Improving Autonomous Vehicle Performance through Integration of an Image Deraining and a Deep Learning-Based Network for Lane Following

Hoang Tran Ngoc *, Phuc Phan Hong, Anh Nguyen Quoc, and Luyl-Da Quach

Software Engineering Department, FPT University, Can Tho, VietNam; Email: PhucPHCE171166@fpt.edu.vn (P.P.H.), AnhNQCE170483@fpt.edu.vn (A.N.Q.), Luyldaquach@gmail.com (L.-D.Q.)
*Correspondence: Hoang2531992@gmail.com (H.T.N.)

***Abstract*—Lane-keeping is a vital component of autonomous driving that requires multiple artificial intelligence technologies and vision systems. However, maintaining a vehicle's position within the lane is challenging when there is low visibility due to rain. In this research, a combination of image deraining and a deep learning-based network is proposed to improve the performance of the autonomous vehicle. First, a robust progressive Residual Network (ResNet) is used for rain removal. Second, a deep learning-based network architecture of the Convolutional Neural Networks (CNNs) is applied for lane-following on roads. To assess its accuracy and rain-removal capabilities, the network was evaluated on both synthetic and natural Rainy Datasets (RainSP), and its performance was compared to that of earlier research networks. Furthermore, the effectiveness of using both deraining and non-deraining networks in CNNs is evaluated by analyzing the predicted steering angle output. The experimental results show that the proposed model generates safe and accurate motion planning for lane-keeping in autonomous vehicles.**

***Keywords*—autonomous vehicle, image processing, rain removal, convolutional neural networks, autoencoder, residual network**

I. Introduction

Driver error is the most common cause of about 1.3 million road traffic deaths annually as reported by the World Health Organization (WHO) [1]. Therefore, the field of autonomous vehicles is rapidly growing, and the development of reliable perception systems is critical for the success of autonomous vehicles. In real-world scenarios, the data taken by the sensors (ex: camera, LiDAR, etc.) during rainy weather are affected by rain streaks, which can negatively impact certain advanced perception tasks [2, 3], such as pedestrian detection [4], object tracking [5], and semantic segmentation [6]. Among them, lane-keeping is an essential aspect of autonomous driving, and numerous survey papers have explored different algorithms for this purpose [7–9]. However, rain is a common environmental factor that can significantly impact self-driving cars' ability to maintain their lane. As a result, the development and testing of lane-keeping algorithms should take into account the effect of challenging weather conditions on their performance.

Recently, numerous methods have been proposed for rain removal in images and videos [10–19]. The majority of techniques for rain removal can be classified as either model-driven or data-driven, and there are more specialized subdivisions within each of these categories. Addressing this issue has become an increasingly popular area of research in algorithm design. Generally, the existing removal rain methods can be divided into four categories: Single-image Deraining [11–13], Video and Multi-image Deraining [14, 15]. Filtering-based Image Restoration [16, 17], and Data Augmentation Techniques [18, 19]. However, these methods are often complex and computationally expensive, making them unsuitable for real-time applications. On the other hand, data-driven methods, such as deep learning-based methods [20], are more computationally efficient and can produce high-quality results. These methods still face challenges in terms of scalability, generalization ability, and computational cost.

In addition, methods of using camera images to predict steering angle and control the vehicle to follow the lane are constantly evolving. In conventional methodologies, the task is commonly broken down into multiple components, including lane detection [21, 22], path planning [23, 24], and control logic [25, 26], which are typically studied in isolation. However, these traditional approaches to self-driving cars have several limitations. One major limitation is that they require significant human engineering and domain expertise to design and tune the perception, planning, and control modules. Moreover, they may not generalize well to new environments or scenarios that were not encountered during the development process.

This research presents a robust progressive ResNet called Progressive Recurrent Network (PReNet) [30] for image deraining combined with CNNs steering controller

Manuscript received April 4, 2023; revised May 25, 2023; accepted June 27, 2023; published November 3, 2023.

doi: 10.12720/jait.14.6.1159-1168

for lane-keeping in rain conditions. Rain removal before inputting data to train and predict steering angle gives more accurate results than using rain data directly to train CNNs to predict steering angle. Both methods are

improved methods that overcome the disadvantages of conventional methods. The main contributions are:

- PReNet model is lightweight, fast, and effective for rain removal. This model is trained and evaluated by the RainSP dataset of vehicle images running in the rain.
- Auto steering control uses end-to-end learning with CNNs architecture and provides greater accuracy in lane-keeping than conventional image processing methods.
- To evaluate the effectiveness of the proposed method for rain removal and lane-keeping, the RainSP dataset is collected by driving a donkey car around a track and adding different rain intensities, orientations, and haziness levels to create pairs of ground-truth and rainy images for training and evaluation. Comparisons with previous methods are also presented with the same data set.

The remainder of this paper is structured as follows: In Section II, a thorough analysis of previous deraining and lane-keeping studies is presented. Sections III and IV outline the de-raining network and end-to-end learning network, detailing the data collection process, evaluation metrics, and network architecture. Section V illustrates the experimental results and provides a discussion. Finally, in Section VI, the paper concludes and suggests areas for future research.

II. Related Work

A. Image Deraining

Rain removal from images has been a persistent challenge in computer vision, and over the years, a variety of techniques have been proposed to tackle this problem. Early methods typically relied on manually engineered features and physical models to separate rain streaks or noise from the underlying scene. Several proposed methods include:

- Denoising Autoencoders (DAE) [27]: DAE is an unsupervised approach for robust feature learning that utilizes an encoder and a decoder network. It is effective in learning invariant features but may not be as good at learning high-level features.
- Deep Detail Network (DDN) [28]: DDN is a deep learning-based method that applies a low-pass filter to decompose the input image into a base and detail layer, but may struggle with heavy rain and generate artifacts.
- DerainNet [29]: DerainNet is a deep CNN-based method for removing rain from single images. It learns the nonlinear mapping function between clean and rainy images and incorporates domain knowledge to improve restoration quality. However, it may not work well for heavy rain, generate artifacts, and require a large amount of training data.
- Progressive ResNet (PRN) [30]: PRN is a deep-learning rain removal method with state-of-the-art performance, but may generate artifacts and is not effective for heavy rain/irregular shapes.
- The majority of the methods proposed for AVs fail to consider their system requirements, resulting in deficiencies that include inadequate precision, lengthy computation periods, inappropriateness for real-time situations, and restricted applicability to certain types of degradation elements. These approaches have been compared and summarized in Table I.

