A Literature Review of Steering Angle Prediction Algorithms for Autonomous Cars

Shang Shi[1] and Jian Qu[2*]

[1,2]Faculty of Engineering and Technology, Panyapiwat Institute of Management, Nonthaburi, Thailand
E-mail: 6572100022@stu.pim.ac.th, jianqu@pim.ac.th[*]

Received: February 22, 2024 / Revised: March 30, 2024 / Accepted: June 25, 2024

***Abstract*—Road tracking is a critical requirement for the development of autonomous cars. It requires the car to continuously navigate within the designated driving area to avoid any deviation. The computation of steering angles is an essential aspect of achieving autonomous driving. Autonomous steering angle techniques, which are essential for enabling road tracking in autonomous cars, are comprehensively reviewed in this paper. Autonomous steering techniques, mainly involving computer vision methods and end-to-end deep learning approaches, are currently receiving considerable attention. The primary objective of this paper is to identify and reimplement state-of-the-art models in end-to-end deep learning approaches within practical scenarios. We carry out a performance evaluation of each model utilizing real-world tests using scale model cars. Furthermore, we offer perspectives on potential avenues for future research and applications. These may include adaptive modifications to dynamic road conditions, the creation of more effective real-time decision-making algorithms, or the investigation of applications in intricate traffic situations.**

***Index Terms*—End-to-End, Neural Network, Road Tracking, Steering Predicate**

I. Introduction

In the domain of road tracking, the calculation of steering angles is a crucial task in using sensor data to steer the car. Typically, LiDAR or radar sensors employ the Euclidean method for calculating steering angles, which is primarily suitable for gradual and smooth turns. Nevertheless, reliance on these sensors may lead to concerns such as system redundancy, cumulative error, and increased cost, particularly in multi-sensor fusion setups.

Therefore, calculating steering angles via camera for road-tracking is a promising route for the advancement of autonomous cars. This paper reviews the process of utilizing camera-derived external information to determine steering angles, enabling precise road tracking and vehicle control. These methods fall into two categories: Traditional computer vision techniques and end-to-end deep learning approaches.

Traditional computer vision methods can be effective but often require high computational requirements, which can impact the real-time performance of autonomous driving systems. On the other hand, the evolution of deep learning suggests that end-to-end methods show promise in overcoming these challenges. Although numerous end-to-end deep learning models have been proposed, some are based on simulations, others heavily rely on specific datasets, and some are tested solely on scale model cars. The evaluation of these models is still an open concern, which creates uncertainty regarding the direction for developing new models.

The purpose of this paper is to evaluate the performance of existing end-to-end deep learning models by replicating and rigorously assessing them in real-world scenarios. This will be achieved through testing on-scale model cars and employing consistent evaluation criteria. Our purpose in focusing on these challenges is to provide clarity on the effectiveness of these models in real-world applications and to bridge the gap between simulated or dataset performance and real-world scenarios.

II. Literature Review

Traditional vision and robotics techniques are often challenged by noise and variability in autonomous navigation. In contrast, artificial neural networks have shown exceptional performance and adaptability in noisy and variable domains.

A. End-to-End Deep Learning

Pomerleau and Dean introduces ALVINN, an autonomous navigation system driven by a neural network. The article explains the design and training methodology of ALVINN, which is a three-layer backpropagation network. The system is specifically designed for road tracking and uses road images captured by a camera and a laser range finder to navigate the car [1].

The effectiveness of ALVINN has been demonstrated through training on simulated road images. This showcased commendable performance, which was further validated on an autonomous navigation test vehicle at Carnegie Mellon University. This experiment serves as an early illustration of the ability of the ALVINN network to navigate competently on real roads in specific scenarios, and it marks a pioneering effort in the use of neural networks for the navigation of autonomous cars.

The ALVINN technique has demonstrated its effectiveness in driving scenarios with few obstacles and simplicity. This is even though it is a shallow network that uses only pixel inputs to anticipate actions. This performance highlights the potential of neural networks to support autonomous navigation. Chen et al. employed the deep convolutional neural network structure of AlexNet to correlate image features with perceptual metrics. They employed the driving game TORCS, gathering 484,815 training images and conducting model training and testing within the TORCS environment. The results demonstrate the adeptness of the model in the virtual environment of TORCS, which facilitates automated driving and provides results for visual representation [2].

It is important to note that although the model performs well in the virtual environment of TORCS, its effectiveness may be affected by differences in graphical outputs when applied to real driving video data. These variations could potentially impact the adaptability and performance of the model in real-world scenarios.

Bojarski et al. describe an end-to-end learning approach for autonomous car control. The paper outlines the training process of a convolutional neural network that translates pixel inputs from a single forward-facing camera into steering commands. The network architecture uses standard convolution-pooling layers to extract features and three fully connected layers to establish a control strategy [3].

During simulated environmental tests that covered up to 100 miles and lasted 3 hours, the network demonstrated stable driving capabilities. It operated autonomously around 90% of the time. Real-world road tests showed that the autonomous performance of the network reached 98% without any human intervention. This end-to-end approach demonstrated efficiency, requiring minimal manual driving training data to navigate conventional roads, motorways, car parks, and even unpaved roads.

However, it is important to note that the model lacks sufficient explanation of its internal knowledge representation and reasoning process. Furthermore, its ability to generalize and handle unseen data needs further validation. The performance of the model in both simulated and real-world environments has not been comprehensively evaluated, especially in complex real-world driving scenarios where it is still difficult to determine the performance of the model. These uncertainties call for more extensive research and validation efforts.

