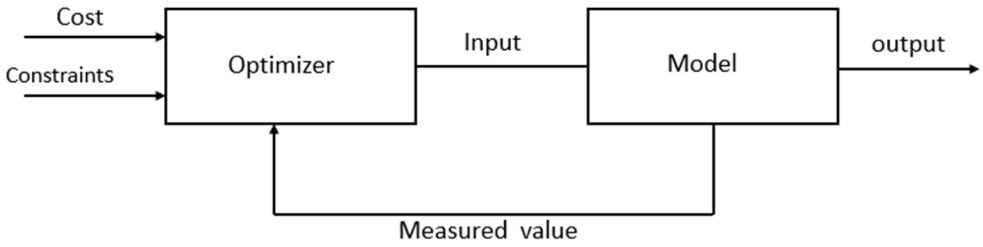

Figure 2. Structure of the MPC algorithm.

Objective function:

The purpose of autonomous vehicle lateral control is to track the proposed path in real-time while also ensuring stability and comfort. The controller adjusts the steering angle such that the vehicle follows the reference path. The controller minimizes the distance between the current vehicle position and the reference path. Furthermore, this controller tracks or predicts future reference points to keep the vehicle on the desired path. The objective function is the component of the MPC algorithm that directly represents the control objective. As a result, the objective function should be consistent with the control objective. The objective function J in this work was chosen as follows:

$$J = \left[Y_{out}(k) - Y_{ref}(k)\right]^T \cdot Q \cdot \left[Y_{out}(k) - Y_{ref}(k)\right] + \Delta U(k)^T \cdot R \cdot \Delta U(k) + \rho\varepsilon^2 \Big] \quad (1)$$

where Y_{out} represents the predicted value of the output sequence, Y_{ref} refers to the reference value of the output sequence, ε defines the relaxation factor, R and Q indicate the weight matrices, and ρ is the weight coefficient. The first term of the objective function represents the error between the output and reference, which reflects the control objective's precise tracking; the second term represents the magnitude of the control increment, which reflects the control objective's stability and comfort; the third term is relation factors, which assure that the optimisation objective has a quadratic programming solution that is feasible.

Constraints:

Control constraints, control increment constraints, and output constraints are the three main types of constraints encountered in a general MPC controller. The three above-mentioned limitations are imposed as follows:

Control constraints:

$$V_{min}(k) \leq V(k) \leq V_{max}(k). \tag{2}$$

Increment constraints:

$$\Delta V_{min}(k) \leq \Delta V(k) \leq \Delta V_{max}(k). \tag{3}$$

Output constraints:

$$V_{out\ min}(k) \leq V_{out}(k) \leq V_{out\ max}(k). \tag{4}$$

The road adhesion constraint, which has a significant impact on the vehicle's dynamic performance, is also taken into account here. Figure 3 shows the vehicle dynamic model. If the friction coefficient of the road is too low, the road cannot achieve sufficient driving force, resulting in a slip. As a result, the following limitation must be introduced to the vehicle's acceleration:

$$\sqrt{l_x^2 + l_y^2} \leq \mu g \tag{5}$$

where l_x represents the longitudinal acceleration, l_y is the lateral acceleration μ indicates the road friction coefficient, and g provides the gravity acceleration. This study only considers the lateral dynamics of the vehicle, and the longitudinal velocity is assumed to be constant.

$$l_y \leq \mu g. \tag{6}$$

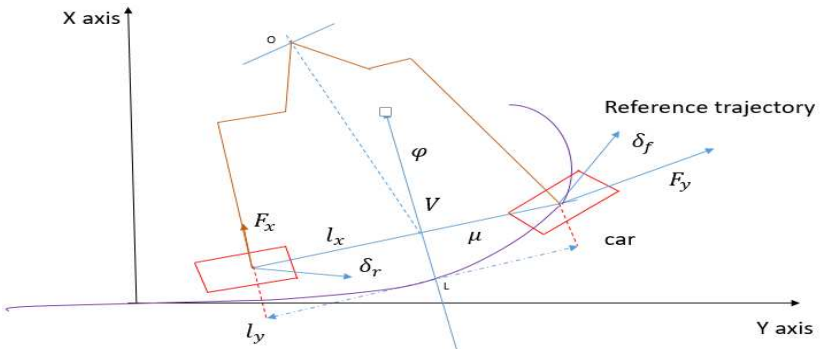

Figure 3. Two degrees of freedom vehicle dynamic model.

Lateral acceleration y must be changed since it is not a state variable:

$$\varphi V = l_y \tag{7}$$

where' φ represents the yaw rate, V denotes the longitudinal velocity, and lateral acceleration constraints may be changed into yaw rate constraints:

$$\varphi \leq \frac{\mu g}{V} \tag{8}$$

2.3. Control Parameters

The MPC parameters include prediction horizon, control horizon, sampling time weight matrix, and constraint range. The prediction horizon, control horizon, and sample time are the three control factors that have the largest influence on control effects. As a result, the focus of our study is primarily on these three control variables. The control impact of the same control parameters varies greatly depending on the operating circumstances. Under diverse and changing working parameters, it is difficult to convert a collection of

control parameters to all working scenarios. As a result, the control parameters must be adjusted in response to changing operating circumstances. This study is concerned with the selection of control parameters for various (changing) vehicle speeds. Vehicles will be influenced by a predicted trajectory that is too far away if the prediction horizon, control horizon, and sample time are set too wide. Furthermore, processing demands will rise, thereby reducing motion tracking accuracy. If the prediction horizon, control horizon, and sample time are too short, vehicles may have trouble reacting to unexpected changes in the trajectory, resulting in vehicle stability issues. Following several tests, it was determined that the greater the speed is, the greater the prediction horizon and sample duration should be. Based on the above conclusion, three types of control parameters are created for low, medium, and high vehicle speeds, respectively. Table 1 presents the relationship between the three categories of control parameters and vehicle speeds. The road-curvature path is used to illustrate the vehicle path tracking accuracy and stability under various situations. On the other hand, simulations of changing circumstances include modifying the friction coefficient of the road and changing the speed to assess the algorithm's capacity to adapt to changing working conditions.

Table 1. Relationship between vehicle speed and control parameters.

Control Parameters	Vehicle Speed (km/h)
$N_p = 1, N_c = 1$	<30 (low)
$N_p = 4,\ N_c = 4$	30–50 (medium)
$N_p = 8,\ N_c = 8$	51–60 (moderate)

2.4. Prediction T_s

The internal prediction model's sample time is represented by the *prediction Ts*, which specifies how long each prediction step seems to last. The values of all manipulated variables (MVs) stay constant throughout. The maximum limit of feasible control bandwidth is determined conceptually by the predicted sample time. The controller's prediction time is the product of the prediction horizon and the prediction sample time—that is, how far into the future the controller is planning. Because the prediction time is often offered as part of the control objective, the value of the prediction sample time is frequently chosen in conjunction with the prediction horizon.