TABLE I. Rain Removal Approaches

Approach (Variables or Priors)	Method	Limitation
Vincent *et al*. (DAE) [27] (Clean image)	The network consists of an encoder and a decoder network, with the encoder network used to extract features from the input data and the decoder network used to reconstruct the original data from the extracted features.	It may not be as effective in learning high-level features as other deep learning methods.
Fu *et al*. (DDN) [28] (Residual)	To enhance the extraction of rain streaks from detail layers, a deeper residual network (ResNet) [31] is utilized in place of a CNN. The network solely receives high-frequency details as input and predicts the difference between the rain and clean images.	The method still cannot handle large and sharp rain streaks and it may also remove some non-rain textures.
Fu *et al*. (DerainNet) [29] (Residual)	The process involves breaking down a rainy image into two parts: a foundational structure layer and a high-frequency detail layer. From there, a 3-layer Convolutional Neural Network (CNN) is used to extract the rain streaks from the high-frequency detail layer.	Can generate artifacts or distortions in the output image, especially in regions with complex textures or patterns.
Ren *et al*. (PRN) [30] (Residual)	Based on a progressive training strategy can learn robust features that are invariant to various types of input perturbations while requiring fewer parameters and computational resources compared to other methods.	May still generate artifacts or distortions in the output image, especially in regions with complex textures or patterns.

B. Auto Steering Controller

In recent years, End-to-end learning has emerged as a promising approach to address the limitations of conventional lane-keeping approaches. In end-to-end learning, the entire system is learned directly from data, without the need for handcrafted modules. The system takes raw sensor inputs, such as images or LiDAR data, and outputs the vehicle's control commands, such as steering angle or throttle. The system is trained using supervised learning, where the training data consists of pairs of sensor inputs and corresponding control commands.

- End-to-End Learning for Self-Driving Cars" by Nvidia [7]: They introduced an end-to-end deep learning approach for self-driving cars using a convolutional neural network to map raw pixels from a front-facing camera to steering commands.
- "End-to-End Learning for Lane Keeping of Self-Driving Cars" by Cheng *et al.* [8]: They present an end-to-end learning approach for lane keeping using input from a front-facing camera and a convolutional neural network.
- "An End-to-End Deep Neural Network for Autonomous Driving Designed for Embedded Automotive Platforms" by Kosic *et al.* [9]: They present a light deep neural network for autonomous driving that is suitable for deployment on embedded automotive platforms.

However, the above methods do not consider the impact of rain when performing train and running the experimental model.

C. Proposed Method

To eliminate the disadvantages of the above analysis methods, the PReNet is applied in combination with CNNs. The images extracted from the camera will undergo rain removal before being fed into CNNs to predict the steering angle for lane-keeping. The PReNet model is designed to be lightweight, fast, and effective, making it a promising solution for rain removal in autonomous driving. This will ensure that the CNNs are trained on high-quality images and thus, enhance their performance in detecting and tracking the lane markings accurately. The architecture of our proposal is shown in Fig. 1.

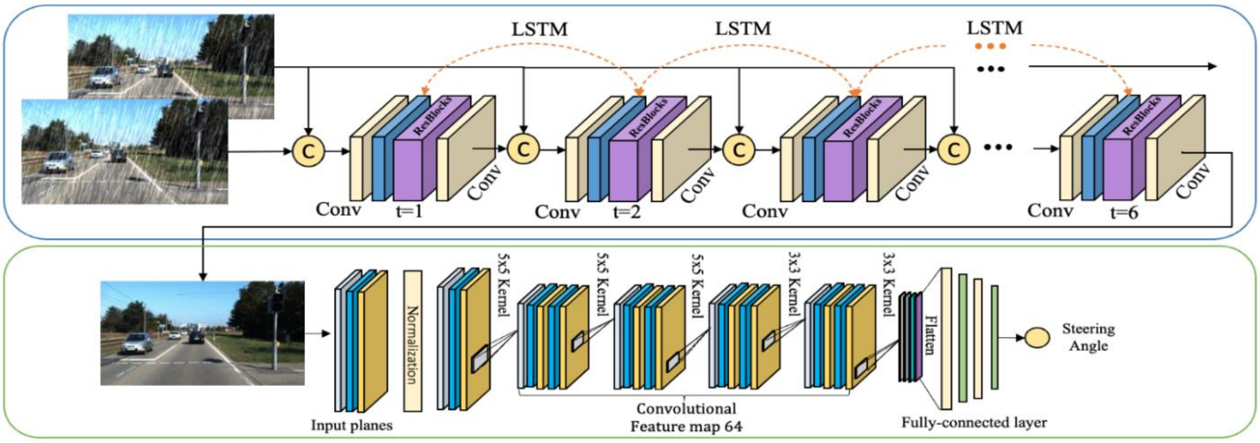


Figure 1. Our proposal integrates rain removal with lane-keeping model architecture.

III. Deraining Network

A. Dataset

To evaluate the effectiveness of PReNet in autonomous driving, experiments are conducted using a synthetic dataset named RainSP-1 dataset and a real-world dataset named RainSP-2 dataset.

The RainSP-1 dataset was gathered by navigating a donkey car on a track. Then, OpenCV is used to add different rain intensities, orientations, and haziness levels to the collected images to create pairs of ground-truth and rainy images for training and evaluation. The dataset consists of 6,000 image pairs. Each image has a resolution of 120×160 pixels. Fig. 2 shows some examples of the RainSP-1 dataset with five rows: Ground Truth, Level 1, Level 2, Level 3, and Level 4. The Ground Truth row shows the original image without any rain. The Level 1 row shows the image with low-intensity rain, while the Level 4 row shows the image with high-intensity rain.

To evaluate the generalization ability of the model, we also collected a real-world dataset named RainSP-2 from Kaggle and the Internet. The RainSP-2 dataset is a synthetic dataset that contains 7,200 pairs of rainy and corresponding ground truth images. Fig. 2 shows some examples of the RainSP-2 dataset with two columns: Ground Truth and Rainy Image. The Ground Truth column shows the original image without any rain, while the Rainy Image column shows the same image with rain added.

Both RainSP-1 and RainSP-2 datasets were used to train and test the PReNet model for rain removal in autonomous driving. The variety of datasets enabled us to enhance the model's resilience and capacity for generalization.

