

A Study of Road Tracking Immunity for Autonomous Driving

Chuanxiang Bi¹ and Jian Qu^{2*}

^{1,2}Faculty of Engineering and Technology, Panyapiwat Institute of Management,
Nonthaburi, Thailand

E-mail: 6572100057@stu.pim.ac.th, jianqu@pim.ac.th*

Received: August 15, 2025 / Revised: October 12, 2025 / Accepted: November 10, 2025

Abstract—With the advancement of deep learning research, autonomous driving technology has become increasingly mature. However, the accompanying issue is that environmental interference may pose safety hazards for autonomous driving, presenting a significant threat. Therefore, this paper aims to evaluate the resilience of autonomous vehicles to environmental interference by studying the fundamental road-tracking task of autonomous driving. Following the model of real autonomous driving vehicles, we constructed a 1:20-scale intelligent model car, Jetracer, and used it to simulate the road-tracking task of autonomous driving. We designed four different environments, first collecting varying numbers of image data in the original environment for comparative analysis. Then, we selected ResNet18 and ResNet34 as models for training and loaded them onto Jetracer for testing in the four different environments. The experimental results indicate that as the number of images in the dataset increases, the effectiveness of road tracking also gradually improves. Meanwhile, we found that ResNet18 is more suitable as a training model compared to ResNet34. Additionally, Jetracer demonstrates a certain degree of interference resistance, where slight or localized environmental changes do not significantly affect its road tracking performance. However, if there are significant overall environmental changes or new environments are introduced, Jetracer's road-tracking performance is severely impacted.

Index Terms—Deep Learning, Autonomous Driving, Environmental Interference, Road-Tracking, ResNet18, ResNet34, Jetracer

I. INTRODUCTION

Automatic driving [1], [2] technology has been a key area of automotive innovation in recent years. However, intelligent vehicles capable of multitasking face challenges in handling various complex road conditions. Among these challenges, road tracking

is considered a primary task for autonomous driving. Cosmas et al. [3] employed a laser-based approach for road boundary detection, achieving road tracking by spatially processing data from laser sensors. Meanwhile, Kumar et al. [4] and Li and Qu [5] proposed traditional lane-keeping visual methods, detecting lanes through image processing to analyze the deviation of intelligent vehicles from the road, ensuring that vehicles travel on the correct trajectory. In addition, various other methods are available. Qi et al. [6] utilized reinforcement learning to train intelligent vehicles in virtual environments and then tested them in real environments. While experimental results in virtual environments show promising prospects, successfully translating them to real environments faces challenges. Real-world environments are dynamic and constantly changing, influenced by factors such as lighting, shadows, and other sources of noise, which are often insufficiently accounted for in virtual environments.

To address this, Peng et al. [7] proposed an end-to-end road-tracking architecture based on deep reinforcement learning and evaluated this method in an open racing simulator. Although some success has been achieved in road tracking through the above studies, the road tracking task still faces many challenges due to the variable and complex road conditions that the intelligent car is facing, such as artificial graffiti lines on the road or placing obstacles, so in this paper, we train the intelligent car in a real environment that has already been constructed, and then through the intentional change of the real environment, which is divided into six environments to study the smart car's resistance to different road disturbances in order to continuously improve its ability to adapt to various road conditions.

II. LITERATURE REVIEW

Current research indicates that many intelligent vehicles still rely on externally connected computers and cannot independently complete tasks involving deep learning or reinforcement learning, thus limiting the autonomy of artificial intelligence. For instance,