Plant dynamics and control response time affect the upper limit of prediction Ts. If prediction *Ts* is slow, for example, one may not have enough control bandwidth to stabilise an open-loop unstable plant. A faster prediction Ts, on the other hand, will require a larger prediction horizon to maintain a constant prediction time. Longer prediction horizons, however, result in more decision variables and constraints, making the optimisation problem more time-consuming (higher memory footprint) and difficult to solve, as stated in the Prediction Horizon section. As a general guide, one should aim to accommodate 10–20 MV (manipulated variable) motions inside the rise time of the open-loop step response.

2.5. Control Ts

The sample time of the MPC is controlled by the control Ts, which specifies how often the MPC optimisation problem is addressed during runtime. The control sample time is usually equal to the prediction sample time; however, it may be changed to be fast (but not slower). In general, a faster control Ts enhances performance (i.e., bandwidth) and resilience (i.e., gain and phase margins) to some degree. Furthermore, when the control Ts decreases, rejections of unknown disturbances, such as differences between the internal MPC model and real plants, often improve and eventually plateau. Qualitatively, this makes sense since the controller can respond to changes in the environment quickly. The control sample time value at which the performance plateaus is generally determined by the dynamic features of the plant. Small control sample times, for example, will not help processes with moderate dynamics as much as real-time control applications such as motor

control. However, when the control sample duration decreases, so does the computing effort as the MPC optimisation issue is performed with greater frequency. As a result, the best option is an exchange between performance and computational effort.

To establish the control sample time, first a less aggressive controller that delivers acceptable results (big control Ts, possibly equal to prediction Ts) is applied. Next, the control Ts is reduced while observing the controller's execution time. When working on a real-time application, the control Ts can be reduced even further, provided the optimisation is completed safely within each sample interval under typical plant operating conditions. As discussed in further detail in the next sections, there is no way to estimate how many solver iterations are necessary to achieve an optimum solution. Therefore, adopting a suboptimal solution for the optimisation problem simplifies tuning the control Ts. Notably, the lowest control Ts that an MPC can obtain is system-dependent and will vary across (real-time) hardware and simulation platforms.

2.6. Prediction Model

The sample time may also be found by considering the transformation correlation between the vehicle coordinates and the inertial coordinates as

$$X = x \sin \varphi + \cos \varphi \tag{9}$$

The 2-DOF vehicle dynamic equation is obtained as

$$\begin{cases} m(y + x\varphi) = 2\left[\delta - \frac{(y+a\varphi)}{x}\right] \\ I_z = 2a\left[\delta - \frac{(y+a\varphi)}{x}\right] \end{cases}$$

Combining the transformation correlation and the 2DOF vehicle dynamic equation, the following equation is obtained:

$$\begin{cases} m(y + x\varphi) = 2\left[\delta - \frac{(y+a\varphi)}{x}\right] \\ I_z\varphi = 2a\left[\delta - \frac{(y+a\varphi)}{x}\right] \\ X = x \sin \varphi + y \cos \varphi \end{cases}$$

The vehicle dynamics model used in this research work is a nonlinear model, and therefore state space form is created through a linear transformation. The following state space variables are selected to design the MPC. Because the MPC algorithm is only suitable for discrete system control, the above continuous system equation must be changed into a discrete system:

$$\begin{cases} x(k+1) = Ax(k) + Bu(k) \\ y(k) = Cx(k) \end{cases}$$

Here, $y(k)$, $x(k)$ and $u(k)$ represents their values in time k.

$$A = \begin{bmatrix} C_{4\times4} & D_{4\times4} \\ 0 & I_{1\times1} \end{bmatrix};$$

$$B = \begin{bmatrix} C_{4\times1} \\ I_{1\times1} \end{bmatrix}$$

$$C = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$

$$N_c = N_p = 1$$

On this basis, we assume

$$x(k) = \begin{bmatrix} x(k) \\ u(k-1) \end{bmatrix}$$

A state space expression is obtained as:

$$\begin{cases} x(k+1) = Ax(k) + B\Delta u(k) \\ y(k) = Cx(k) \end{cases}$$

Considering N_p as the prediction horizon and Nc as the control horizon, the system's future output is:

$$Y_{out}(k) = x(k) + \theta\Delta U(k) \tag{10}$$

Through the inertial coordinate and vehicle coordinate transformation relationship, it can be found using the following coordinates:

$$Y_{out}(k) = \begin{bmatrix} y(k+1)k \\ y(k+2)k \\ y(k+N_n)k \end{bmatrix}, \theta\Delta U = \begin{bmatrix} BA \\ BA^2 \\ BA^{N_n} \end{bmatrix}$$

$$\Delta U(k) = \begin{bmatrix} \Delta\mu(k) \\ \Delta\mu(k+1) \\ \Delta\mu(k+N)_n \end{bmatrix}$$

Sample time: We can determine the frequency at which the controller performs the control algorithm by selecting the sample time. If the value is too large, the controller will not be able to respond quickly enough to the turns. Figure 4 below shows the impact of selecting a higher value. On the other hand, if the sample time is too short, the controller can respond to disturbances and set point changes considerably sooner, but this will generate an excessive computing load.

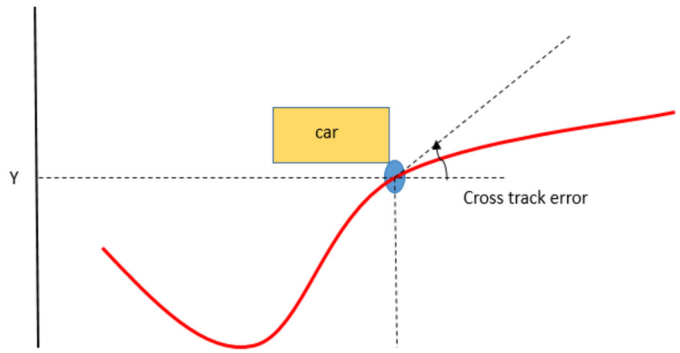

Figure 4. Illustration of the cross track error.

Prediction horizon: The length of expected future steps determines how far into the future the controller can predict. Observations can determine if the horizon is too short. In the present model, the car must decelerate as it approaches the turn. However, because of the shorter prediction horizon, the controller is unable to estimate the velocity, resulting in delayed braking, which causes the vehicle path to deviate from the reference trajectory.

Control horizon: Each control move in the control horizon can be considered as a free variable that the optimiser must evaluate. As a result, the fewer the calculations, the smaller the control horizon. However, the optimiser may not always provide the required manoeuvrability. Increasing the control horizon, on the other hand, will enhance the outcome but likely increase the complexity. Setting the control horizon to 10 to 20% of the prediction horizon is a good strategy.

The weights are the other key characteristics that must be correctly adjusted. One must specify the weights for velocity tracking, lateral error, longitudinal acceleration, and steering angle in the block. A good principle is to determine what is desired from the controller. In the present model, for example, our goal is to direct the vehicle to follow the track. As a result, we chose a larger weight for the lateral error and steering angle change. If the priority is changed to follow the reference velocity, the weights associated

with longitudinal motion must be increased. However, the reference velocity map must take into account the path's steep curves. When the tyre is in a low slip limit, the adhesion ratio is usually proportionate to the slip. This relationship holds true for roads with varying friction coefficients.