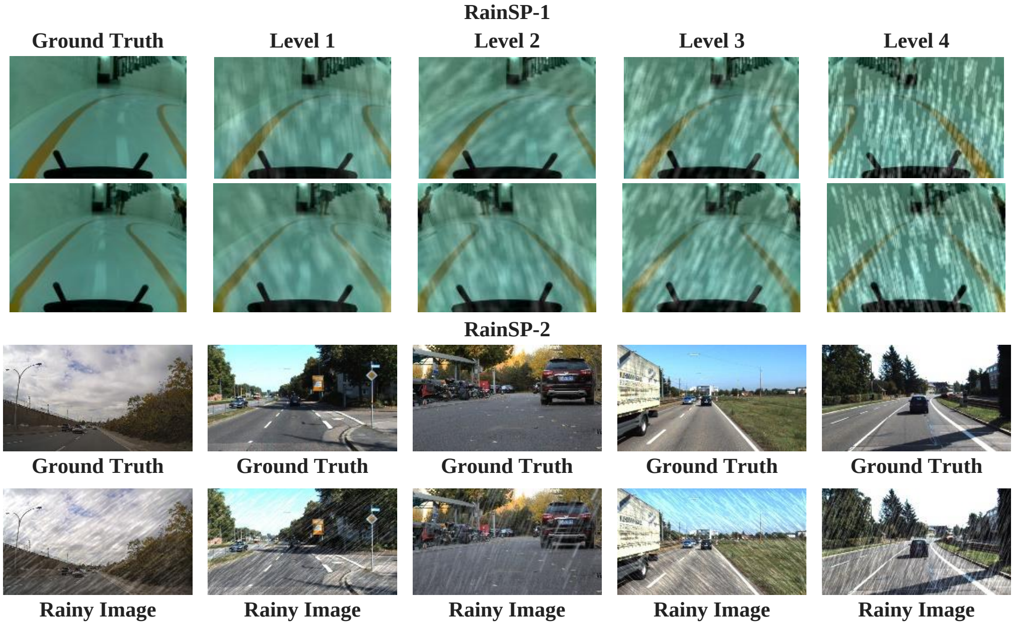


Figure 2. Example images of the RainSP-1 and RainSP-2 datasets.

B. *Evaluation Metrics*

The performance of the Progressive Image Deraining Networks (PReNet) for rain removal was assessed using two main evaluation criteria based on reference rain-free images: Peak-Signal-to-Noise Ratio (PSNR) and Structural Similarity Index (SSIM) [32].

PSNR is a commonly used metric for image quality assessment that measures the maximum pixel value ratio to the root mean squared error between the predicted de-rained image and the reference rain-free image. A higher PSNR value indicates better image quality, while a small PSNR value implies significant numerical differences between images. The PSNR formula is:

$$PSNR(x,y) = 10\ \log_{10}\left(\frac{255^2}{MSE(x,y)}\right) \tag{1}$$

where x and y are the two compared images, and MSE is the mean squared error between the original and reconstructed images. The Mean Squared Error (MSE) [36] is defined as:

$$MSE(x,y) = \frac{1}{MN}\sum_{i=1}^{M}\sum_{i=1}^{N}\left(x_{ij} - y_{ij}\right)^2 \tag{2}$$

where M and N are the column and row pixels of images.

On the other hand, SSIM [33] measures the similarity between two images and is modeled in terms of three factors: loss of correlation, luminance distortion, and contrast distortion. The SSIM metric varies between 0 and 1, with a score of 1 indicating complete similarity and a score of 0 indicating no similarity. An increased SSIM value indicates better quality of the image. The formula for SSIM is:

$$SSIM(x,y) = [L(x,y)]^{\alpha} \cdot [C(x,y)]^{\beta} \cdot [S(x,y)]^{\gamma} \tag{3}$$

where $[L(x,y)]^{\alpha}$, $[C(x,y)]^{\beta}$, and $[S(x,y)]^{\gamma}$ are the luminance, contrast, and structure comparíon. $\alpha,\ \beta,$ and γ are positive parameters, the magnitude of which depends on the requirement to compare which factor is more prominent.

For training PReNet with t stages, a single loss function (either MSE loss or negative SSIM loss) was used, and only the final output x^t was supervised. In terms of both PSNR and SSIM, the negative SSIM loss demonstrated superior performance when compared to the MSE loss, and it was employed to assess the effectiveness of PReNet. The specifics of PReNet's architecture will be discussed in the subsequent section.

C. *Network Architecture for Deraining*

Recently, numerous networks raining from simple to the complex have been researched using residual networks (ResNets) [31] for eliminating rain streaks. In this section, the network architecture of PReNet is described, as well as compared to two other rain removal networks, the DDN and PRN.

Through a deep detail network, Fu *et al.* [28] introduced the DDN technique to eliminate rain from individual images, as shown in Fig. 3(a). A low-pass filter decomposition technique is used to separate the detail layer before putting it into the negative ResNet (Neg-ResNet). However, this approach is still insufficient in processing large and distinct types of rain, such as our diverse RainSP-1 dataset with varying levels of rainfall, and there is a possibility that it might eliminate some structures that are not related to rain.

Ren *et al.* [30] compared two networks PRN and PReNet. PReNet is a proposed network that uses a combination of ResBlocks and Long Short-Term Memory (LSTM) [34] cells to effectively remove rain from images. The results proved that PReNet is better than PRN and simpler baseline than DDN. Therefore, in this paper, PreNet is applied and trained on the self-driving car dataset under rainy conditions before combining it with the CNNs steering controller to overcome the disadvantages of previous methods.

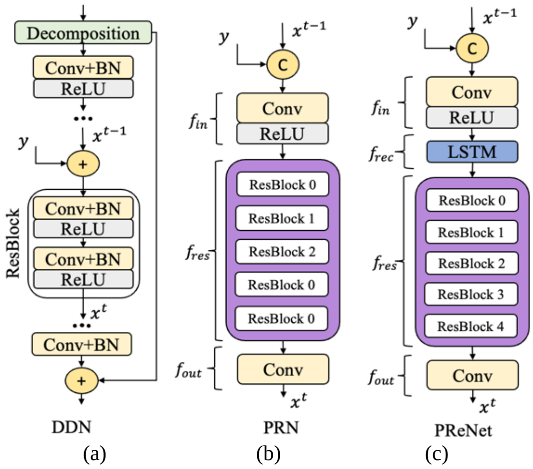


Figure 3. The three network architectures of DDN, PRN and PReNet for the rain removal problem. (a) DDN network, (b) PRN network, and (c) PReNet network.