Bojarski et al. investigate the image recognition capabilities of PilotNet for automotive steering control. The paper presents a methodology for the identification of “salient objects” in images, which are crucial in influencing the steering decisions made by PilotNet. This innovative approach sheds light on the decision-making process of PilotNet [4].

The results suggest that PilotNet can effectively identify important road elements, including lane markings, road edges, and other vehicles. Notably, it can also detect more subtle details such as roadside bushes, which are difficult to capture using traditional methods. This method offers a simple yet effective way to gain insight into the learning mechanism of the end-to-end autonomous car model. It is a significant step towards interpreting the functionality of the model and building trust. However, it is important to note that the training dataset used in the paper was relatively limited, which may have restricted the range of driving scenarios covered. This limitation suggests that model learning is limited to the specifics of the provided data, which may limit performance in more diverse and extensive driving scenarios. As a result, further research and validation is needed to determine the ability of the model to generalize across different scenarios and to adapt effectively to a wider range of driving conditions.

This proposal by Rausch et al. propose using a convolutional neural network as a control strategy. This involves directly correlating monocular camera inputs to steering control outputs for comprehensive, End-to-end control. To gather marker data, they employed vehicle simulation software and conducted manual driving learning to document input images alongside corresponding steering angle outputs for network training. The paper presents a four-layer convolutional neural network structure designed for a specific purpose. Various optimization algorithms were evaluated for their training efficacy. The findings indicate that the network has robust generalization capabilities, as it demonstrated proficiency with unseen data and the ability to emulate human driving behaviors [5].

However, it is important to acknowledge that the simplicity of the model design, limited to four layers, may limit its capacity and hinder the acquisition of complex mapping relationships. Additionally, the reliance on simulation-based training without real vehicle testing data hampers the direct assessment of the real-world control effectiveness of the model. This gap raises concerns about the adaptability and performance of the model in authentic road scenarios. Further research and validation are necessary to address these uncertainties.

Do et al. present a real-time autonomous car navigation system that operates on a Raspberry Pi single-board computer, using deep convolutional neural networks. The system is designed for low cost and utilizes a Raspberry Pi, a front-facing camera, and a remote-controlled vehicle. The team collected driving data and augmented it to acquire a substantial training dataset. The proposed end-to-end deep neural network consists of a 9-layer architecture. It includes 5 convolutional layers for feature extraction and 4 fully connected layers for learning steering control tasks. This model predicts steering angles directly from raw images, enabling real-time autonomous driving functionality [6].

However, the limited computational power of the Raspberry Pi introduces significant latency in the real-time reasoning process. This delay could potentially affect the responsiveness of the system, particularly in driving scenarios that require rapid decision-making. To improve real-time performance, optimizing the model structure or transitioning to a higher-performance computing platform could be crucial. Furthermore, the hardware limitations of the Raspberry Pi may restrict the handling of more complex models or larger datasets, which could potentially impact the accuracy of the model and its ability to generalize. Addressing these computational power constraints is a critical consideration for improving system performance.

Chen et al. adopt a simpler architecture consisting of a 3-layer convolutional and a 2-layer fully connected neural network for an end-to-end mapping learning task, mapping raw images to steering angles. The comma.ai dataset was used for training and testing, and the model demonstrated proficiency in generating accurate steering control commands for vehicle navigation [7].

However, there is a concern about the potential for overfitting of the data, which could have an impact on the ability of the model to generalize across different scenarios. Overfitting occurs when the model excessively tailors itself to specific features and noise within the training data, resulting in inadequate adaptation to new and distinct situations not encountered during training. This limitation may result in suboptimal model performance in real-world applications, especially when dealing with new and unobserved scenarios.

Eraqi et al. propose a novel architecture, the Convolutional Long Short-Term Memory Recurrent Neural Network (C-LSTM), designed for End-to-end learning of autonomous car control direction. The C-LSTM framework combines a Convolutional Neural Network (CNN) for visual feature extraction with a Long Short-Term Memory network (LSTM) to capture both visual information and dynamic temporal dependencies. This design allows the model to understand both static and dynamic dependencies that are essential for driving control [8].

The approach redefines the problem of controlling direction regression as a deep classification issue. This is achieved by establishing topological relationships among discrete classification outputs, which enhances the regression objective. The innovation lies in transforming the regression problem into a classification one and refining the accuracy and resilience of the regression objective through optimized relationships between classification outputs.

The model uses a hybrid CNN and LSTM architecture to efficiently extract significant features from image data and learn temporal dynamics, enhancing its ability to predict and control driving directions. This approach is expected to improve the performance and robustness of control systems used in autonomous cars.

Jhung et al. [9] introduce an end-to-end steering controller that uses Convolutional Neural Networks (CNN) for autonomous cars. They demonstrate that this approach enhances driving performance compared to traditional CNN-based methods. The paper explains the use of a neural network called DAVE-2SKY, which was pre-trained using camera-acquired vehicle images. Subsequently, the network is enabled to infer steering angles for lateral control in autonomous cars through augmented closed-loop training. The paper utilizes the PreScan simulator and the Caffe deep learning framework for training within a Software-In-the-Loop (SIL) simulation environment. To validate the proposed end-to-end controller, experiments involve implementing the autonomous car on a DRIVE™ PX2 computer, assessing system performance through simulations and real road tests. The results suggest that the CNN-based end-to-end controller can provide robust steering control, even under partially observable road conditions. This indicates the feasibility of an autonomous vehicle controlled entirely by such a steering controller. However, the paper lacks sufficient discussion on the scalability of the method and its practicality in large-scale real-world scenarios. The article does not extensively address crucial aspects such as computational efficiency and real-time performance in authentic road conditions, which are pivotal for deploying the method on real roads. Conducting more comprehensive experiments and research could provide insights into the applicability of the method and its performance in real-world environments.