2.7. State Space Model for Adaptive Cruise Control System

Two state variables are established to develop a state space model for the (adaptive cruise control) ACC system design: inter-vehicle distance following error $\Delta d = d - dr$ and velocity following error $\Delta v = v_p - v_h$. The dr is calculated using the constant time headway policy provided by

$$dr = T_{hw} + d_0 \tag{11}$$

where T_{hw} represents the constant time headway, and d_0 denotes the stopping distance within the safety margin. We can describe the plant's state variables as $x = [x1\ x2\ x_3^T]^T$ and $\in R^2 + nf$ is the respective state, where $x1 = \Delta d$, $x2 = \Delta v$, and $x3 = xf$ are the input and output of the system. The state-space model is, therefore, written as

$$\begin{cases} \dot{x} = (x, u) + G_v + Hw \\ y = Cx + J \end{cases}$$

where $x \in R^2$.

The following variables are selected for the state space variables for the plant dynamics model:

$$A = \begin{bmatrix} 0 & 1 & -T_h \\ 0 & 0 & -1 \\ 0 & 0 & A(t) \end{bmatrix}$$

and

$$B = \begin{bmatrix} 0 \\ 0 \\ B(t) \end{bmatrix}$$

$$v = r_{travel}$$
$$r_{travel} = r_{accel} \times v_h + \dot{r}_{travel}$$

where the first component is known as the acceleration resistance, while the second component is known as other resistance:

$$\dot{r}_{travel} = r_{air} \times v_h^2 + r_{roll} + r_{grad}(\theta)$$

Let us define a state transformation for x_3:

$$\dot{x}_3 = \frac{1}{1 + \frac{r_{accel.}}{m}} x_3$$

In this case, the travel resistance may be expressed using new state $\dot{x}_3$ rather than v_h. The vehicle's acceleration is determined by the vehicle dynamic:

$$v_h = x_3 - \tfrac{1}{m} r_{travel}$$
$$= \left(1 + \frac{r_{acceleartion}}{m}\right) x_3 - \frac{1}{m} r_{travel}$$

The travel distance r_{travel} influences plant dynamics and may lead to prediction errors. The acceleration resistance, in particular, affects transient reactions, i.e., stop and go actions. To avoid deterioration in control performance as a result of such disturbances, it is vital to include all perturbance features in the MPC design explicitly. The model that considers acceleration resistance is linearly related to the host vehicle's acceleration. The disturbance characterization is included in the prediction model via state transformation. This simplistic

strategy may reject the disturbance impact while reducing computational load. The major components of the MPC are shown in Figure 5, which includes the forecasting model, receding horizon optimisation, and response correction. The predictive model considers the selection of an appropriate mathematical model that integrates present information and control inputs to anticipate future outcomes. This assists in explaining the dynamic behaviour of the controlled car. Receding horizon optimisation aims to explain the system uncertainty and the state deviation with a receding advance of period depending on the present conditions, ultimately bringing the real control as close to the ideal control as possible. Feedback correction provides appropriate feedback on the discrepancy between the expected and real values to assist optimisation in the future. Figure 6 illustrates that the plant model used as the basis for the adaptive MPC must be a discrete-time, state-space model. The adaptive MPC changes the plant model and nominal parameters at each control interval. The model and parameters stay unchanged across the prediction horizon once updated.

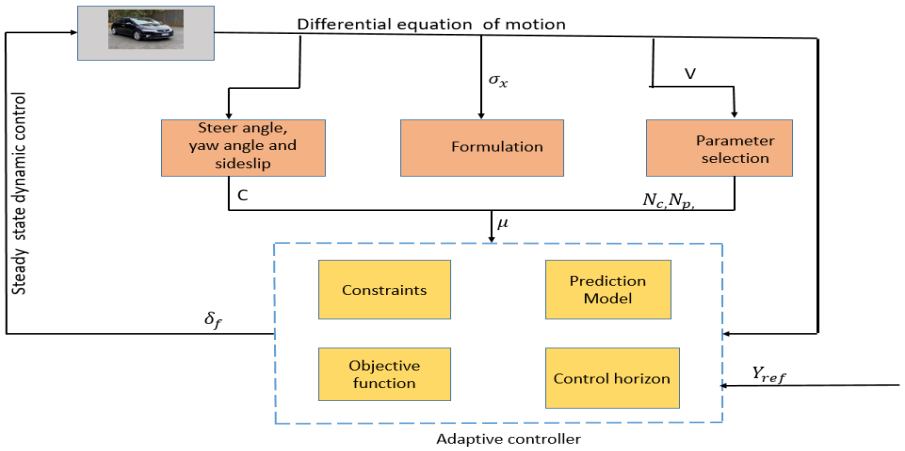

Figure 5. Shows the structure of the controller.

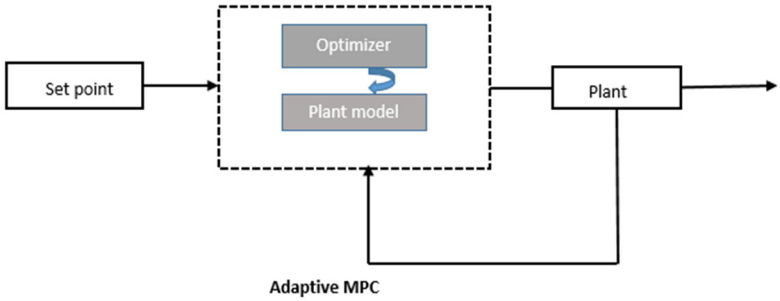

Figure 6. Illustration of the plant model.

2.8. Building Model and Implementation of an Adaptive Controller

The MPC needs a plant model to anticipate the plant's future behaviour over a specified time horizon. As a prediction model, we may utilise the plant model with the unmeasurable disturbance component removed. However, in order to decrease the computational cost of the microprocessor, a simple prediction model with lower dimensionality is urgently desired. The travel disturbance r_{travel} has an effect on plant dynamics and may lead to prediction errors. The acceleration resistance, in particular, influences the transient response, i.e., the stop-and-go maneuver. To avoid loss in control performance as a result of such perturbance, it is critical to address the perturbance features in MPC design explicitly.

Bear in mind that the MPC makes predictions about the system's future behaviour using a model of the system. The algorithm applies an online optimisation method to determine the best control action to transmit the expected output to the reference. As a

result, it is important to select and perfect the parameters in the "Path Following Control System" block to obtain the desired result.

Vehicles will be impacted by the predicted trajectory that is too far away if the prediction horizon, control horizon, and sample time are set too big. Furthermore, the processing burden will rise, reducing path tracking accuracy. If the prediction horizon, control horizon, and sample time are all set too low, the vehicle will have difficulties reacting to unexpected changes in the course, resulting in vehicle instability. Following thorough research, the conclusion is that the greater the speed, the bigger the prediction horizon and sample duration should be. Based on the above discussion, three sets of control parameters are created for low, medium and moderate vehicle speeds, respectively. Table 1 depicts the correlation between the three categories of control settings and vehicle speeds.