Yunyong and Qu [8] utilized Arduino as the mainboard for the vehicle to execute reinforcement learning tasks. However, due to Arduino's limited performance, it cannot handle complex computational problems, necessitating a Bluetooth connection to a computer as the backend. Arduino, as a microcontroller-based hardware platform, can only execute programs preprogrammed into the chip and is therefore only suitable for simple scenarios. To make intelligent vehicles an independent unit, some researchers have turned to microprocessor-based hardware platforms, such as the Raspberry Pi, which can handle complex applications like network communication and image processing. For example, Du et al. [9] implemented road-tracking for intelligent vehicles using cameras and radar. Although many studies use the Raspberry Pi as a development board, we found that the deep neural networks in these studies typically have fewer layers and simpler architectures. Therefore, in this work, we chose the Jetson Nano as the mainboard to enable intelligent vehicles to perform tasks independently. For example, Karni et al. [10] and Bae et al. [11] have explored the possibilities of using Jetson Nano. Although research using toy cars is cost-effective, it is difficult to replicate real-world driving conditions due to significant differences between toy cars and real vehicles. In Karni's study and that of Lee and Lam [12], researchers combined cameras and ultrasonic sensors. Meanwhile, Du et al. [9] and Zang et al. [13] utilized various cameras and LiDAR. Despite the adoption of multiple sensors in current research, human drivers typically rely only on vision and hearing to operate vehicles. Therefore, this paper selected a 1:20-scale intelligent car and combined it with the Jetson Nano [14] as the mainboard to investigate the road-tracking task for autonomous driving. The intelligent car's steering and driving control capabilities can, to some extent, simulate the driving systems of real vehicles.

The end-to-end control [15], [16] method is a typical approach in the field of vision-based autonomous driving. In this method, vehicles directly collect sensor data on pedestrians, obstacles, and traffic lights, and then uniformly input these signals into a neural network. The output of the neural network system is closely related to vehicle control, guiding the vehicle's next actions. As early as 1989, Pomerleau utilized neural networks to predict steering commands based on simulated road images. In 2016, NVIDIA unveiled its end-to-end deep learning technology for autonomous vehicles. This technology trains Convolutional Neural Networks (CNNs) to directly map the raw pixels from a single front-facing camera to steering commands. The end-to-end control system can optimize all processing steps, thus having wide-ranging application prospects.

In this paper, the intelligent vehicle we used adopts the end-to-end control method.

Convolutional Neural Networks [17] (ConvNets or CNNs) have emerged as one of the most commonly used neural network architectures in autonomous driving, rooted in the concept of Multilayer Perceptrons (MLPs) and inspired by Hubel and Wiesel's studies of the visual cortex in cats. CNNs offer optimization advantages by reducing the number of parameters, lowering model complexity, and mitigating overfitting risks, particularly evident when processing image inputs. CNNs directly ingest image data, eliminating the need for manual feature extraction and data reconstruction processes inherent in traditional recognition algorithms. Specifically adept at handling two-dimensional images, CNNs autonomously extract features such as color, texture, and shape, granting them a significant advantage in two-dimensional image processing and computational efficiency. In autonomous driving, CNNs are widely used for tasks such as road tracking. For instance, Rausch et al. [15] proposed a CNN model comprising three convolutional layers, two pooling layers, and one fully connected layer, whereas Lin et al. [18] presented a more complex model with five convolutional layers and four fully connected layers. These models underwent testing in computer-simulated virtual environments, successfully achieving road tracking, albeit with disparities compared to real-world conditions. Further research utilizes miniature intelligent driving platforms and custom-built tracks for testing. Studies by Bechte et al. [19] and Do et al. [20] also employ similar CNN models, effectively achieving road tracking on custom-built tracks. Consequently, Convolutional Neural Networks play a pivotal role in autonomous driving, and this research provides insights into their key architectural components, proposes end-to-end autonomous driving methodologies, and discusses road-tracking implementation strategies.

III. METHODOLOGY

A. Hardware Designs

This paper selects Jetracer as the research subject to closely simulate real-world driving scenarios. Unlike many toy cars used by researchers, Jetracer features adjustable parameters that enhance vehicle performance. In contrast to toy cars that can only accept binary signals (1 or 0), Jetracer's design permits adjustments to parameters such as steering gain, thereby preventing deviation from the normal trajectory. Unlike other sensors that solely rely on a single camera to perceive the surrounding environment, Jetracer, akin to human drivers, utilizes both vision and hearing.