The architecture of PReNet and PRN are shown in Fig. 1 and Figs. 3(b)–(c). The PReNet architecture consists of f_{in} as a convolutional layer with ReLU (Rectified Linear Unit) activation [35], f_{rec} as an LSTM module, f_{res} as five residual blocks, and f_{out} as a convolutional layer.

- The input to the network is a rainy image with three channels, and the output is the corresponding derained image with the same dimensions.
- At each iteration, the network takes a concatenated input of the rainy image x^{t-1} and the previous output y, which is fed into f_{in} that consists of a 2D convolutional layer with a kernel size of 3×3 and ReLU activation.
- The output is then concatenated with the hidden state of the LSTM (f_{rec}). The LSTM module consists of four 2D convolutional layers, each with a kernel size of 3×3, followed by a Sigmoid activation for input gate i, forget gate f, and output gate o, and a Tanh activation for the input modulation gate g. The output of the gates is used to control the flow of information within the recurrent loop.
- The output of the LSTM module is then passed through f_{res} which includes five residual blocks, each consisting of two convolutional layers with 32 filters of size 3×3, ReLU activation functions. The output of the last residual block is passed through f_{out}, which is a 2D convolutional layer with a kernel size of 3×3 to obtain the initial estimate of the rain-free image.
- To ensure that the dimensions of the input and output images remain the same throughout the network, padding is applied to the convolutional layers with a kernel size of 3×3 and a stride of 1. Specifically, zero padding is added to the input image so that the output has the same dimensions as the input.

Once the rain is eliminated from the image, it will be inputted into the CNN model to predict the steering angle for the vehicle to follow the lane.

IV. End-To-End Steering Controller Network

A. Data Collection

In order to train and evaluate the performance of our end-to-end learning network, we utilized the original RainSP-1 dataset which includes images with the corresponding steering angles. During the data collection process, we used the Raspberry Pi-based donkey car to drive around the track for approximately 30 laps. The car's camera captured images of the road, which were then used to train the network. To obtain the corresponding steering angle for each image, the car's built-in motor control system is used to record the steering angle during the driving session. During the data collection process, it was also made sure by us to vary the steering angle to ensure that the dataset was diverse and representative of real-world driving scenarios. The dataset was divided into training and testing sets, where 80% of the data was allocated to training and the remaining 20% was allocated to testing.

B. Evaluation Metrics

In order to train and evaluate the performance of our end-to-end learning network, we utilized the original RainSP-1 dataset which includes images with the corresponding steering angles. In order to evaluate the performance of the end-to-end steering prediction network, the MSE loss function is used for training the model as it is a regression problem. The MSE is another commonly used metric for regression tasks that measures the average of the squared differences between the predicted values and the ground-truth values. The MSE is defined as:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}\left(y_{pred} - y_{gt}\right)^2 \tag{4}$$

where y_{pred} is the predicted steering angle, y_{gt} is the ground-truth steering angle, and n is the total number of data samples. A lower MSE value indicates better performance.

C. Network Architecture

The proposed network architecture used an end-to-end neural network called PilotNet, which takes in input RGB images of size 120×160×3 after deraining and outputs steering angles for self-driving cars. It is implemented using the Keras library in Python. As shown in Fig. 4, the PilotNet model consists of five convolutional layers followed by five fully connected layers:

- The input to the model is an image of size 120×160×3, where 120 is the height, 160 is the width, and 3 is the number of color channels.
- The model comprises of five convolutional layers, with the first three having a kernel size of 5×5 and a stride of 2, and the final two having a kernel size of 3×3 and a stride of 1. The convolutional layers contain 24, 36, 48, 64, and 64 kernels, respectively, and implement the ReLU activation function. Padding is not utilized in this instance.
- After the last convolutional layer, the output is flattened and fed into the first fully connected layer,

which has 6656 neurons. The output of this layer is then passed through a ReLU activation function and dropout regularization before being fed into the second fully connected layer with 100 neurons. The same process is repeated for the third fully connected layer with 50 neurons and the fourth fully connected layer with 10 neurons. Finally, the output of the fourth fully connected layer is passed through and the output layer contains a single neuron, which predicts the steering angle.

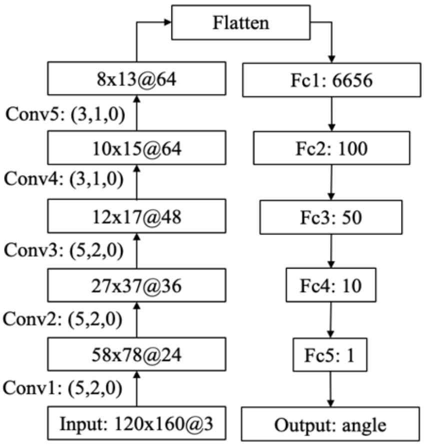


Figure 4. The network architecture of Nvidia PilotNet.

V. Experimental Result

A. Experimental Setting

A Raspberry Pi-based Donkey Car is built to collect data and evaluate the proposed models. As shown in Fig. 5, the car is equipped with three components, including a Raspberry Pi4 for processing, a Creative BlasterX® Senz3D® camera for image capturing, and a servo motor for steering angle control. The car was used in two modes: data collection and autonomous driving. During data collection mode, frames and steering angles were recorded while driving the car around a map using a joystick. In autonomous driving mode, the car was equipped with the trained model and assessed for accuracy and performance during the autonomous driving mode.

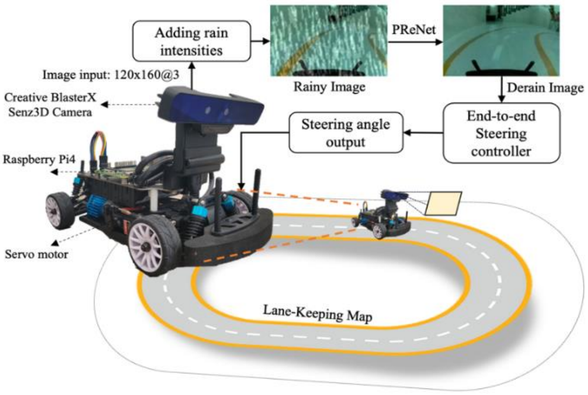


Figure 5. Experimental vehicle and block diagram of the proposed method.