3. Lane Detection

Lane detection is vulnerable to surrounding environmental variables (e.g., low light, road tortuosity, and obstacles under various road conditions). Previous image processing relied on one detection picture. In addition, reliable lane detection must be used in dynamic driving environments ensuring a recognition rate with sufficiently superior accuracy. In this study, interpolation operations of the lane were carried out first. Then, on the basis of the model predictive control principle, the learning-based algorithm was deployed to obtain precise optimisation and lanes' curvature calculations. Lastly, a simulation test experiment was conducted using driving videos and the BDD100K datasets. Finally, an analysis of experimental results was conducted, along with comparing the performance of the proposed approach with other algorithms.

3.1. Simulation Test Experiment

The simulation test environment included a Windows 10 64-bit operating system, MATLAB, and the Driving Scenario Designer app. The hardware environment included an Intel (R) Core (TM) i5-6000 CPU@3.20 GHz processor, 16.00 GB memory, and a 2 TB hard disk. Model Predictive Control Toolbox™ in MATLAB, which provides functions, an app, and Simulink® blocks to develop the MPC, was used for this study.

The basic features of a path-tracking system employing the MPC to estimate the front wheel steering angle and the vehicle controller consist of three parts: the reference generation, the MPC optimiser, and the vehicle model. The reference generation module computes the reference yaw rate as well as the projected vehicle condition in advance. The lateral acceleration determined by the MPC at each sample time of each prediction horizon will affect the anticipated vehicle state in the following sampling period. Then, in the next sampling period, the altered state variables will create a new optimum preview lateral acceleration, and therefore the reference yaw rate is determined by the repeatable process of obtaining the ideal preview lateral acceleration and anticipating vehicle states. In other words, in general MPC, the state estimation is completed at one time, but once preview is applied, the lateral vehicle speed and acceleration will be changed at the next sample time, making rolling computation essential for reducing accumulating error and obtaining a more accurate reference yaw rate.

The reference comprises two modules: reference lateral displacement and reference yaw rate. The reference lateral displacement is calculated from the reference route in the predictive horizon, whereas the reference yaw rate is calculated by superimposing the intended trajectory in the preview distance on the predictive horizon. In comparison to the conventional MPC, this approach increases the length of the effective reference route without adding additional computation effort. To follow the reference route, the MPC optimiser with input and output constraints computes the steering wheel angle. The designed image processing and lane detection method offer inputs to the MPC. The centreline of the automobile is utilised to calculate the offset of the car location from this centreline as well as the yaw angle. The MPC uses this information to keep the automobile on the intended course on unstructured roadways. Figure 7 depicts the algorithm's working procedure.

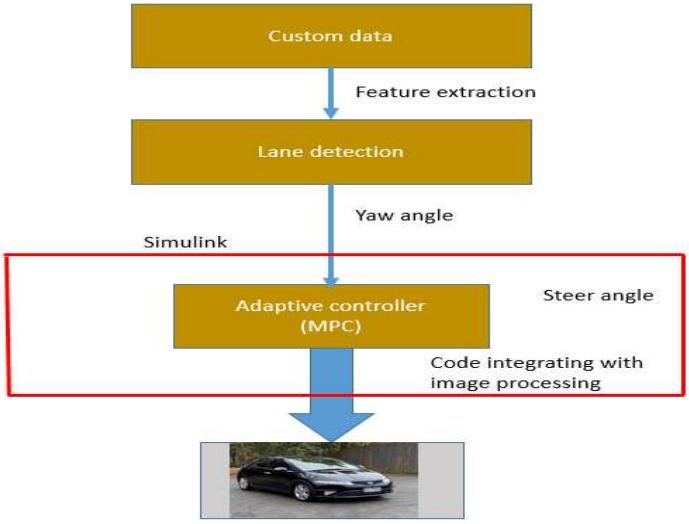

Figure 7. Overview of the major steps involved in executing the algorithm.

3.2. Lane Detection Based on Road Driving Videos

Road driving videos were used for the simulation test experiment in this study to evaluate the lane identification algorithm under challenging road conditions. In the experiment, such videos were taken through a real-time vehicle-mounted camera, and lane lines were recognised for various challenging road conditions (e.g., highways, hilly roads, and structured and unstructured roads).

Additionally, the interpolation approach was used to generate testing datasets. This approach was utilised to dynamically analyse the lane detection algorithm in the simulation test experiment under different scenarios. Firstly, the interpolation approach was designed for evaluation models based on hardware and software. We modified the time manual approach in several ways to generate a dataset that can simulate scenarios under real traffic conditions. This proposed approach is called interpolation. It is difficult to estimate lane boundaries in the gaps. To address this challenge, we selected user-annotated points to specify the maximum number of rows close to each other. Then, we applied cubic interpolation to these annotated points to obtain smooth curves because lane markings are not straight everywhere. In this way, the controller can predict the lane boundaries based on optimisation, and we can generate datasets for lane detection and apply the tracking algorithm to arbitrary scenarios with different traffic-flow conditions.

The linear interpolant is the straight line connecting the two known positions. The equation of slopes gives the value of y along the straight line for x in the interval:

$$\frac{y^* - y_1^*}{y_2^* - y_1^*} = \frac{x^* - x_1^*}{x_2^* - x_1^*} \tag{12}$$

y^* = linear interpolation value.
x^* = independent variable.
x_1^* & x_2^* = values of the function at one point.
y_1^* & y_2^* = values of the function at another point.

Figures 8–10 illustrate the lane line detection results under the structured road (road with lane markings) and unstructured road (road without lane markings) conditions.

Figure 8 illustrates the test results of a typical detected frame from a driving video of a straight road. The figure illustrates that the proposed algorithm can identify the left and right lane lines despite the vehicle's moderate speed on the roadway (51–60 km/h). The curve degree of the lane was the shortest compared to other road circumstances, the curvature radius was the highest, and the related sliding window offset was the smallest. Figure 9 presents the test result of a typical detected frame in an unstructured road driving video. The figure illustrates that, despite the poor illumination conditions (shadows)

on the unstructured roads, the proposed algorithm was able to detect the left and right lane markings. When compared to roadway circumstances, the lane line's curve degree was more considerable, the radius of the curvature was lower, and the associated sliding window offset was greater. Figure 10 shows the test results of a typical detected frame in the driving video under different weather conditions. The figure shows that the algorithm could detect the left and right lane markings despite the road being convoluted and variable, with numerous vegetation on both sides. Compared to other road conditions, the lane line's curve degree was the greatest, the curvature radius was the shortest, and the related sliding window offset was the greatest. A summary of the performance metrics and equations used to validate the detection rate of the algorithm is provided in Tables 2 and 3.