After comparing three embedded mainboards, Arduino, Raspberry Pi, and Jetson Nano [2], [21], this paper ultimately selects Jetson Nano as the host board for Jetracer. As shown in Fig. 1 and Table I, the Jetson Nano boasts powerful performance and rich functionality, enabling it to connect to cameras and control their operation. The camera serves as the information-perception sensor, responsible for collecting and transmitting data to the Jetson Nano. The control board acts as the command transmission medium, relaying instructions from Jetson Nano to the motors to control the vehicle's propulsion. With this design, we establish an autonomous driving platform using the Jetracer hardware architecture and the hardware framework illustrated in Fig. 2.

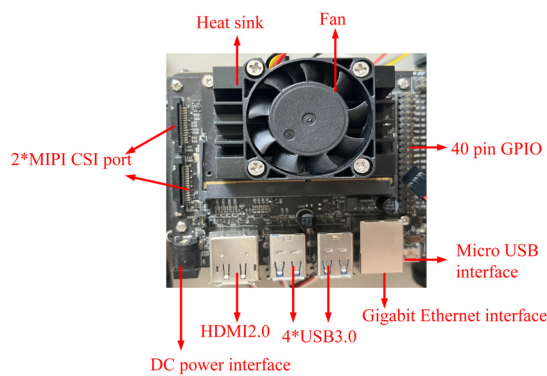


Fig. 1. Jetson Nano construction diagram

TABLE I
JETSON NANO SPECIFIC INFORMATION

Project	NVIDIA Jetson Nano
CPU	Quad-Core ARM Cortex-A57
GPU	NVIDIA Maxwell
Memory	4GB LPDDR4
Networking	Gigabit Ethernet/M.2Key E
Video Encode	H.264/H.265(4Kp30)
Camera	MIPI CSI port
USB	4*USB3.0,USB2.0 Micro-B
Display	HDMI 2.0 and eDPI.4
Other	40-pin GPIO
Storage	Micro-SD

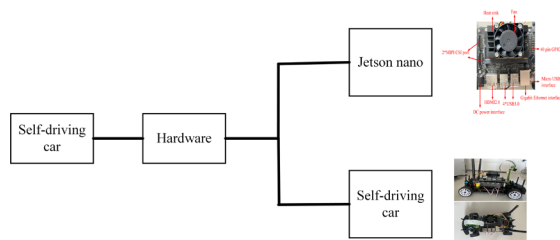


Fig. 2. Jetracer hardware framework diagram

B. Environment Construction

Regarding the driving environment of the intelligent vehicle, we utilize a designed map. Environment 1, as depicted in Fig. 3, is a scenario we designed. It consists of an elliptical track, with yellow lines representing the road edges and white dashed lines representing the lane centerlines. The main objectives of this environment are twofold: data collection and validation of the intelligent vehicle's road-tracking capabilities in the training environment.



Fig. 3. Environment 1

Environment 2 (as shown in Fig. 4): We replaced the white dotted middle lane line with a solid black line based on Environment 1, taking into account that the road environment changes the middle lane line color, to test whether a change in the middle lane line makes the smart car less capable of performing the road tracking task.



Fig. 4. Environment 2

Environment 3 (as shown in Fig. 5): We unfolded two additional curved paths on the white dotted line in the middle into a three-way intersection on one of the road segments in Environment 1, considering whether someone maliciously modified the lane lines, to test whether it would interfere with the normal driving of the smart car.



Fig. 5. Environment 3

Environment 4 (as shown in Fig 6): We built on Environment 1 by paving the road with stickers of various lengths and colors, considering whether a crash occurs when the road is full of obstacles. The aim is to test whether the smart car will not be able to perform the task due to a drastic change in the environment.



Fig. 6. Environment 4

Environment 5 (as shown in Fig. 7): We drew an identical median next to the lane median based on Environment 1, with the aim of testing whether the smart car would sway from side to side due to the two lane medians.



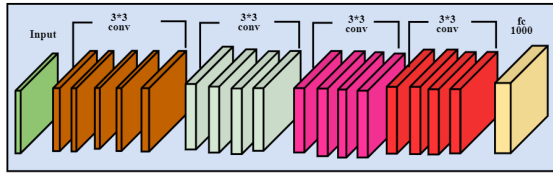
Fig. 7. Environment 5

Environment 6 (as shown in Fig. 8): We change the original lane median to yellow on the basis of Environment 5, with the purpose of testing whether the intelligent vehicle will change its path because the original lane median is changed.