Sharma, Tewolde, and kwon implemented an end-to-end learning approach for lateral position control in automated vehicle driving a critical aspect of autonomous driving. The model was tested on the e-road training track and demonstrated the ability to traverse a complete circle, indicating that it had learned human driving patterns. In evaluations on an unseen single-lane track, the model maintained lane position for approximately 89.02% of the time,

demonstrating strong generalization. When navigating the multi-lane CG track 2, the model completed the loop with only two-lane changes, achieving 96.62% autonomy [10].

The main conclusion of the paper is that the model closely mirrors the human driving model and is highly adaptable in different environments. However, it is important to note that the model only addresses lateral control and not longitudinal control. Despite its exceptional performance in simulation environments, its safety and performance in real driving conditions remain unconfirmed. Therefore, it is imperative to conduct additional field tests and validations to ensure the model's robustness and safety in various real-world environments.

Yang et al. employ a common CNN+LSTM structure within an end-to-end multitasking framework for vehicle control prediction. The model architecture is enhanced by utilizing multimodal multitask learning to improve prediction accuracy. This involves integrating visual features captured by the forward-looking camera and feedback vehicle speed inputs from the initial 10-time steps as inputs to the model. The model simultaneously employs multiple fully connected layers to predict steering wheel angle and vehicle speed values for the following moment [11]. This approach combines visual information with vehicle speed data and uses a multi-task learning strategy to improve predictions of vehicle behavior. Although the structure of the model is not particularly innovative, its ability to combine different data sources and handle multiple prediction tasks is promising. This approach has the potential to enhance prediction performance for vehicle control, demonstrating improved robustness and accuracy in real-world applications.

Smolyakov utilized the CarND Udacity simulator to collect data and simulate vehicle driving, assembling a dataset consisting of left, center and right camera images along with steering angle data. The researcher explored models with limited parameters by introducing two convolutional neural network structures, each with a modest number of parameters suitable for embedded system applications. The experimental findings demonstrate that neural networks with only 26,000 parameters can achieve a commendable 78.5% accuracy in predicting steering angles [12].

This research showcases innovative methodology by using simulators to quickly generate training data and explore the potential of compact models for steering angle prediction. However, to ensure the prediction accuracy and real-world applicability of the model in autonomous driving scenarios, it is necessary to further validate its performance using genuine road conditions. Simulated data may not capture the full range of variations and complexities encountered on real roads, which could better illustrate the robustness and practicality of the model. Bechtel et al. developed DeepPicar, an autonomous driving testbed that uses a Raspberry Pi as the computing platform and employs the same CNN model as the NVIDIA DAVE-2 project for real-time control. However, the study does not include comparative performance tests against high-performance GPU platforms, which limits the depth of its comparative insights [13].

The paper discusses the challenges of implementing end-to-end deep learning for real-time applications in autonomous driving and offers valuable conclusions. However, it highlights the need for further optimization of models and algorithms, especially in enhancing efficiency and real-time performance when computational resources are limited, such as with low-cost computing platforms like the Raspberry Pi. Optimization techniques could include model compression, quantization, or tailored adaptations for embedded platforms to improve real-time and overall performance.

DeepPicar is a promising low-cost autonomous driving testbed, and this research provides valuable insights into real-time deep learning applications in autonomous driving. However, there is still room for further improvement and optimization to enhance its effectiveness and performance.

Sharma et al. applied two separate models for steering and speed control, aiming to improve performance by running them in parallel. The steering model had eight convolutional layers, while the speed model had five. To allow for parallel computation of model predictions, they used a multithreading mechanism [14].

Using the open-source TORCS simulation environment, both models demonstrated 100% autopilot capability on the e-road and CG-track 2, the roads where training data was gathered. This highlights the effectiveness of the training. However, the models showed subpar results on other roads, indicating limitations in their generalization ability. Improving the generalization of the models may require optimizing the network structure and collecting more diverse experimental data.

The researcher discussed the challenges of implementing multi-threading and its potential impact on real-time performance. They noted that thread switching and data transfer could cause delays that affect system responsiveness. Additionally, the researcher pointed out that the independence of steering and speed control tasks may limit GPU resource utilization.

Wang et al. proposed a hybrid architecture that combines CNN and LSTM networks to improve steering angle and speed predictions. By incorporating spatial-temporal and spatial information from image

sequences and speed feedback inputs, this approach resulted in more accurate predictions compared to the CNN single-task network [15].

The study also compared traditional LSTM networks with the proposed state-passing LSTM in terms of prediction time and effectiveness. The results indicate that integrating the auxiliary task with the state-passing LSTM significantly reduces time overhead while maintaining a high level of prediction accuracy.

The paper conducted online tests in both the GTAV simulation environment and real-road scenarios and observed a decrease in intervention frequency from 6 to 2 instances. This implies an improvement in the quality of actual control, suggesting that the model has the potential to enhance automatic control and reduce human intervention in real-world scenarios.

The research combined CNN and LSTM networks using improved models and training methods, resulting in a significant improvement in the accuracy of steering angle and speed predictions. The validation of these improvements was carried out through simulated environments and real-road tests, indicating the potential application of the methodology in automated driving.

Shuyang et al. conducted a study to explore the prediction of steering wheel angle in autonomous cars using two deep learning models. The first model combined a 3D convolutional layer with residual concatenation and an LSTM network to capture spatial and temporal relationships within visual features. This model extracted visual features using 3D convolution and captured sequential relationships through an LSTM network. During model training, transfer learning was employed by keeping the parameters of the initial 45 layers of ResNet50 fixed and feeding their output through a fully connected layer to predict the steering wheel angle [16].