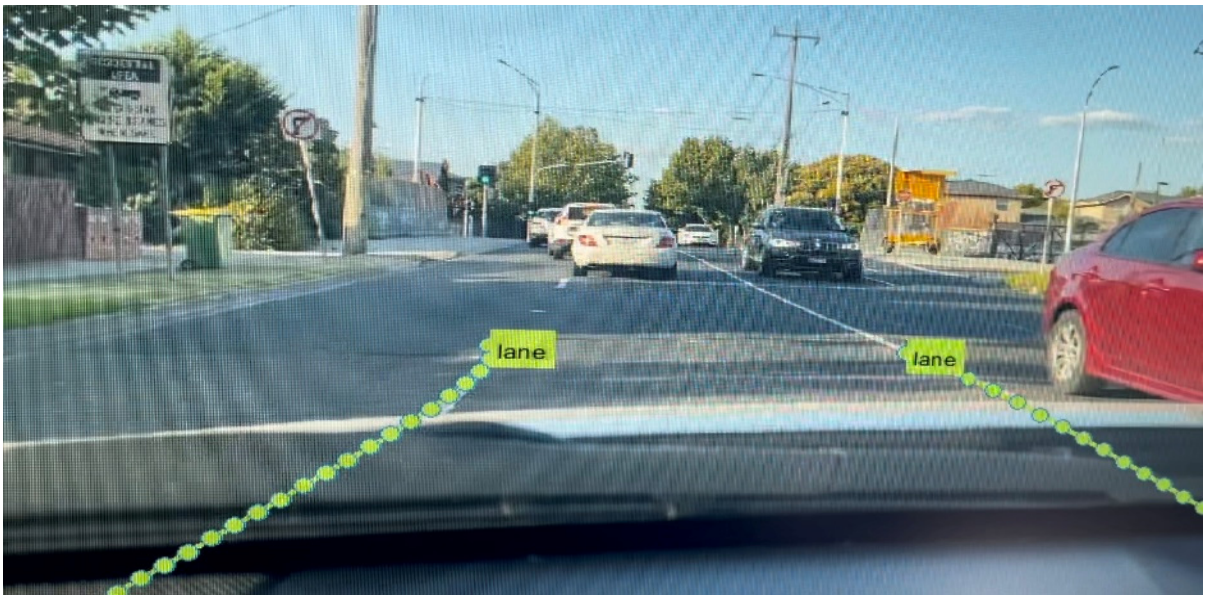

Figure 8. Lane detection in a highway driving video on a straight road.

Figure 9. Lane detection for unstructured road driving.

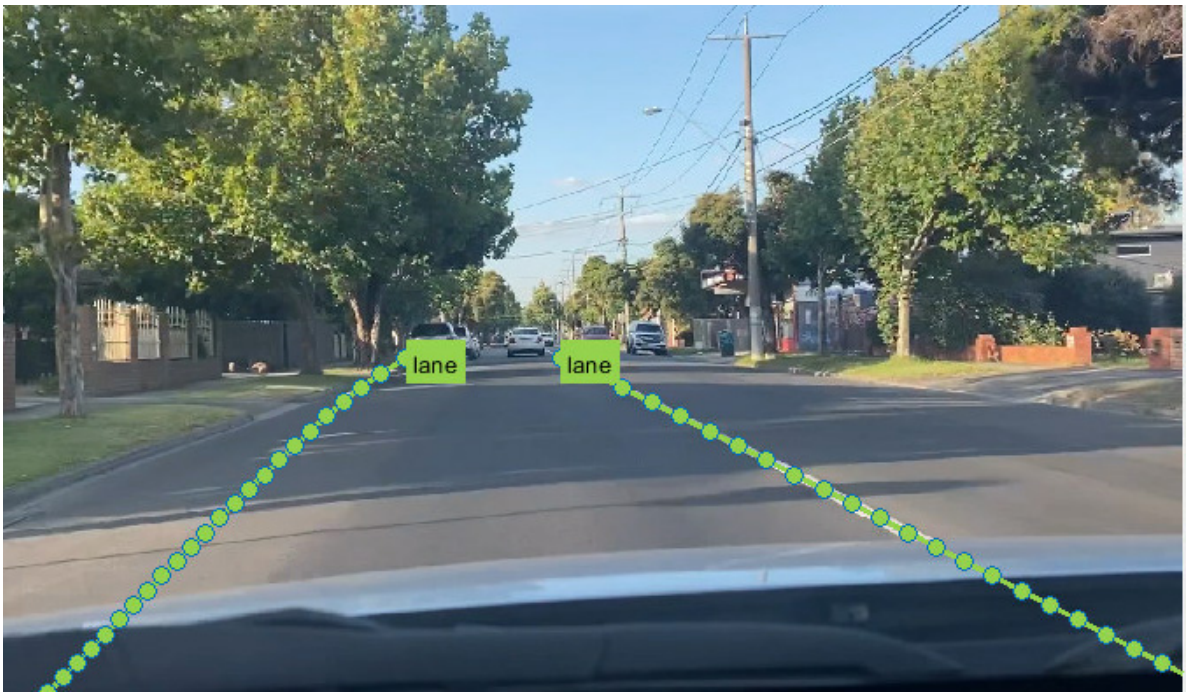

Figure 10. Lane detection in the unstructured road with shadows in the daytime.

Table 2. Performance metrics for evaluation of our proposed algorithm, compiled from [39].

Possibility	Condition 1	Condition 2
True positive	Existence of ground truth	Lane markers detected by the algorithm
False positive	Ground truth does not exist	Lane markers detected by the algorithm
False negative	Image has the ground truth	Lane markers detected by the algorithm
True negative	Image has no ground truth	No detection of lane markers by the algorithm

Table 3. Formulas to evaluate the algorithm's performance, compiled from [41].

Sr. No.	Metrics	Formula *
1	Accuracy(A)	$A = \frac{(TP+TN)}{(TP+TN+FP+FN)}$
2	Detection rate (DR)	$DR = \frac{(TP)}{(TP+FN)}$
3	False positive rate (FPR)	$FPR = \frac{(FN)}{(TP+FN)}$
4	False negative rate (FNR)	$FNR = \frac{(FN)}{(FN+TP)}$
5	True negative rate (TNR)	$TNR = \frac{(TN)}{(TN+TP)}$

* As per Table 2: True positive (TP), False positive (FP), True negative (TN), False negative (FN).

The accurate recognition rate under highway conditions reached 97.88 percent in the lane detection test experiment, and the average processing time per frame was 20.8 ms, showing that the lane detection algorithm achieved strong real-time performance and accuracy. Under unstructured road conditions, the accurate recognition rate was 98.73 percent, and the average processing time per frame was 21 ms, suggesting that the lane detection algorithm was still adaptable in the presence of poor light. Under unstructured road conditions during the daytime, the correct identification rate reached 97.71 percent, and the average processing time per frame was 22 ms, demonstrating that the lane detection algorithm had resilience and antijamming abilities in the presence of road tortuosity and background interference. According to the sorting and analysis of

false or missed detection images, most false or missed detection images were caused by motion blur, vehicle occlusion, poor light, and extreme road bending. Improving the lane detecting algorithm in the future will be important to increase its complexity and efficiency. Collecting road traffic videos under varying road conditions will also be necessary to continually enhance the inclusivity and robustness of the lane recognition system.