Fig. 8. Environment 6

Regarding the models used in the research, most existing studies have employed simple Convolutional Neural Network (CNN) models. For example, Victor et al. [22], used a 6-layer CNN neural network in their study, while Truong-Dong Do et al. [23] utilized a nine-layer CNN neural network. However, shallow neural networks may encounter two problems: Insufficient feature extraction and the need for large image datasets. However, He et al. proposed ResNet18 in 2016 [24] (as shown in Fig. 9). They significantly increased the network depth by introducing residual connections, thereby alleviating network degradation. ResNet is a deep residual network that effectively avoids the vanishing gradient problem by skipping layers via residual connections, enabling it to train very deep structures. ResNet uses a large number of residual blocks as the base blocks. The design of ResNet18 consists of a 7x7 convolutional layer, followed by a 3x3 max-pooling layer for feature extraction. Subsequently, multiple residual blocks of the same structure are used for feature extraction, each containing two 3x3 convolutional layers and using residual concatenation. ResNet18 contains a total of four composite blocks, with the number of channels in the convolutional layers within each block increasing in the order 64, 64, 128, and 256, and the number of channels varied by cross-scaling the residual blocks. Finally, the average pooling layer and softmax classifier are picked up. ResNet18 has fewer but more significant parameters and possesses strong representation-learning capability. At the same time, the computational overhead is relatively small and well-suited to the task at hand. Therefore, we chose ResNet18 as our model.



ResNet18

Fig. 9. ResNet18 construction diagram

IV. EXPERIMENTAL SETUP

A. Dataset Collection

Data collection is a crucial step in intelligent vehicle research. We utilize the intelligent vehicle to traverse along the map and capture images for collecting image data. To ensure data quality, we close curtains and doors to maintain stable lighting conditions, preventing sunlight from affecting the environment and reducing sunlight noise. All data are collected and tested under such lighting conditions. The neural networks used in this paper take images with a resolution of 224x224 pixels as input.

B. Model Training

During the model training phase, we leverage the Colab platform's powerful computational resources to train efficiently. By configuring an environment that meets deep learning requirements, we train the model on the Colab platform. The environment is configured as in Table II. We divide the collected dataset into two parts: the training set and the test set. The training set comprises 80% of the dataset and is used to train the model, while the test set comprises 20% and is used to validate the model's accuracy.

TABLE II
ENVIRONMENT CONFIGURATION

Environment Configuration				
Request	TensorRT	Pytorch	CUDA	CUDNN
Version	6.0.1.10	1.4.0	10.0.326	7.6.3.28

C. Loss functions

Loss functions are used in deep learning to measure the difference between a model's predictions and the actual values. Since this paper implements a simple road-tracking task, it uses an existing cross-entropy loss function rather than designing a new one. In this paper, we choose the Mean Squared Error (MSE). MSE is a common loss function because it requires no parameters, is computationally efficient, and has a clear physical meaning. Its formula is shown below.

$$MSE = \frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2 \quad (1)$$

We employed a consistent hyperparameter configuration scheme for training all models. Considering the memory constraints of the embedded

device's mainboard, we selected appropriate mini-batch sizes and mini-epochs within the Jetson Nano's memory allowance for model training. Here, an epoch refers to the number of times all data are processed during training. Our study aimed to identify the minimum loss value and the most efficient training epochs. We implemented an early stopping strategy during model training. When the training loss ceases to decrease, we conduct an additional five training iterations. If the training loss does not decrease after these five iterations, training is automatically terminated. Furthermore, we opted for the widely adopted Adam optimizer due to its simplicity, computational efficiency, and relatively modest memory requirements.

In our experiments, the model car's speed is fixed. This simplifies the control logic, which only needs to adjust the steering angle. During training, the model learns to keep the vehicle in the center of the road by predicting the appropriate steering angle based on the input image. The trained model is deployed to an autonomous driving platform and tested and validated in a real-world environment. The steering angle output from the model will be used to control the vehicle's direction when the vehicle speed is fixed. By processing camera input images in real time, the model can continuously output the steering angle to maintain road tracking and drive steadily.