The evaluation of the Udacity dataset resulted in RMSE values of 0.1123 and 0.0709 for the two models on the test set, respectively. Additionally, the models achieved 10th and 4th positions in the Udacity Challenge.

This paper utilizes various deep learning architectures, specifically combining 3D convolutional layers with LSTM networks, to improve the accuracy of steering wheel angle predictions by capturing spatial and temporal relationships. The models also demonstrate promising performance on both the Udacity dataset and the Challenge, indicating potential applicability in autonomous driving scenarios.

Lee et al. proposed a novel method for capturing spatiotemporal elements in images by combining CNN and LSTM networks. The method takes into account the influence of time-series data on driving judgments. The authors collected extensive real driving data using a driving simulator, which significantly enriched the training dataset of the network. As a result, the proposed method accurately predicts steering wheel angles, even on challenging S-curves, demonstrating its efficacy in real driving scenarios [17].

However, although simulated environments are used for data collection and training, there are still discrepancies between simulated and real driving environments. This difference may limit the model's ability to adapt to the diverse variations and intricacies encountered on actual roads. Therefore, further validation is required to assess the performance and generalizability of the model in real driving environments.

Although the model excels in simulated environments, additional data acquisition and fine-tuning may be necessary to enhance its generalizability and adaptability for real-world operations.

Kumar et al. propose a fully end-to-end deep learning model for steering wheel angle estimation. The model uses direct image inputs from the forward-looking camera to predict corresponding steering angles without intermediate feature extraction. The authors employ Dropout regularization within the CNN model to counter overfitting, alongside meticulous hyper-parameter tuning. These efforts have resulted in an accuracy of 98.6%, which is higher than previous benchmarks in the literature.

The innovation of the paper lies in its comprehensive end-to-end methodology, which eliminates the need for intermediate feature extraction or processing steps in steering angle estimation. Furthermore, the use of Dropout regularization and hyper-parameter tuning significantly improves model performance. The achieved accuracy highlights the potential application of this approach in advancing autonomous driving technology.

Mishra et al. [17] utilized a Convolutional Neural Network (CNN) to estimate steering angles, harnessing the power of deep learning and CNNs to enable automated learning of driving skills. The CNN model learned and predicted steering commands directly from raw pixel data captured by a single front-facing camera, utilizing the power of deep learning and CNNs to enable automated learning of driving skills. Data was collected using the Udacity autonomous car simulator, which provided a virtual environment to simulate diverse driving scenarios and gather data for training and validating the model's performance.

Sokipriala and Jonah utilised pre-trained deep convolutional neural networks to predict steering wheel control angles in autonomous cars. The paper employed transfer learning methods and utilised the VGG16 model to extract image features from a comprehensive image classification dataset such as ImageNet. This technique allows for resource sharing

and reduces the number of training parameters, thereby enhancing model performance [18].

The research presents a regression problem for predicting the steering wheel control angle. Mean Square Error (MSE) is used as the loss function for supervised learning. To adapt this approach for autonomous driving, the pre-trained model is further fine-tuned, resulting in more precise predictions of the steering wheel control angle. This method refines the model to better suit the specific requirements of autonomous driving.

Showrov et al. conducted an investigation involving collecting vehicle front images paired with corresponding steering angle data. They compared the performance of deep learning models such as VGG16, ResNet-152, DenseNet-201, and Nvidia. The study revealed that the Nvidia model outperformed the other pre-trained models, showcasing a Mean Square Error (MSE) value of 0.3521 [19].

The Nvidia model performed well in the trial. However, the paper does not mention any performance evaluation or validation in an autonomous driving system used in the real world. Therefore, there is a gap in validating the feasibility and effectiveness of this method in real-world applications. To implement this model in practical driving scenarios, further field tests and validation are imperative to ensure its reliability and accuracy within authentic autonomous driving systems.

Podbucki et al. presented a method for automated trolley driving using a Jetson Nano microcomputer and a ResNet18 convolutional neural network. The experimental outcomes showed that this method enabled stable cart movement along a predefined track, with improved operation as the experiments progressed. The implementation used open-source frameworks such as PyTorch and TensorRT for real-time inference, meeting the real-time demands of cart autopilot [20].

However, the paper lacks crucial algorithm evaluation metrics, such as quantitative measurements like success rate and tracking error. These metrics are pivotal in evaluating algorithm performance, enabling the assessment of accuracy and reliability in ensuring stable travel along a predefined track. The absence of these metrics hinders a comprehensive evaluation and comparison of the performance of the algorithm with other methodologies. Therefore, a comprehensive evaluation of the usefulness and effectiveness of the model requires that its performance metrics be thoroughly assessed and reported.

Khidhir et al. conducted a comparative analysis of three deep learning models for end-to-end autonomous driving. The study highlights the advantages of ResNet18 and DenseNet121 in steering angle prediction and driving performance. The results indicate that ResNet18 is more accurate, while DenseNet121 is more consistent across diverse driving tracks [22].

However, the dataset used in the study was relatively limited, consisting of only 681 images. The validation process and presentation of results may need further refinement, and a larger dataset coupled with a comprehensive validation methodology could better illustrate the performance and stability of the model. Expanding the dataset and refining validation techniques would contribute to a more robust demonstration of the model's capabilities.

Hassan et al. developed a lightweight Convolutional Neural Network (CNN) that predicts vehicle steering angles using raw image inputs. The model was trained and evaluated using data from the CARLA simulator. The experimental results showed that this lightweight model achieved comparable steering angle prediction performance to Nvidia's PilotNet, despite having only a quarter of the parameters. This highlights the ability of lightweight models to deliver satisfactory steering angle predictions with a significantly smaller parameter size, which could be particularly beneficial in resource-constrained environments or situations demanding compact model sizes [23].