The BDD100K dataset, which consists of 100,000 videos and is based on driving data gathered by the Nexar network, is the largest and most diversified open driving dataset for computer vision research. The fact that this dataset is large-scale, diversified (in terms of location, weather, and time of day), and taken on real-world roads defines it and makes it relevant to researchers. These features are especially crucial when it comes to developing robust perception algorithms. BDD100K is the world's largest driving dataset created by the Berkeley DeepDrive Industrial Consortium in collaboration with the University of California, Berkeley Artificial Intelligence Research Lab. Researchers can access the datasets to test the algorithm on the BDD100K webpage.

To further verify the proposed algorithm's comprehensive performance across a range of challenging road conditions, the BDD100K dataset (general-purpose benchmark dataset) was employed for lane detection, in addition to the road driving videos, for simulation experiments. The BDD100K dataset includes 3626 training images and 2782 testing images. Each image sequence consists of 20 consecutive frames collected in 1 s, with the first 19 frames unstructured and the 20th frame labelled with the lane ground truth. The images in the collection can be loosely classified into four conditions: weather, daylight, pavement, and traffic environments [41,42].

In this study, 600 frames representing various situations were randomly picked from the custom dataset, and the lane recognition tests were carried out using the proposed algorithm. We used quantitative evaluations to verify the performance of the proposed methods. The most basic criterion for evaluation is accuracy, which evaluates overall classification performance based on properly categorised images. Table 4 shows the performance of randomly selected samples. Likewise, 150 images representing various conditions were selected randomly from BDD100K, and lane detection tests were carried out using the proposed algorithm. Table 5 shows the performance statistics of the algorithm with the BDD100K.

Table 4. The lane detection results under different road conditions using our dataset.

Video Sequence	Road Geometry	Total Number of Frames	True Positive	False Negative	Accuracy Rate	Detecting Time
1	Straight road in the day	474	461	12	97.88%	20 ms
2	Unstructured road	373	364	9	98.73%	21 ms
3	Structured road	563	550	13	97.71%	22 ms

Table 5. The experimental lane detection results under different conditions in the BDD100K dataset.

Video Sequences	Different Condition	Total Numbers of Frames	True Positive	False Negative	Detection Rate	Detecting Time
1	Straight road	324	311	13	97%	20 ms
2	Daytime	354	350	14	98.2%	21 ms
3	Speed (51–60)	363	350	13	98.36%	20 ms

Tables 4 and 5 show that the proposed algorithm performed well under various scenarios. The lane detection accuracy was as high as 98.73%, with a mean processing time (per frame) of 20 ms (millisecond). Overall, the accuracy rate ranges from 97% to 98.73%, and the detecting time ranges from 20 ms to 22 ms. When compared to the lane detection in the video sequences of driving, the mean lane detection rate was slightly lower, and the mean time interval (per frame) was significantly longer. However, in the BDD100K dataset, the proposed algorithm still provided higher performance along with acceptable accuracy and adaptability.

To validate the superiority of the proposed lane detection approach, it was compared to other algorithms described in the existing literature. The learning-based lane detection approach was used in [43]. Although the average processing time was very rapid compared to other algorithms, the mean detection accuracy was at the bottom and more probable to generate negative or missed detection in real driving scenarios. The model-based lane detection approach was used in [44]. The proposed algorithm was compared to conventional detection methods and the learning-based approaches in this research. As in [45,46], all relevant lane recognition tests were analysed using the proposed algorithm on the BDD100K dataset. Although the average detection accuracy was increased compared to that in [43], the average processing time was too high. Upon application of this algorithm to a real-world road scenario, difficulties were observed in real-time performance. The lane geometry was forecasted by training FastDraw Resnet in [45], and the proposed approach was evaluated successfully using the CVPR 2017 Tusimple lane marking challenge, albeit with problems on CULane datasets. Compared to conventional detection methods, this method achieved greater detection performance while keeping the values of FP and FN low, illustrating the competitive advantage of a learning-based approach methodology. Lane detection was conducted in [46] by deploying the ConvLSTM network model and using the Tusimple dataset and the authors' own lane datasets. The proposed algorithm was compared to existing traditional lane detection methods and deep learning-based approaches in Table 6. It can be seen from Table 6 that our developed algorithm performed well in three different scenarios as compared to existing literature. The proposed method and existing studies [41–43,45,46] all performed important lane detection test experiments on different datasets. However, our algorithm's average processing time was rapid compared to other algorithms. Our proposed method's average lane detection accuracy was greater than 98% and was more likely to generate fewer missing detection frames in real road scenarios.

Table 6. Comparison of our results with existing literature.

Methods	Road Geometry	Accuracy Rate (Exiting Literature)	Accuracy Rate (This Study)
[41] Traditional method	Structured road	<97.00%	98%
[43] Spatial Ray Feature extractions	Straight road	94.40%	98.73%
[44] Hough transform	Structured road	95.70%	98.40%
[45] Fast Draw Resnet	Structured road	95.2%	98.88%
[46] ConvLSTM (Deep learning)	Unstructured road	97.3%	≅ 99%

In comparison to the results in [45], the recognition accuracy of our algorithm was enhanced, the mean time interval for detection of lanes was reduced, and also the values of true positive and false positive were decreased dramatically, indicating the strengths and stability of the algorithm. Because of the challenging construction of the training algorithm and the high computational requirements, learning-based approaches are prone to issues such as long training time and inadequate algorithm performance. Compared to the results in [45,46], the proposed algorithm obtained better results with a similar data set, with a mean lane detection rate of 98.8 percent and a mean time interval of 22 ms for processing. This algorithm has apparent benefits in terms of improved performance. With the greater accuracy of lane detection, it can improve the safety of automated vehicles in real-world driving experience, results in a lesser time interval for processing the lane information, and assists in meeting the real-time information processing requirements of autonomous vehicles in real-world driving environments. Moreover, reliable detection under different operating conditions and dynamic environments is essential for improving the anti-interference abilities of intelligent vehicles.

4. Conclusions

Autonomous vehicles are predicted to arrive in the near future. These vehicles are expected to provide higher safety and reduce road traffic emissions. Lane detection and tracking are critical building blocks in developing autonomous vehicles or intelligent cars. In this study, a lane detection algorithm for autonomous cars under challenging road conditions and dynamic environments was developed. In particular, a lane detection algorithm was developed for intelligent vehicles on different road pavements (structured and unstructured roads) with challenging conditions (shadows) to solve challenges such as the low detection accuracy of old techniques and the unsatisfactory real-time performance of learning-based methodologies. To begin, datasets for performance evaluation were created using an interpolation method. Next, the learning-based approach was used to create an algorithm that uses the steering angle, yaw angle, and sideslip angle as inputs for the adaptive controller. Finally, simulation tests for the lane recognition method were carried out by utilising a road driving video in Melbourne, Australia and the BDD100K dataset created by the Berkeley DeepDrive Industrial Consortium. The mean detection accuracy ranges from 97% to 99%, and the detection time ranges from 20 to 22 ms under various road conditions with our proposed algorithm. The proposed lane detection algorithm ultimately outperformed conventional methodologies and learning-based approaches in terms of efficiency and overall performance in real-time, as well as detection efficiency and anti-interference abilities. The accuracy performance and the mean time interval were both greatly improved.