The image deraining and end-to-end learning networks were trained using the Pytorch 1.4.0 and Tensorflow 2.2.0 frameworks on a computer with an Intel Core I3-10105F CPU and an Nvidia GeForce GTX 1660 GPU. Both networks are trained using the Adam optimizer with a learning rate of 1×10^{-3}. For image deraining, the learning rate is reduced by a factor of 10 after 30, 50, and 80 epochs. The deraining network is trained for a total of 100 epochs with a batch size of 18. For the end-to-end learning network, the training is performed with a checkpoint callback function that saves the best model based on validation loss.

B. Experimental Results

1) Evaluation of rain removal performance

This section presents a comparison of all competing methods on the proposed datasets, RainSP-1 and RainSP-2, in a quantitative manner. RainSP-1 contains 6,000 synthesized clean/rainy image pairs in four different rain levels, while RainSP-2 contains 7,200 synthesized clean/rainy image pairs. The competing methods include DAE [27], DerainNet [29], DDN [28], PRN [30], and PReNet [30]. For all datasets, PSNR and SSIM serve as the quantitative evaluation metrics, whereby higher PSNR and SSIM scores correspond to better deraining outcomes.

Table II presents the quantitative comparison of all competing methods on RainSP-1 and RainSP-2. It can be seen that PReNet achieves the best deraining performance among all competing methods on each dataset. Specifically, on the RainSP-1 dataset, PReNet achieves an average PSNR of 32.17 dB and an average SSIM of 0.941. while the second-best method achieves an average PSNR of 31.81 dB and an average SSIM of 0.932. On the RainPS-2 dataset, the average PSNR improvement of PReNet over the second-best method PRN is 2.64 dB, and the average SSIM improvement is 0.017.

TABLE II. Average PSNR and SSIM Comparison on RainSP-1 and RainSP-2 Synthetic Datasets

Method	RainSP-1		RainSP-2	
	PSNR	**SSIM**	**PSNR**	**SSIM**
DAE	19.38	0.583	20.48	0.705
DDN	21.61	0.656	24.3	0.81
DerainNet	19.42	0.687	20.4	0.772
PRN	31.81	0.932	33.45	0.959
PReNet	**32.17**	**0.941**	**36.09**	**0.976**

In addition, Figs. 6 and 7 visualize the output of all competing methods in RainSP-1 and RainSP-2, respectively. PReNet exhibits better restoration performance, especially on RainSP-1, which includes diverse rain levels. In Fig. 6, DAE tends to over-smooth the images, while DerainNet and DDN lose background textures, especially the lane lines. In contrast, PRN and PReNet perform better in removing the rain streaks and preserving image details. However, in level 4 of rain, lane lines from PRN are not as clear and sharp as PReNet. This proves that PReNet can address different rain patterns effectively. Overall, PReNet not only effectively addresses different rain patterns but also preserves background details better than other competing methods. Fig. 7

illustrates that the output images from other methods tend to blur the texture of the image or leave some rain streaks visible, while PReNet can effectively remove the rain without leaving any noticeable rain streaks.

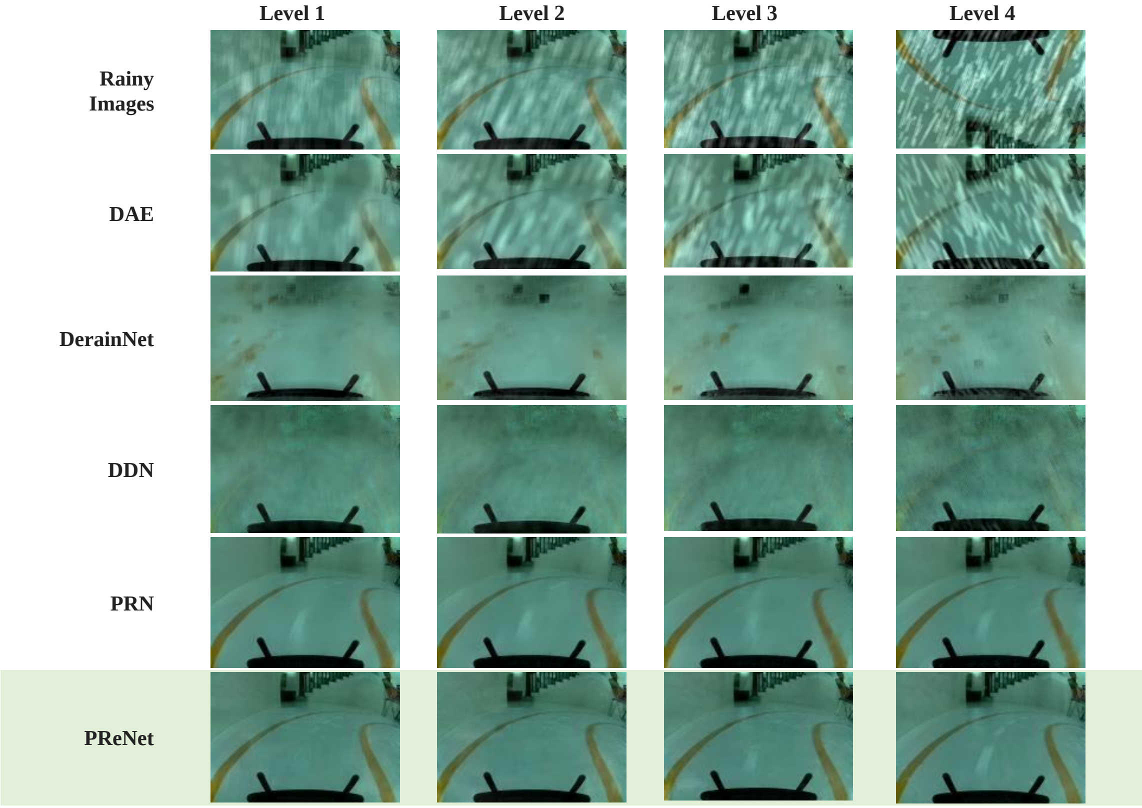


Figure 6. Qualitative results on RainSP-1 dataset. There are 4 columns with each column being a rainy image and the results of competing methods in one rain level.