Ding et al. introduced a deep learning model that integrates the CBAM attention mechanism into the ResNet18 structure, enhancing its ability to concentrate on specific objects and extract features effectively. This model accomplishes road tracking and obstacle avoidance tasks. The authors improved the model's performance through hyperparameter optimization and thoughtful parameter combinations. Experimental findings demonstrate impressive scores. The study reports that the model achieved 98% accuracy in a training setting and 72% in a mildly noisy environment, significantly higher than the 32% benchmark set by existing methods [24].

However, the study does not discuss potential challenges and limitations that may arise in practical applications. Addressing these aspects could provide valuable insights into the adaptability and performance of the model under real-world conditions. Understanding the constraints and potential obstacles in practical implementation is crucial when assessing the model's viability beyond controlled environments.

Zihao et al. developed a self-sufficient smart car system that uses a single-camera sensor and the Jetson Nano as its central processing unit. They introduced a novel neural network structure called MT-ResNet26 to achieve autonomous driving. This structure leverages ResNet residual blocks to enhance lower-layer feature extraction while integrating Batch Normalization (BN) layers for optimized training. This architecture aims to avoid the data demands posed by excessively deep or wide networks, prioritizing generalization ability and robustness [25].

However, the effectiveness of the model was assessed solely using qualitative scoring systems, lacking quantitative metrics for comprehensive analysis of the results. Incorporating quantitative measures can allow for a more extensive evaluation, providing precise insights into how the model performs, where strengths lie, and where potential improvements exist.

Ding et al. propose an Advanced Driver Assistance System (ADS) to execute five autonomous driving tasks using a single-camera sensor. The system comprises two neural network models: an end-to-end model named Efficientnet_b0-SA-RE for overall functionality and a safety assistance model, Efficientnet_b0-CA, designed for specific safety measures. These models collaborate to achieve tasks related to route tracking, turn sign recognition, lane changing, obstacle stopping, and traffic light recognition [26].

However, the proposal lacks an in-depth analysis comparing different model fusion schemes. Such an analysis would be valuable in determining the most effective approach for integrating the two models and optimizing their collaborative performance. Evaluating and contrasting alternative fusion methods could improve our understanding of how different techniques impact the overall efficiency, accuracy, and adaptability of the system across various driving scenarios.

Ding et al. implemented road tracking and traffic sign recognition and designed three real simulated environments with three sets of experiments. Their model ResNet34_B32 is capable of road tracking in both trained and untrained environments [27].

Li et al. achieved multi-task autonomous driving by adjusting the model's network structure. However, they found that the trained neural network model performed poorly in untrained scenarios after implementing multi-task autonomous driving. Therefore, Li proposed improving the model's transfer efficiency in new scenarios through transfer learning [28].

Shi et al. proposed the RDNet18-CA model, which not only achieves road tracking but also completes multiple tasks. Additionally, they introduced a new loss function LiSHTL-S to enhance the model's performance in untrained scenarios [29].

III. Methodology

This section mainly introduces the design of the evaluation method for the model, which is divided into the construction of the model car, road tracking design, data collection, and new evaluation methods. The experimental design aims to build a Jetbot model car to collect the dataset. Subsequently, each test model is used to train the dataset to obtain the model. Then, these models are deployed onto the Jetbot model car and placed on the track for testing. Evaluation data is collected, and assessment criteria are applied for calculation. Finally, data analysis is conducted to assess the performance of each model.

A. The Jetbot Platform

This paper aims to leverage the components of the open-source JetBot scale model car provided by Nvidia, including the front camera, Jetson Nano motherboard, JetBot driver board, and DC motor. As shown in Fig. 1. The Jetson Nano is chosen as the embedded computing platform due to its quad-core Cortex-A57 processor, 128-core Maxwell GPU, and 4GB LPDDR memory, which offer substantial AI computing power and facilitate the execution of various AI algorithms. This paper evaluates depth models that will be deployed on the Jetson Nano for real-time inference, demonstrating the capability of the system to perform complex AI tasks.

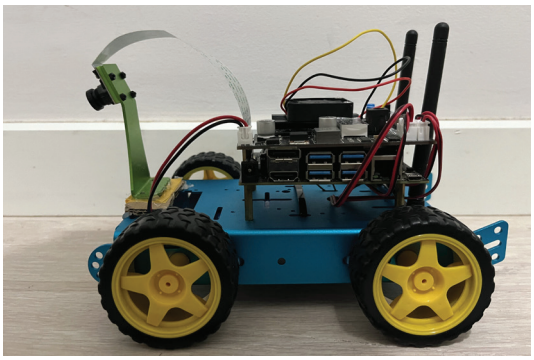

Fig. 1. The scale model car

B. Data Collection and Parameterization

To assess the performance of each model in real-world scenarios, we designed the track illustrated in Fig. 2. The track includes straight lines, curves, T-junctions, intersections, and various obstacles and traffic signs. We collected tracking data along the outer track while using the inner track as an unknown scenario to evaluate the adaptability of the model to unseen environments. For detailed information on data collection, please refer to Table I.

TABLE I
COMPOSITION OF THE DATASET

Dataset	Numbers
Road Tracking	600
Turn Right/Left	500
Barrier Avoidance Cars	900
Total	**2000**

Images of the track are captured through the camera of a model car, where each image contains X and Y coordinate labels that represent the location of the vehicle. Our goal is to train a neural network model to accurately predict the values of the X and Y

coordinates from these images. This prediction can provide us with information about the position of the vehicle on the track.