The proposed lane detection algorithm demonstrated significant benefits in terms of lane detection accuracy and algorithm time savings. Our algorithm considerably improved the driving safety of autonomous vehicles in real-world driving environments and efficiently achieved the real-time objective requirements of self-driving cars, in addition to playing an essential role in the automated vehicle for driving assistance. In addition, the inclusivity and accuracy of the lane detection algorithm could be further optimised and enhanced to maximize the method's overall performance. First and foremost, the entire model should be validated using a simulator that uses input images to simulate real-world road environments and provides feedback from the vehicle model. Furthermore, to address the limitations of the camera's lane detecting function (for crossroads and roads without lane marking), data from additional sensors, such as a LiDAR, could be used. For road friction, it is also suggested to consider the International Roughness Index (IRI). In challenging environments (e.g., inclement weather), sensor fusion between the camera and LiDAR could enhance detection. Lane detection could also be integrated with additional detection systems to produce autonomous driving vehicles. These systems could include other vehicles, pedestrians, traffic signs, and road text detection.

Author Contributions: Conceptualization, investigation, data collection, methodology, writing—original draft preparation, S.W.; supervision, writing—review and editing, N.S.; supervision, writing—review and editing, P.S. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Acknowledgments: The first author would like to acknowledge the Government of India, Ministry of Social Justice and Empowerment, for providing a full scholarship to pursue study at RMIT University.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Bimbraw, K. Autonomous Cars: Past, Present and Future—A Review of the Developments in the Last Century, the Present Scenario and the Expected Future of Autonomous Vehicle Technology. In Proceedings of the 12th International Conference on Informatics in Control, Automation and Robotics, Alsace, France, 21–23 July 2015; pp. 191–198.
2. Andreev, S.; Petrov, V.; Huang, K.; Lema, M.A.; Dohler, M. Dense Moving Fog for Intelligent IoT: Key Challenges and Opportunities. *IEEE Commun. Mag.* **2019**, *57*, 34–41. [CrossRef]
3. Chen, C.J.; Peng, H.Y.; Wu, B.F.; Chen, Y.H. A real-time driving assistance and surveillance system. *J. Inf. Sci. Eng.* **2009**, *25*, 1501–1523.
4. Zhou, Y.; Wang, G.; Xu, G.; Fu, G. Safety driving assistance system design in intelligent vehicles. In Proceedings of the 2014 IEEE International Conference on Robotics and Biomimetics (ROBIO 2014), Bali, Indonesia, 5–10 December 2014; pp. 2637–2642.
5. D'Cruz, C.; Zou, J.J. Lane detection for driver assistance and intelligent vehicle applications. In Proceedings of the 2007 International Symposium on Communications and Information Technologies, Sydney, NSW, Australia, 17–19 October 2007; pp. 1291–1296.
6. Kum, C.-H.; Cho, D.-C.; Ra, M.-S.; Kim, W.-Y. Lane detection system with around view monitoring for intelligent vehicle. In Proceedings of the 2013 International SoC Design Conference (ISOCC), Busan, Korea, 17–19 November 2013; pp. 215–218.
7. Scaramuzza, D.; Censi, A.; Daniilidis, K. Exploiting motion priors in visual odometry for vehicle-mounted cameras with non-holonomic constraints. In Proceedings of the 2011 IEEE/RSJ International Conference on Intelligent Robots and Systems, San Francisco, CA, USA, 25–30 September 2011; pp. 4469–4476.
8. Li, B.; Zhang, X.; Sato, M.; Sato, M. Pitch angle estimation using a Vehicle-Mounted monocular camera for range measurement. In Proceedings of the 2014 12th International Conference on Signal Processing (ICSP), Hangzhou, China, 19–23 October 2014; pp. 1161–1168.
9. Schreiber, M.; Konigshof, H.; Hellmund, A.-M.; Stiller, C. Vehicle localization with tightly coupled GNSS and visual odometry. In Proceedings of the 2016 IEEE Intelligent Vehicles Symposium (IV), Gothenburg, Sweden, 19–22 June 2016; pp. 858–863.
10. Zhang, Y.; Liang, W.; He, H.; Tan, J. Perception of Vehicle and Traffic Dynamics Using Visual-Inertial Sensors for Assistive Driving. In Proceedings of the 2018 IEEE International Conference on Robotics and Biomimetics (ROBIO), Kuala Lumpur, Malaysia, 12–15 December 2018; pp. 538–543.
11. Wang, J.; Ma, H.; Zhang, X.; Liu, X. Detection of Lane Lines on Both Sides of Road Based on Monocular Camera. In Proceedings of the 2018 IEEE International Conference on Mechatronics and Automation (ICMA), Changchun, China, 5–8 August 2018; pp. 1134–1139.
12. Li, Y.; Zhang, W.; Ji, X.; Ren, C.; Wu, J. Research on Lane a Compensation Method Based on Multi-Sensor Fusion. *Sensors* **2019**, *19*, 1584. [CrossRef] [PubMed]
13. Zheng, B.; Tian, B.; Duan, J.; Gao, D. Automatic detection technique of preceding lane and vehicle. In Proceedings of the 2008 IEEE International Conference on Automation and Logistics, Qingdao, China, 1–3 September 2008; pp. 1370–1375.
14. Haselhoff, A.; Kummert, A. 2D line filters for vision-based lane detection and tracking. In Proceedings of the 2009 International Workshop on Multidimensional (nD) Systems, Thessaloniki, Greece, 29 June–1 July 2009; pp. 1–5.
15. Son, J.; Yoo, H.; Kim, S.; Sohn, K. Real-time illumination invariant lane detection for lane departure warning system. *Expert Syst. Appl.* **2015**, *42*, 1816–1824. [CrossRef]
16. Amini, H.; Karasfi, B. New approach to road detection in challenging outdoor environment for autonomous vehicle. In Proceedings of the 2016 Artificial Intelligence and Robotics (IRANOPEN), Qazvin, Iran, 9 April 2016; pp. 7–11.
17. Kong, H.; Sarma, S.E.; Tang, F. Generalizing Laplacian of Gaussian Filters for Vanishing-Point Detection. *IEEE Trans. Intell. Transp. Syst.* **2012**, *14*, 408–418. [CrossRef]
18. Hervieu, A.; Soheilian, B. Roadside detection and reconstruction using LIDAR sensor. In Proceedings of the 2013 IEEE Intelligent Vehicles Symposium (IV), Gold Coast, QLD, Australia, 23–26 June 2013; pp. 1247–1252.
19. Hata, A.Y.; Osorio, F.S.; Wolf, D.F. Robust curb detection and vehicle localization in urban environments. In Proceedings of the IEEE Intelligent Vehicles Symposium, Piscataway, MI, USA, 8–11 June 2014; pp. 1257–1262.
20. Geiger, A.; Lauer, M.; Wojek, C.; Stiller, C.; Urtasun, R. 3D Traffic Scene Understanding From Movable Platforms. *IEEE Trans. Pattern Anal. Mach. Intell.* **2013**, *36*, 1012–1025. [CrossRef] [PubMed]
21. Liang, J.-H.; Lee, C.-H. Efficient collision-free path-planning of multiple mobile robots system using efficient artificial bee colony algorithm. *Adv. Eng. Softw.* **2015**, *79*, 47–56. [CrossRef]
22. Bosaghzadeh, A.; Routeh, S.S. A novel PCA perspective mapping for robust lane detection in urban streets. In Proceedings of the 2017 Artificial Intelligence and Signal Processing Conference (AISP), Shiraz, Iran, 25–27 October 2017; pp. 145–150.
23. He, B.; Ai, R.; Yan, Y.; Lang, X. Accurate and robust lane detection based on Dual-View Convolutional Neutral Network. In Proceedings of the IEEE Intelligent Vehicles Symposium, Gothenburg, Sweden, 19–22 June 2016; pp. 1041–1046.
24. Badrinarayanan, V.; Kendall, A.; Cipolla, R. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.* **2017**, *39*, 2481–2495. [CrossRef] [PubMed]
25. Pan, X.; Shi, J.; Luo, P.; Wang, X.; Tang, X. Spatial as deep: Spatial cnn for traffic scene understanding. In Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, New Orleans, LA, USA, 2–7 February 2018; Volume 32.
26. Tan, H.; Wang, J.F.; Zhang, K.; Cui, S.M. Research on Lane Marking Lines Detection. *Appl. Mech. Mater.* **2013**, *274*, 634–637. [CrossRef]