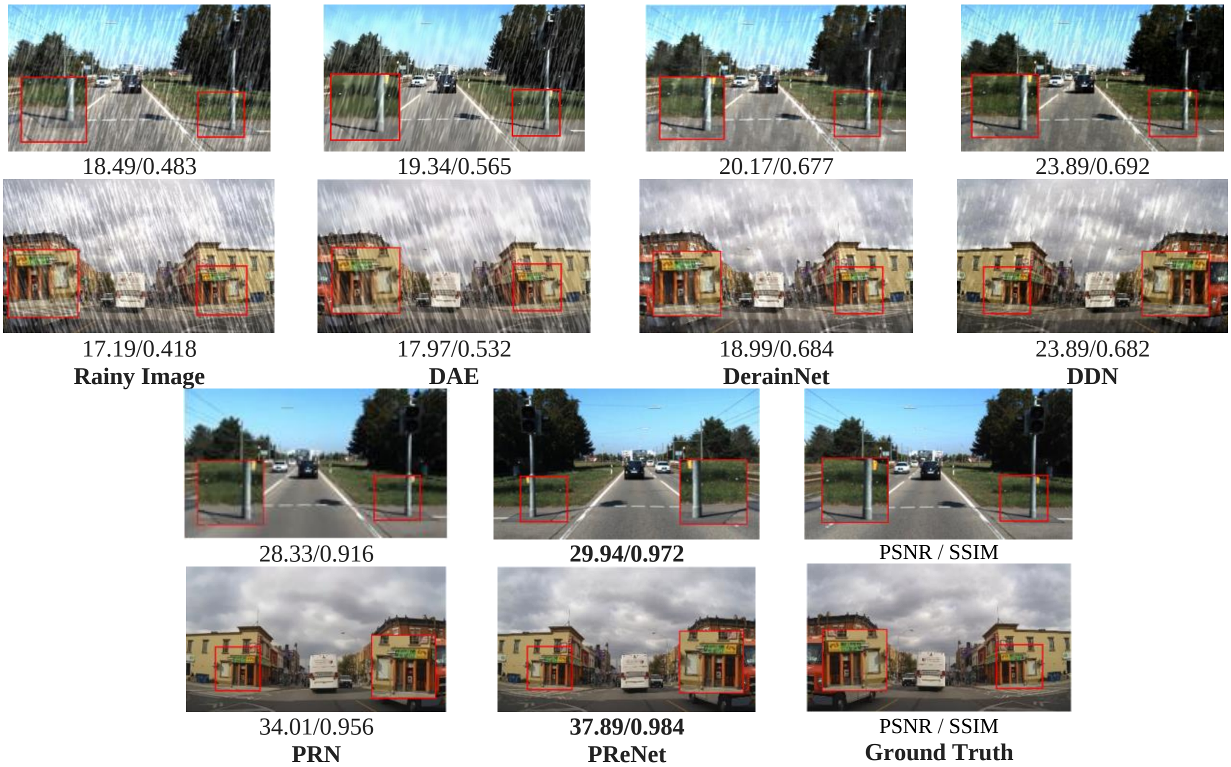


Figure 7. Rain removal performance on a rainy image from RainSP-2. PSNR/SSIM values are provided below each derained image to facilitate easy comparison. The best results are underlined.

The demanding real-time requirements of autonomous vehicle tasks, in this case - steering prediction, necessitate that any additional processing does not add high computational complexity that could potentially hinder real-time performance. Hence, as computational efficiency is crucial for autonomous driving, the comparison of parameters and processing time of PReNet with other competing methods is necessary. Table III presents the model complexity (in parameters) and processing time (in seconds) of all methods based on a computer equipped with an NVIDIA GTX 1660 GPU. The processing time of each method is an average time, which is calculated on 535 images of 120×160 size from the RainSP-1 dataset. The results show that PReNet achieves remarkable improvements over other methods. Specifically, PReNet has 168,963 parameters, which is smaller than DerainNet, and it takes only 0.008 seconds per image on average, which is about 266.3 times faster than DAE and 1.9 times faster than DerainNet. These results demonstrate the effectiveness of PReNet as a lightweight and efficient method for rain removal in the context of autonomous driving, where real-time performance is crucial.

TABLE III. THE PARAMETER AND PROCESSING TIME OF COMPETING

Method	Time	Param
DAE	2.13	112.001
DDN	0.006	58.175
DerainNet	0.015	754.691
PRN	0.007	95.107
PReNet	**0.008**	**168.963**

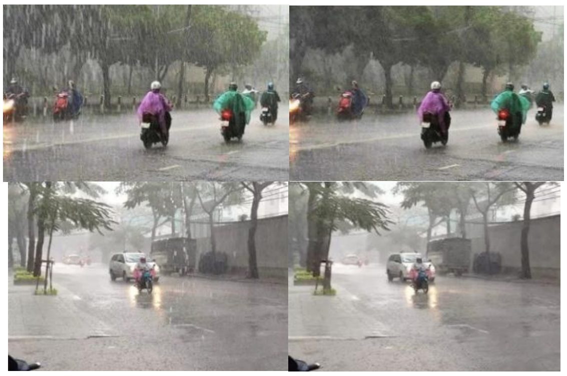


Figure 8. Rain removal performance on real rain image that we colect on the internet.

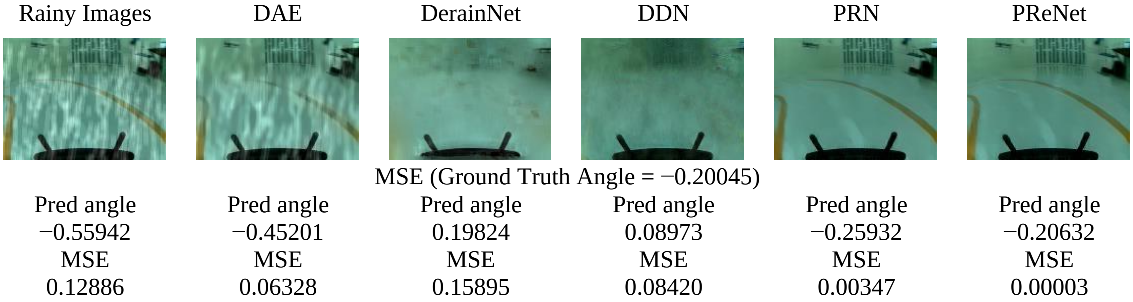
Fig. 8 showcases the effectiveness of PReNet in deraining real rain images through a pair-wise comparison. The first column exhibits original rain images, while the second column presents the same images after applying PReNet for deraining. PReNet demonstrates remarkable performance by reducing rain-induced distortions and enhancing image quality. The side-by-side evaluation emphasizes the practical applicability and effectiveness of PReNet in removing rain artifacts. The visual evidence provided by Fig. 8 solidifies PReNet's efficacy as a reliable solution for deraining real rain images.