Barrier Avoidance Cars (BAC) are vehicles designed to be used on tracks to provide obstacle avoidance support for self-driving cars. These vehicles are typically placed on specific test tracks to simulate various obstacles or challenges encountered during real-world road travel. As the self-driving vehicle travels on the test track, BACs are placed in its path to act as potential obstacles. The self-driving vehicle uses its camera system to detect the BACs and successfully avoids these obstacles by performing appropriate steering maneuvers.

Before training the data, we need to preprocess the data to improve the generalization ability and performance of the model. First, we can process the image using a random horizontal flip technique where the flip probability is set to 0.5, which helps the model to learn more features from different angles. Next, we can apply color jitter to the image, where the jitter magnitude for each value is set to 0.3, which helps to increase the diversity of the data and makes the model more robust to lighting and color changes. We then resized the image to 224 × 224 pixels, the input size for many standard convolutional neural network models. We then converted the image to a tensor (tensor) format to be processed in the neural network. Finally, we normalize the image by scaling the pixel values to a specific range to ensure stability and convergence of the model training.

To test the performance of the model, we separated the test track from the track for collecting data. We designed the track shown in Fig. 3 as the test track. This test track was carefully designed to introduce scenarios not previously encountered by the model to assess the robustness and generalization ability of the model more fully.

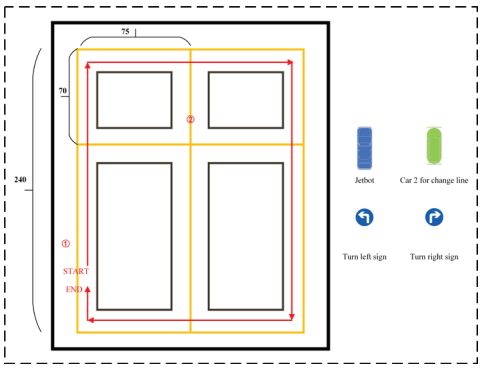

Fig. 2. Map track

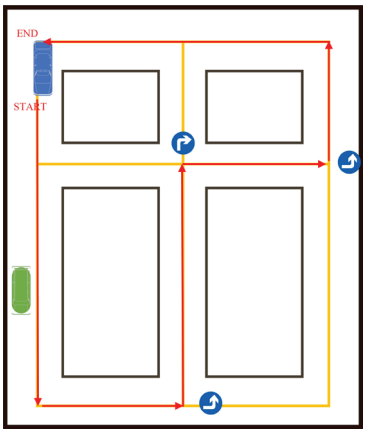

Fig. 3. Test track

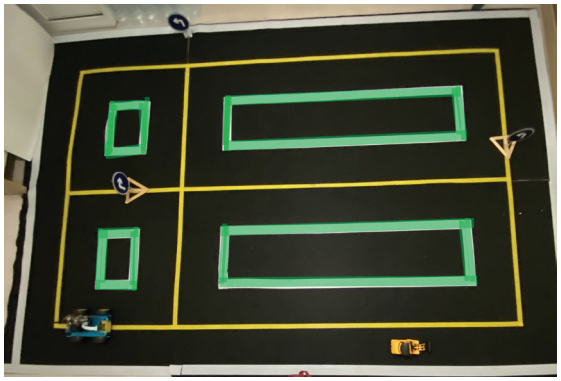

Fig. 4. Actual test track

In this experiment, the data were trained through the colab platform, and the experimental parameters were set to the epoch of 80, batch size of 8, Adam as an optimizer, and learning rate of 0.0001.

C. *Criteria for Assessment*

To comprehensively evaluate each model's performance on actual roads, we introduce a novel evaluation metric known as the Road Integrity Comprehensive Score (RICS). RICS combines two critical factors: Road Distance Completion (RDC) and Road Sign Completion (RSC). Road Distance Completion is the ratio between the distance the vehicle covers and the total route distance. Road Sign Completion is the ratio between accurately detected road traffic signs and the total number of signs.

$$\text{RICS} = \frac{1}{\text{N}_c}\sum_{i=1}^{N_c}\left(RDC_i \times RSC_i\right) \quad (1)$$

N_c : Number of Test Lap
i : -th route
RDC_i: The percentage of completion of the route
RSC_i : The infraction penalty for a route

This method provides a more detailed and comprehensive evaluation approach by considering both the coverage distance of the vehicle and the accurate identification of traffic signs. It not only accounts for the route coverage of the car but also assesses the accuracy of the model in identifying crucial traffic information. Therefore, the Road Integrity Comprehensive Score offers a more precise measure of the real-world performance and reliability of each model in actual road environments.

IV. Result

Models with fewer layers, like VGG and AlexNet, have higher sizes, as Table II demonstrates. When these models are put on embedded devices, latency concerns may arise. Although SqueezeNet has a smaller model size, its corresponding loss values are higher, resulting in suboptimal performance on the model car. When comparing ResNet18, ResNet34, and ResNet50, an layer increase directly impacts model size, while the change in loss values remains marginal. The MT-26 model is an extension of ResNet18 to 26 layers.

TABLE II
Results of Model Training

Model	Epoch	Train loss	Test Loss	Model Size(MB)
Alexnet	10	0.231967	0.224831	217
Vgg16	1	1.876685	0.200343	512
Squeezenet	37	0.228815	0.221599	2.77
Mobilenet	57	0.003410	0.003572	8.73
Densenet201	46	0.002949	0.001997	70.30
Resnet18	52	0.002777	0.001751	42.70
Resnet34	60	0.002566	0.002466	81.30
Resnet50	62	0.003016	0.001323	89.90
MT-26	79	0.003338	0.001247	90.60
Efficientnet _b0_sa	73	0.001838	0.001106	15.90
Resnet-CBAM	75	0.001598	0.001516	43.00

Despite having the largest model size, it has the lowest loss value among the ResNet series. On the other hand, ResNet-CBAM is an enhancement of ResNet18 with an attention mechanism that prioritizes significant image features during training. This yields superior performance without significantly increasing model size, with only a 0.3MB increase.