27. Fernandez, C.; Izquierdo, R.; Llorca, D.F.; Sotelo, M.A. Road curb and lanes detection for autonomous driving on urban scenarios. In Proceedings of the 17th International IEEE Conference on Intelligent Transportation Systems (ITSC), Qingdao, China, 8–11 October 2014; pp. 1964–1969.
28. Kumar, P.; McElhinney, C.P.; Lewis, P.; McCarthy, T. An automated algorithm for extracting road edges from terrestrial mobile LiDAR data. ISPRS J. Photogramm. *Remote Sens.* **2013**, *85*, 44–55. [CrossRef]
29. Wang, Y.Q.; Liang, B.; Zhuang, L.L. Applied Technology in Unstructured Road Detection with Road Environment Based on SIFT-HARRIS. *Adv. Mater. Res.* **2014**, *1014*, 259–262. [CrossRef]
30. Xiaolin, L.; Yufeng, J.; Yan, G.; Xiaoxue, F.; Weixing, L. Unstructured road detection based on region growing. In Proceedings of the 2018 Chinese Control And Decision Conference (CCDC), Shenyang, China, 9–11 June 2018; pp. 3451–3456.
31. Wang, G.; Wu, J.; He, R.; Yang, S. A Point Cloud-Based Robust Road Curb Detection and Tracking Method. *IEEE Access* **2019**, *7*, 24611–24625. [CrossRef]
32. Hernandez, D.C.; Filonenko, A.; Shahbaz, A.; Jo, K.-H. Lane marking detection using image features and line fitting model. In Proceedings of the 2017 10th International Conference on Human System Interactions (HSI), Ulsan, Korea, 17–19 July 2017; pp. 234–238.
33. Li, L.; Luo, W.; Wang, K.C. Lane Marking Detection and Reconstruction with Line-Scan Imaging Data. *Sensors* **2018**, *18*, 1635. [CrossRef] [PubMed]
34. Zhang, X.; Yang, W.; Tang, X.; Liu, J. A Fast Learning Method for Accurate and Robust Lane Detection Using Two-Stage Feature Extraction with YOLO v3. *Sensors* **2018**, *18*, 4308. [CrossRef] [PubMed]
35. Tian, Y.; Gelernter, J.; Wang, X.; Chen, W.; Gao, J.; Zhang, Y.; Li, X. Lane marking detection via deep convolutional neural network. *Neurocomputing* **2018**, *280*, 46–55. [CrossRef] [PubMed]
36. Huang, Q.; Liu, J. Practical limitations of lane detection algorithm based on Hough transform in challenging scenarios. *Int. J. Adv. Robot. Syst.* **2021**, *18*, 17298814211008752. [CrossRef]
37. Feng, J.; Wu, X.; Zhang, Y. Lane Detection Base on Deep Learning. In Proceedings of the 2018 11th International Symposium on Computational Intelligence and Design (ISCID), Hangzhou, China, 8–9 December 2018; pp. 315–318.
38. Van Gansbeke, W.; De Brabandere, B.; Neven, D.; Proesmans, M.; Van Gool, L. End-to-end Lane Detection through Differentiable Least-Squares Fitting. In Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW), Seoul, Korea, 27–28 October 2019; pp. 905–913.
39. Waykole, S.; Shiwakoti, N.; Stasinopoulos, P. Review on Lane Detection and Tracking Algorithms of Advanced Driver Assistance System. *Sustainability* **2021**, *13*, 11417. [CrossRef]
40. Camacho, E.F.; Alba, C.B. *Model Predictive Control*; Springer: Berlin/Heidelberg, Germany, 2007.
41. Neven, D.; De Brabandere, B.; Georgoulis, S.; Proesmans, M.; Van Gool, L. Towards End-to-End Lane Detection: An Instance Segmentation Approach. In Proceedings of the IEEE Intelligent Vehicles Symposium, Rio de Janeiro, Brazil, 8–13 July 2018; pp. 286–291.
42. Liu, P.; Yang, M.; Wang, C.; Wang, B. Multi-Lane Detection via Multi-Task Network in Various Road Scenes. In Proceedings of the 2018 Chinese Automation Congress (CAC), Xi'an, China, 30 November–2 December 2018; pp. 2750–2755.
43. Kuhnl, T.; Kummert, F.; Fritsch, J. Spatial ray features for real-time ego-lane extraction. In Proceedings of the 2012 15th International IEEE Conference on Intelligent Transportation Systems, Anchorage, AK, USA, 16–19 September 2012; pp. 288–293.
44. Zheng, F.; Luo, S.; Song, K.; Yan, C.-W.; Wang, M.-C. Improved Lane Line Detection Algorithm Based on Hough Transform. *Pattern Recognit. Image Anal.* **2018**, *28*, 254–260. [CrossRef]
45. Philion, J. FastDraw: Addressing the Long Tail of Lane Detection by Adapting a Sequential Prediction Network. In Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 15–20 June 2019; pp. 11574–11583.
46. Zou, Q.; Jiang, H.; Dai, Q.; Yue, Y.; Chen, L.; Wang, Q. Robust Lane Detection From Continuous Driving Scenes Using Deep Neural Networks. *IEEE Trans. Veh. Technol.* **2019**, *69*, 41–54. [CrossRef]