2) Evaluation of steering angle prediction

This section presents a comparison of all competing methods on the proposed datasets, RainSP-1 and RainSP-2, in a quantitative manner. RainSP-1 contains 6,000 synthesized clean/rainy image pairs in four different rain levels, while RainSP-2 contains 7,200 synthesized clean/rainy image pairs. The competing methods include DAE [27], DerainNet [29], DDN [28], PRN [30], and PReNet [30]. PSNR and SSIM are used as the quantitative evaluation metric for all datasets, with larger PSNR and SSIM values indicating better deraining results.

This section describes the testing of the trained models on real autonomous driving scenarios using the Raspberry Pi-based donkey car. Two models have been transferred to the car: the PReNet model for pre-processing frames and a PilotNet model for predicting steering angles in real-time. As shown in Fig. 5, the models were evaluated by driving the car on an unknown second track, where the input image from the camera was modified by adding different levels of rain intensity. The modified image was then fed into the PReNet model for deraining and the PilotNet model for steering angle prediction. The results show that the proposed method was effective in enabling the car to drive autonomously under simulated adverse weather conditions. The car remained stable and drove accurately under rainy conditions. The same setup was also used to test other competing methods, where only DDN was able to keep the car in its lane while DAE, DerainNet, and DDN caused the car to go straight ahead all the time.

For better visualization, Fig. 9 illustrates the Mean Squared Error (MSE) between the angle predicted by the PilotNet and the actual angle in some RainSP-1 dataset images for each competing method. It can be observed that PReNet produces the smallest MSE value among all the compared methods, which indicates that PReNet can effectively remove rain streaks and retain more image details. Specifically, in the last case, PReNet achieves MSE values of 1×10^{-5}, which are five times lower than the second-best method PRN. From these results, the proposed method can be used for real-world autonomous driving applications, especially in areas with heavy rainfall.



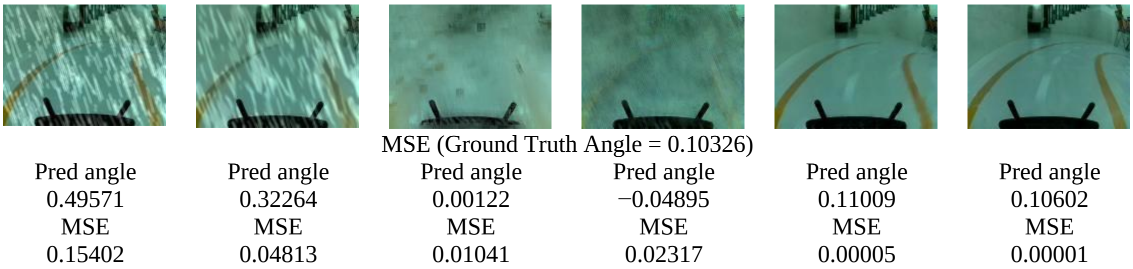

MSE (Ground Truth Angle = 0.10326)
Pred angle 0.49571 MSE 0.15402
Pred angle 0.32264 MSE 0.04813
Pred angle 0.00122 MSE 0.01041
Pred angle −0.04895 MSE 0.02317
Pred angle 0.11009 MSE 0.00005
Pred angle 0.10602 MSE 0.00001

Figure 9. The MSE values between predicted angles of PilotNet from deraining results of the competing methods on the RainSP-1 dataset.

VI. Conclusions

In this paper, the deraining method (PReNet) was combined with auto-steering controllers (CNNs) for rain removal and lane-keeping in autonomous driving. A comprehensive review of previous studies on rain removal methods was conducted, and PReNet was found to be one of the most efficient methods due to its lightweight architecture, fast processing time, and excellent deraining performance. The method was evaluated using the synthetic RainSP dataset and showed improvements in end-to-end steering prediction accuracy in the rain. The results of this study demonstrate that PReNet can improve and enhance the performance of autonomous driving systems in rain and contribute to the development of safer and more advanced systems.

Conflict of Interest

The authors declare no conflict of interest.

Author Contributions

Hoang Tran Ngoc and Luyl-Da Quach had the idea for the research and provided support for Phuc Phan Hong and Anh Nguyen Quoc. Phuc Phan Hong conducted the research, while Anh Nguyen Quoc wrote the paper and prepared the dataset; all authors approved the final version.

References

[1] Road Traffic Injuries. [Online]. Available: https://www.who.int/news-room/fact-sheets/detail/road-traffic-injuries

[2] K. Garg and S. K. Nayar, "When does a camera see rain?" in *Proc. Tenth IEEE International Conference on Computer Vision*, October 2005, vol. 2, pp. 1067–1074.

[3] G. Kshitiz and S. K. Nayar, "Vision and rain," *International Journal of Computer Vision*, vol. 75, pp. 3–27, October 2007.

[4] X. Wang, T. Xiao, Y. Jiang, S. Shao, J. Sun, and C. Shen, "Repulsion loss: Detecting pedestrians in a crowd," in *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7774–7783, June 2018.

[5] F. Li, C. Tian, W. Zuo, L. Zhang, and M.-H. Yang, "Learning spatial-temporal regularized correlation filters for visual tracking," in *Proc. CVPR*, December 2018, pp. 4904–4913.

[6] Y.-C. Chen, Y.-Y. Lin, M.-H. Yang, and J.-B. Huang, "Show, match and segment: Joint weakly supervised learning of semantic matching and object cosegmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, pp. 3632–3647, April 2020.

[7] B. Mariusz, D. D. Testa, D. Dworakowski, B. Firner, B. Flepp, P. Goyal, L. D. Jackel *et al*., "End to end learning for self-driving cars", arXiv preprint, arXiv:1604.07316, April 2016.