D. Experiment

In Environment 1, we captured 400, 600, 800, and 1000 images for road tracking using the intelligent vehicle, naming these four datasets Dataset A, Dataset B, Dataset C, and Dataset D. We trained ResNet18 models separately on each dataset. Subsequently, we employed the trained models to perform road-tracking tasks in Environments 1, 2, 3, and 4, recording data to assess the models' resistance to interference. Due to the limited number of images in the collected dataset, setting the epoch count too low may result in underfitting, while setting it too high may lead to overfitting. Therefore, we conducted experiments with several parameter combinations and determined to utilize an epoch of 40 and a batch size of 16 based on our evaluations. As shown in Table III, subsequent training sessions were also conducted using this configuration.

TABLE III
HYPERPARAMETER SETTING

Hyperparameter		
Optimizer	Batch Size	Epoch
Adam	16	40

For comparison, we selected ResNet34 as another model. Similarly, we trained ResNet34 models with the aforementioned four datasets individually and

utilized the trained models to execute road-tracking tasks in Environments 1, 2, 3, and 4. This was done to compare the road-tracking performance with the previous ResNet18 models. A diagram of the experimental framework is shown in Fig. 10.

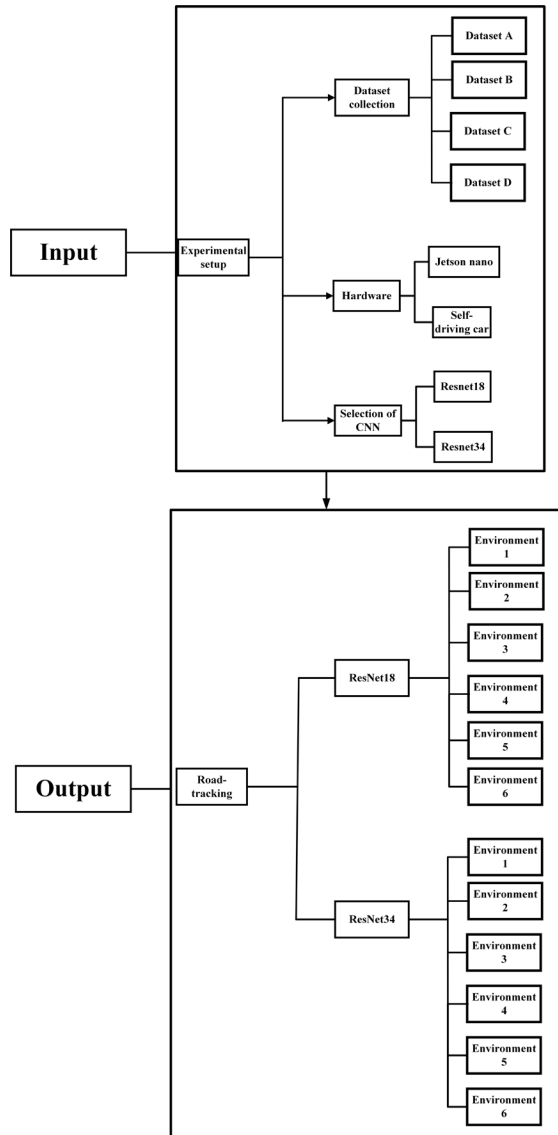


Fig. 10. Experimental framework diagram

V. RESULT

A. Evaluation Methodology

To evaluate the intelligent vehicle's road-tracking performance, we propose a new method for assessing autonomous driving. The vehicle drives five laps in the environment, with each completed lap earning 2 points, for a total score of 10. The scoring rules are as follows:

(1) In each lap, if the intelligent vehicle touches the inner or outer lane line once, one point is deducted; if it touches more than twice, the score for that lap is zero.

(2) If the intelligent vehicle runs off the map boundary during a lap, the score for that lap is zero, and subsequent lap scores are also zero.

The total score can be calculated by adding up the scores from the five laps.