The real-world performance scores of several models are shown in Table III. It is observed that shallow networks such as AlexNet, VGG, and SqueezeNet face difficulties in achieving real-time corner predictions on embedded systems, which hinders their ability to control the model car for road tracking and sign recognition. On the other hand, MobileNet, a lightweight network at 8.73MB, achieves an RICS score of 0.81, ranking second among all tested models and showcasing commendable practical applicability. Despite its larger model size compared to MobileNet, MT-26 is ranked first. It achieves real-time execution on embedded systems with an RICS score of 1, effectively controlling the model car for road tracking and sign recognition. Other models achieve an average RICS score of 0.57, indicating their operability on model cars.

TABLE III
Tracking Test Results

Model	RICS	RDC	RSC
Alexnet	0.08	0.31	0.25
Vgg16	0.08	0.31	0.25
Squeezenet	0.00	0.31	0.00
Mobilenet	0.81	0.81	1.00
Densenet201	0.61	0.81	0.75
Resnet18	0.54	0.72	0.75
Resnet34	0.54	0.72	0.75
Resnet50	0.08	0.31	0.25
MT-26	1.00	1.00	1.00
Resnet-CBAM	0.54	0.72	0.75
Efficientnet _b0_ca	0.61	0.81	0.75
Ideal	1.00	1.00	1.00

Finally, we combine Tables II and III to analyze the performance of each model in this task. Table II shows that AlexNet, VGG16, and SqueezeNet achieved their optimal models at epoch 10, epoch 1, and epoch 31, respectively. However, their loss values are relatively high compared to other models. Although they can be deployed and run on Jetson Nano, through Table III, we found their final RICS scores to be low, all at 0.08, indicating that they failed to learn the dataset information thoroughly, and their models are too large, making running on Jetson Nano challenging.

In contrast, MobileNet and MT-26 achieved their optimal models at epoch 57 and epoch 79, respectively, in Table II. Their loss values have significantly decreased compared to AlexNet, VGG16, and SqueezeNet. By observing Table III, we can see that the final RICS scores for MobileNet and MT-26 are 0.81 and 1, respectively, indicating that they are better suited for this task. This is because their loss values have decreased significantly during the training process, and the final RICS scores indicate that they have learned the dataset features better. Additionally, their models are relatively lighter, making them more suitable for running in resource-constrained environments, such as Jetson Nano.

V. Conclusion

This paper presents an overview of steering angle prediction algorithms for autonomous driving. Our study involved reproducing and testing existing models in real environments using model cars. We found that models with fewer layers, such as AlexNet, VGG16, and SqueezeNet, could not fully comprehend the information in the training data. As a result, they could not perform autonomous driving tasks in embedded systems. ResNet50 achieved an RICS of only 0.08 in the ResNet series, while ResNet18 and ResNet34 scored 0.54. ResNet-CBAM and EfficientNet_B0_CA exhibited similar performance to the ResNet series. MobileNet, a lightweight network, attained an RICS of 0.81 in the embedded system scale model car, ranking second among all the tested models. The top-performing model was MT-26, achieving an RICS of 1 and ranking first despite having a more significant model size than MobileNet.

Acknowledgment

The first author experimented and drafted the manuscript. The last author guided and advised the experiment and co-drafted the manuscript. The first and last authors each contributed 50% equally to this work. The last author is the corresponding author.

The first author received scholarship support from CPALL for conducting this research in PIM.

References

[1] D. A. Pomerleau, "Alvinn: An Autonomous Land Vehicle in a Neural Network," in *Proc. NIPS*, 1988, pp. 305-313.

[2] C. Chen, A. Seff, A. Kornhauser et al., "Deep Driving: Learning Affordance for Direct Perception in Autonomous Driving," in *Proc. ICCV*, 2015, pp. 2722-2730.

[3] M. Bojarski, D. D. Testa, D. Dworakowski et al. (2016, Apr. 25). *End to End Learning for Self-Driving Cars*. [Online]. Available: https://arxiv.org/abs/1604.07316

[4] M. Bojarski, D. D. Testa, D. Dworakowski et al. (2017, Apr. 25). *Explaining How a Deep Neural Network Trained with End-to-End Larning Steers a Car*. [Online]. Available: https://arxiv.org/abs/1604.07316

[5] V. Rausch, A. Hansen, E. Solowjow et al., "Learning a Deep Neural Net Policy for End-to-End Control of Autonomous Vehicles," in *Proc. ACC*, 2017, pp. 4914-4919.

[6] T. D. Do, M. T. Duong, Q. V. Dang et al., "Real-Time Self-Driving Car Navigation Using Deep Neural Network," in *Proc. GTSD*, 2018, pp. 7-12.

[7] Z. Chen and J. Huang, "End-to-End Learning for Lane Keeping of Self-Driving Cars," in *Proc. IV*, 2017, pp. 1856-1860.

[8] H. M. Eraqi, M. N. Moustafa, and J. Honer. (2017, Oct. 10). *End-to-End Deep Learning for Steering Autonomous Vehicles Considering Temporal Dependencies*. [Online]. Available: https://arxiv.org/abs/1710.03804

[9] J. Jhung, I. Bae, J. Moon et al., "End-to-End Steering Controller with CNN-Based Closed-Loop Feedback for Autonomous Vehicles," in *Proc. IVS*, 2018, pp. 617-622.