[8] Z. Chen and X. Huang, "End-to-end learning for lane keeping of self-driving cars," in *Proc. 2017 IEEE Intelligent Vehicles Symposium (IV)*, June 2017, pp. 1856–1860.

[9] K. Jelena, N. Jovičić, and V. Drndarević, "An end-to-end deep neural network for autonomous driving designed for embedded automotive platforms," *Sensors*, vol. 19, May 2019.

[10] H. Wang, Y. Wu, M. Li, Q. Zhao, and D. Meng, "A survey on rain removal from video and single image", arXiv preprint, arXiv:1909.08326, October 2019.

[11] Z. Tu, H. Talebi, H. Zhang, F. Yang, P. Milanfar, A. Bovik, and Y. Li, "Maxim: multi-axis MLP for image processing," in *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition*, June 2022, pp. 5769–5780.

[12] T. Wang, X. Yang, K. Xu, S. Chen, Q. Zhang, and R. W. Lau, "Spatial attentive single-image deraining with a high quality real rain dataset," in *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition*, August 2019, pp. 12262–12271.

[13] Q. Guo, J. Sun, F. J. Xu, L., Ma, X. Xie, W. Feng, Y. Liu, and J. Zhao, "Efficientderain: Learning pixel-wise dilation filtering for high-efficiency single-image deraining," in *Proc. AAAI Conference on Artificial Intelligence*, February 2021, vol. 35, pp. 1487–1495.

[14] Z. Yue, J. Xie, Q. Zhao, and D. Meng, "Semi-supervised video deraining with dynamical rain generator," in *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition*, June 2021, pp. 642–652.

[15] W. Yan, R. T. Tan, W. Yang, and D. Dai, "Self-aligned video deraining with transmission-depth consistency," in *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2021, pp. 11961–11971.

[16] S. P. Awate and R. T. Whitaker, "Unsupervised, information-theoretic, adaptive image filtering for image restoration," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, pp. 364–376, January 2006.

[17] D. Kundur and D. Hatzinakos, "A novel blind deconvolution scheme for image restoration using recursive filtering," *IEEE Transactions on Signal Processing*, vol. 46, pp. 375–390, February 1998.

[18] S. Yun, D. Han, S. J. Oh, S. Chun, J. Choe, and Y. Yoo, "Cutmix: Regularization strategy to train strong classifiers with localizable features," in *Proc. International Conference on Computer Vision (ICCV)*, May 2019, pp. 6022–6031.

[19] T. DeVries and G. W. Taylor, "Improved regularization of convolutional neural networks with cutout," arXiv preprint, arXiv:1708.04552, August 2017.

[20] X. Wang, Z. Li, H. Shan, Z. Tian, Y. Ren, and W. Zhou, "FastDerainNet: A deep learning algorithm for single image deraining," *IEEE Access*, vol. 8, pp. 127622–127630, July 2020.

[21] J. Zhao, B. Xie, and X. Huang, "Real-time lane departure and front collision warning system on an FPGA," in *Proc. IEEE High Performance Extreme Computing Conference (HPEC)*, September 2014, pp. 1–5.

[22] A. J. Humaidi and M. A. Fadhel, "Performance comparison for lane detection and tracking with two different techniques," in *Proc. Al-Sadeq International Conference on Multidisciplinary in IT and Communication Science and Applications (AIC-MITCSA)*, May 2016, pp. 1–6.

[23] C. Li, J. Wang, X. Wang, and Y. Zhang, "A model based path planning algorithm for self-driving cars in dynamic environment," in *Proc. Chinese Automation Congress (CAC)*, November 2015, pp. 1123–1128.

[24] S. Yoon, S. E. Yoon, U. Lee, and D. H. Shim, "Recursive path planning using reduced states for car-like vehicles on grid maps,"

IEEE Transactions on Intelligent Transportation Systems, vol. 16, pp. 2797–2813, May 2015.

[25] J. Kong, M. Pfeiffer, G. Schildbach, and F. Borrelli, "Kinematic and dynamic vehicle models for autonomous driving control design," in *Proc. IEEE Intelligent Vehicles Symposium (IV)*, July 2015, pp. 1094–1099.

[26] D. Wang and F. Qi, "Trajectory planning for a four-wheel-steering vehicle," in *Proc. IEEE International Conference on Robotics and Automation*, May 2001, vol. 4, pp. 3320–3325.

[27] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol, "Extracting and composing robust features with denoising autoencoders," *Association for Computing Machinery*, pp. 1096–1103, July 2008.

[28] X. Fu, J. Huang, D. Zeng, Y. Huang, X. Ding, and J. Paisley, "Removing rain from single images via a deep detail network," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017, pp. 1715–1723.

[29] X. Fu, J. Huang, X Ding, Y. Liao, and J. Paisley, "Clearing the skies: A deep network architecture for single-image rain removal," *IEEE Transactions on Image Processing*, vol. 26, pp. 2944–2956, April 2017.

[30] D. Ren, W. Zuo, Q. Hu, P. Zhu, and D. Meng, "Progressive image deraining networks: A better and simpler baseline," in *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition*, June 2019, pp. 3937–3946.

[31] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, June 2016, pp. 770–778.

[32] E. Khatab, A. Onsy, M. Varley, and A. Abouelfarag, "A lightweight network for real-time rain streaks and rain accumulation removal from single images captured by AVS," *Applied Sciences*, vol. 13, December 2022.

[33] K. Wang, L. Chen, T. Wang, Q. Meng, H. Jiang, and L. Chang, "Image deraining and denoising convolutional neural network for autonomous driving," in *Proc. 2021 International Conference on High Performance Big Data and Intelligent Systems (HPBD&IS)*, December 2021, pp. 241–245.

[34] X. Shi, Z. Chen, H. Wang, D.-Y. Yeung, W.-K. Wong, and W.-C. Woo, *Convolutional LSTM Network: A Machine Learning Approach for Precipitation Nowcasting*, MIT Press, September 2015.

[35] A. L. Maas, A. Y. Hannun, and A. Y. Ng. "Rectifier nonlinearities improve neural network acoustic models," in *Proc. International Conference on Machine Learning (ICML)*, 2013.

[36] J. Qi, J. Du, S. M. Siniscalchi, X. Ma, and C.-H. Lee, "On mean absolute error for deep neural network based vector-to-vector regression," *IEEE Signal Processing Letters*, vol. 27, pp. 1485–1489, August 2020.