B. Discussions

Based on the data in Table IV, we observe the following: as the number of images in the dataset increases, the intelligent vehicle's score in Environment 1 gradually improves. This indicates that as data volume increases, the intelligent vehicle's performance in road-tracking tasks becomes more stable. Environment 2 has a relatively minor impact on the intelligent vehicle's road tracking, but it does interfere with the vehicle's travel, making it more prone to touching the inner and outer lane lines. The influence of Environment 3 on the road-tracking of the intelligent vehicle is almost negligible, indicating that changing the center lane line has minimal impact on the vehicle. In Environment 4, the intelligent vehicle's score is the lowest, indicating that its performance on road-tracking tasks is the worst. In most cases, the intelligent vehicle is directly disturbed and runs off the map, making it unable to continue the task. This suggests that creating large-scale disturbances on the ground can cause autonomous driving to fail at road-tracking tasks.

TABLE IV
TABLE ON ROAD-TRACKING SCORES FOR EACH OF
THE FOUR MODELS OF RESNET18 FOR EACH OF
THE FOUR ENVIRONMENTS

Dataset	Quantities	Model	Environment	score
Dataset A	400	ResNet18	Environment 1	5
			Environment 2	3
			Environment 3	4
			Environment 4	0
			Environment 5	6
			Environment 6	2
Dataset B	600	ResNet18	Environment 1	7
			Environment 2	5
			Environment 3	6
			Environment 4	0
			Environment 5	7
			Environment 6	4
Dataset C	800	ResNet18	Environment 1	10
			Environment 2	8
			Environment 3	9
			Environment 4	2
			Environment 5	10
			Environment 6	6
Dataset D	1000	ResNet18	Environment 1	10
			Environment 2	9
			Environment 3	10
			Environment 4	2
			Environment 5	10
			Environment 6	6

Based on the training and testing results of the ResNet34 model, compared with the previous use of the ResNet18 model, as shown in Table II, we found that the road tracking scores across all environments were lower than those of the ResNet18 model. Since ResNet34 is more complex than ResNet18 and contains more layers. However, in the simple road-tracking task in this paper, this extra complexity does not improve performance and may even lead to overfitting. In this case, it may not lead to a significant performance improvement; instead, it may cause training difficulties and reduced generalisation due to an overparameterised model. Overfitting occurs when the model performs well on training data but poorly on test data. Whereas the simpler model, ResNet18, may be sufficient to express the information needed for the task. Suggesting that in a specific task (road tracking), ResNet18 may perform better due to its lower complexity, faster inference, and better generalization.

This suggests that ResNet18 is more suitable than ResNet34 for this autonomous driving task. However, as with the ResNet18 model, the scores for environments 2 and 3 did not differ much from those for environment 1. This suggests that these environments have relatively little effect on the performance of the ResNet34 model. In contrast, in environment 4, the ResNet34 model also exhibits a lower score profile, indicating that this environment has a greater impact on the road-tracking task, consistent with previous observations.

TABLE V
THE TABLE ON ROAD-TRACKING SCORES FOR EACH OF
THE FOUR MODELS OF RESNET34 FOR EACH OF
THE FOUR ENVIRONMENTS

Dataset	Quantities	Model	Environment	Score
Dataset A	400	ResNet34	Environment 1	3
			Environment 2	0
			Environment 3	2
			Environment 4	0
			Environment 5	3
			Environment 6	1
Dataset B	600	ResNet34	Environment 1	6
			Environment 2	4
			Environment 3	6
			Environment 4	0
			Environment 5	8
			Environment 6	4
Dataset C	800	ResNet34	Environment 1	8
			Environment 2	5
			Environment 3	6
			Environment 4	1
			Environment 5	8
			Environment 6	6
Dataset D	1000	ResNet34	Environment 1	10
			Environment 2	7
			Environment 3	10
			Environment 4	2
			Environment 5	10
			Environment 6	7

Regarding the reasons why the experimental model car above crashed into the outer or inner lane, we conclude the following.

(1) In terms of image processing, noise in the image (shadows, halos) may interfere with the model's judgment, leading to incorrect lane tracking. Low-resolution images may also fail to capture details, resulting in the model's inability to accurately identify lane lines and other road features.