[10] S. Sharma, G. Tewolde, and J. Kwon, "Behavioral Cloning for Lateral Motion Control of Autonomous Vehicles Using Deep Learning," in *Proc. EIT*, 2018, pp. 0228-0233.

[11] Z. Yang, Y. Zhang, J. Yu et al., "End-to-End Multi-Modal Multi-Task Vehicle Control for Self-Driving Cars with Visual Perceptions," in *Proc. ICPR*, pp. 2289-2294.

[12] M. V. Smolyakov, A. I. Frolov, V. N. Volkov et al., "Self-Driving Car Steering Angle Prediction Based on Deep Neural Network an Example of CarND Udacity Simulator," in *Proc. AICT*, 2018, pp. 1-5.

[13] M. G. Bechfel, E. McEllhiney, M. Kim et al. "Deep Picar: A Low-Cost Deep Neveal Network-Based Autonomous on Embedded and Real-Time Compution Systems and Applications (RTCSA), 2018, pp. 11-21.

[14] D. Wang, J. Wen, Y. Wang et al., "End-to-End Self-Driving Using Deep Neural Networks with Multi-auxiliary Tasks," *Automot. Innov.*, vol. 2, no. 2, pp. 127-136, May. 2019.

[15] S. Du, H. Guo, and A. Simpson. (2019, Dec. 11). *Self-Driving Car Steering Angle Prediction Based on Image Recognition*. [Online]. Available: https://arxiv.org/abs/1912.05440

[16] M. J. Lee and Y. G. Ha, "Autonomous Driving Control Using End-to-End Deep Learning," in *Proc. BigComp*, 2020, pp. 470-473.

[17] A. Kumar and S. Palaniswamy, "Steering Angle Estimation for Self-Driving Car using Deep Learning," *MLMHA*, vol. 1203, pp. 196-207, Apr. 2020.

[18] A. K. Mishra, "Deep Learning for Steering Angle Prediction in Autonomous Vehicles," *MSEA*, vol. 70, no. 2, pp. 1584-1590, Dec. 2021.

[19] J. Sokipriala, "Prediction of Steering Angle for Autonomous Vehicles Using Pre-Trained Neural Network," *EJETR*, vol. 6, no. 5, pp. 171-176, Aug. 2021.

[20] M. I. H. Showrov, M. R. Islam, M. A. Amin et al., "Comparative Analysis of Steering Angle Prediction for Automated Object Using Deep Neural Network," in *Proc. ICRITO*, 2021, pp. 1-7.

[21] K. Podbucki and T. Marciniak, "Aspects of Autonomous Drive Control Using NVIDIA Jetson Nano Microcomputer," in *Proc. CCSIS*, 2022, pp. 117-120.

[22] Y. G. Khidhir and A. H. Morad, "Comparative Transfer Learning Models for End-to-End Self-Driving Car," *AKEJ*, vol. 18, no. 4, pp. 45-59, Apr. 2022.

[23] I. U. Hassan, H. Zia, H. S. Fatima et al., "A Lightweight Convolutional Neural Network to Predict Steering Angle for Autonomous Driving Using CARLA Simulator," *MSE*, vol. 2022, no. 9, pp. 1-11, Aug. 2022.

[24] S. Ding and J. Qu, "Smart Car with Road Tracking and Obstacle Avoidance Based on Resnet18-CBAM," in *Proc. ICBIR*, 2022, pp. 582-585.

[25] Z. Nie and J. Qu, "Multi-Task Autonomous Driving Based on Improved Convolutional Neural Network and ST Loss in MTS and MOD Modes," *CAST*, vol. 23, no. 3, pp. 10-36, Nov. 2023.

[26] S. Ding and J. Qu, "Research on Multi-Tasking Smart Cars Based on Autonomous Driving Systems," *SNCS*, vol. 4, no. 3, p. 292, Mar. 2023.

[27] S. Ding and J. Qu, "Automatic Driving for Road Tracking and Traffic Sign Recognition," *STA*, vol. 27, no. 4, pp. 343-362, Dec. 2022.

[28] Y. Li and J. Qu. (2024, Apr. 12). *A Novel Neural Network Architecture and Cross-Model Transfer Learning for Multi-Task Autonomous Driving*. [Online]. Available: https://doi.org/10.1108/DTA-08-2022-0307

[29] J. Qu and S. Shi, "Multi-Task in Autonomous Driving through RDNet18-CA with LiSHTL-S Loss Function," *ECTI-CIT*, vol. 18, no. 2, pp. 158-173, Apr. 2024.

Shang Shi is studying for the Master of Engineering Technology, Faculty of Engineering and Technology, Panyapiwat Institute of Management, Thailand. He received a B.B.A from Nanjing Tech University Pujiang Institute, China, in 2022. His research interests are Research direction is artificial intelligence, image processing, and autonomous driving.

Jian Qu is an Assistant Professor at the Faculty of Engineering and Technology, Panyapiwat Institute of Management. He received Ph.D. with Outstanding Performance award from Japan Advanced Institute of Science and Technology, Japan, in 2013. He received a B.B.A with Summa Cum Laude honors from the Institute of International Studies of Ramkhamhaeng University, Thailand, in 2006, and M.S.I.T from Sirindhorn International Institute of Technology, Thammast University, Thailand, in 2010. He has been serving as a house committee for the Thai SUPERAI project since 2020. His research interests are natural language processing, intelligent algorithms, machine learning, machine translation, information retrieval, image processing, and autonomous driving.