(2) In terms of environmental factors and external interferences, this paper sets up road interferences, such as lane lines of other colours and obstacles of different lengths on the road, and the training data does not contain sufficiently diverse map data, which may prevent the model from accurately identifying them.

(3) In terms of hardware processing capability, the model car's ability to process images in real time is not very strong, and processing delays may lead to incorrect judgments.

(4) In terms of sensor and hardware limitations, a single sensor (e.g., using only a camera) may not be sufficient to cope with complex environments, and a combination of multiple sensors is needed to improve accuracy.

VI. CONCLUSION

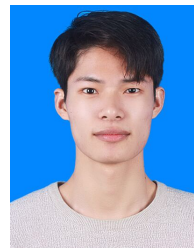
Achieving road tracking is the most fundamental and critical task in autonomous driving, and it is also the focus of the current experiments in this paper. By establishing an autonomous driving platform - Jetracer, to replace real autonomous driving vehicles, we have successfully conducted research on the road tracking task. Our experimental results demonstrate that the intelligent vehicle can perform road-tracking tasks using a single camera. In addition, we conducted three experiments: first, varying the degree of environmental interference; second, comparing the performance of different models; and third, analysing the impact of the number of images in the dataset on road-tracking effectiveness. The results show that the effectiveness of road tracking improves as the number of images in the training dataset increases. However, once the number of images reaches a certain threshold, further increasing the dataset size does not significantly improve effectiveness. Furthermore, the autonomous driving vehicle exhibits good anti-interference capability against minor environmental disturbances, but when facing severe environmental interference, its road-tracking effectiveness is significantly degraded. Finally, by comparing the performance of two models, we conclude that ResNet18 outperforms ResNet34 in road-tracking tasks and is better suited to autonomous driving.

ACKNOWLEDGMENT

The first author received scholarship support from CP ALL to conduct this research at PIM. The last author is the corresponding author.

REFERENCES

- [1] J. Qu, "Multi-task in autonomous driving through RDNet18-CA with LiSHTL-S loss function," *ECTI Trans. Comput. Inf. Technol.*, vol. 18, no. 2, pp. 158-173, Apr. 2024.
- [2] S. Ding and J. Qu, "Research on multi-tasking smart cars based on autonomous driving systems," *SN Comput. Sci.*, vol. 4, no. 3, p. 292, Mar. 2023.
- [3] K. Tsiakas, I. Kostavelis, D. Giakoumis, and D. Tzovaras, "Road tracking in semi-structured environments using spatial distribution of LiDAR data," in *Artificial Intelligence Applications and Innovations*, Cham, Switzerland: Springer, 2021, ch. 32, pp. 432-445.
- [4] A. Kumar, T. Saini, P. B. Pandey, and A. Agarwal, "Vision-based outdoor navigation of self-driving car using lane detection," *Int. J. Inf. Technol.*, vol. 14, no. 1, pp. 215-227, Aug. 2022.
- [5] Y. Li and J. Qu, "Intelligent road tracking and real-time acceleration-deceleration for autonomous driving using modified convolutional neural networks," *Curr. Appl. Sci. Technol.*, vol. 22, no. 6, pp. 1-26, Mar. 2022.
- [6] Q. Zhang, T. Du, and C. Tian, "Self-driving scale car trained by deep reinforcement learning," *arXiv*, 2019. [Online]. Available: <https://arxiv.org/abs/1909.03467> [Accessed: Aug. 15, 2025].
- [7] B. Peng et al., "End-to-end omous driving through dueling double deep Q-network," *Automot. Innov.*, vol. 4, pp. 328-337, Jun. 2021.
- [8] S. Yuenyong and Q. Jian, "Generating synthetic training images for deep reinforcement learning of a mobile robot," *J. Intell. Inform. Smart Technol.*, vol. 2, no. 3, pp. 16-20, 2017.
- [9] X. Du, M. H. Ang, and D. Rus, "Car detection for autonomous vehicle: LiDAR and vision fusion approach through deep learning framework," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Sep. 2017, pp. 749-754.
- [10] U. Karni, S. S. Ramachandran, K. Sivaraman, and A. K. Veeragahaven, "Development of autonomous downscaled model car using neural networks and machine learning," in *Proc. Int. Conf. Comput. Methodol. Commun. (ICCMC)*, 2019, pp. 1089-1094.
- [11] Y. Bae et al., "Prototyping a system of cost-effective autonomous guided vehicles," in *Proc. Annu. Gen. Donald R. Keith Memorial Conf.*, 2021, pp. 138-143.
- [12] K. L. Lee and H. Y. Lam, "Development of deep learning autonomous car using Raspberry Pi," *Prog. Eng. Appl. Technol.*, vol. 2, no. 1, pp. 534-548, Jun. 2021.
- [13] S. Zang et al., "The impact of adverse weather conditions on autonomous vehicles: How rain, snow, fog, and hail affect the performance of a self-driving car," *IEEE Veh. Technol. Mag.*, vol. 14, no. 2, pp. 103-111, Mar. 2019.
- [14] S. Valladares et al., "Design and implementation of an autonomous vehicle prototype," in *Proc. 4th Int. Conf. Intell. Human Syst. Integr. (IHSI)*, 2021, pp. 343-349.
- [15] V. Rausch et al., "Learning a deep neural net policy for end-to-end control of autonomous vehicles," in *Proc. Am. Control Conf. (ACC)*, 2017, pp. 491-4919.
- [16] Z. Chen and X. Huang, "End-to-end learning for lane keeping of self-driving cars," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, 2017, pp. 1856-1860.
- [17] C. J. Willmott and K. Matsuura, "Advantages of the Mean Absolute Error (MAE) over the Root Mean Square Error (RMSE) in assessing average model performance," *Clim. Res.*, vol. 30, no. 1, pp. 79-82, 2005.
- [18] W. -Y. Lin, W. -H. Hsu, and Y. -Y. Chiang, "A combination of feedback control and vision-based deep learning mechanism for guiding self-driving cars," in *Proc. IEEE Int. Conf. Artif. Intell. Virtual Reality (AIVR)*, 2018, pp. 262-266.
- [19] M. G. Bechtel et al., "Deeppicar: A low-cost deep neural network-based autonomous car," in *Proc. IEEE Int. Conf. Embedded Real-Time Comput. Syst. Appl. (RTCSA)*, 2018, pp. 11-21.
- [20] T. -D. Do et al., "Real-time self-driving car navigation using deep neural network," in *Proc. Int. Conf. Green Technol. Sustain. Dev. (GTSD)*, 2018, pp. 7-12.
- [21] Z. Nie and J. Qu, "Multi-task autonomous driving based on improved convolutional neural network and ST loss in MTS and MOD modes," *Curr. Appl. Sci. Technol.*, vol. 23, no. 3, pp. 1-26, Jun. 2023.
- [22] B. Goldfain et al., "Autorially: An open platform for aggressive autonomous driving," *IEEE Control Syst. Mag.*, vol. 39, no. 1, pp. 26-55, Jun. 2019.
- [23] S. Mehta, C. Paunwala, and B. Vaidya, "CNN based traffic sign classification using Adam optimizer," in *Proc. Int. Conf. Intell. Comput. Control Syst. (ICCS)*, 2019, pp. 1293-1298.
- [24] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2016, pp. 770-778.



Chuanxiang Bi is currently pursuing the Master of Engineering Technology at the Faculty of Engineering and Technology, Panyapiwat Institute of Management, Thailand. He received B.B.A from Nanjing Tech University Pujiang Institute, China, in 2022. His research interests are artificial intelligence, image processing, and autonomous driving.



Jian Qu is a full-time lecturer at the Faculty of Engineering and Technology, Panyapiwat Institute of Management. He received a Ph.D. with an Outstanding Performance award from the Japan Advanced Institute of Science and Technology, Japan, in 2013. He received B.B.A with Summa Cum Laude honours from the Institute of International Studies of Ramkhamhaeng University, Thailand, in 2006, and M.S.I.T from Sirindhorn International Institute of Technology, Thammasat University, Thailand, in 2010. His research interests are natural language processing, intelligent algorithms, machine learning, machine translation, information retrieval, and image processing.