An Improved Lane-Keeping Controller for Autonomous Vehicles Leveraging an Integrated CNN-LSTM Approach

Hoang Tran Ngoc, Phuc Phan Hong, Nghi Nguyen Vinh, Nguyen Nguyen Trung,
Khang Hoang Nguyen, Luyl-Da Quach
Software Engineering Department, FPT University, Cantho City, Vietnam

***Abstract*—Representing the task of navigating a car through traffic using traditional algorithms is a complex endeavor that presents significant challenges. To overcome this, researchers have started training artificial neural networks using data from front-facing cameras, combined with corresponding steering angles. However, many current solutions focus solely on the visual information from the camera frames, overlooking the important temporal relationships between these frames. This paper introduces a novel approach to end-to-end steering control by combining a VGG16 convolutional neural network (CNN) architecture with Long Short-Term Memory (LSTM). This integrated model enables the learning of both the temporal dependencies within a sequence of images and the dynamics of the control process. Furthermore, we will present and evaluate the estimated accuracy of the proposed approach for steering angle prediction, comparing it with various CNN models including the Nvidia classic model, Nvidia model, and MobilenetV2 model when integrated with LSTM. The proposed method demonstrates superior accuracy compared to other approaches, achieving the lowest loss function. To evaluate its performance, we recorded a video and saved the corresponding steering angle results based on human perception from the robot operating system (ROS2). The videos are then split into image sequences to be smoothly fed into the processing model for training.**

Keywords*—*End-to-end steering control; convolutional neural network; LSTM; nvidia model; MobileNetv2; VGG16

I. Introduction

For over a decade, autonomous driving techniques have captured significant attention from both academic and industrial research and development sectors. During the initial phases of autonomous driving research, the predominant strategies employed were rule-based, primarily focused on image processing. In these approaches, perception, and control were treated as distinct functional modules, operating independently from each other [1]-[9]. However, with the advent of deep learning technologies, there has been a notable shift towards end-to-end vehicle control as a leading research area in autonomous driving [10]-[12]. This approach integrates perception and control into a seamless system, leveraging the power of deep learning to optimize autonomous driving performance.

In 2016, Nvidia introduced the pioneering end-to-end driving model for steering angle control [13]. This model employs Convolutional Neural Networks (CNN) to directly predict the steering angle using raw pixel data from a single frame obtained from a front-view camera. Subsequently, other research studies emerged, exploring various CNN architectures like MobileNetV2, ResNet50, and VGG16, with the aim of enhancing the accuracy and speed of steering angle estimation. These papers are presented with different definitions. [14]. However, these end-to-end driving models have neglected temporal information by focusing solely on individual frames.

In recent years, LSTM has been considered and incorporated into the CNN structure to learn continuous information from the image sequences of the past. Eraqi et al. [15] presented a C-LSTM (CNN with Long Short-Term Memory) model that captures both visual and dynamic temporal dependencies in driving. By incorporating both a CNN and an LSTM network, this model utilizes multiple frames from the front-facing camera input to estimate the steering angle. In a similar vein, Xu et al. [16] proposed an end-to-end architecture called FCN-LSTM, which not only predicts the steering angle but also aims to understand the scene simultaneously. In addition, Yang et al. [17] proposed a multi-modal multi-task network that takes an end-to-end approach and aims to simultaneously predict the steering angle and speed. However, the utilization of the conventional combination of CNN and LSTM in these methods limits their accuracy. With the continuous development and progress of processing hardware, CNN architectures with millions of parameters have been developed and successfully employed to achieve higher accuracy. In light of this, we present a novel approach in this paper by integrating the VGG16 model with LSTM to enhance the estimation accuracy. We leverage the relevant information from input image sequences to improve performance. Through a comparison with traditional methods, we demonstrate the exceptional accuracy achieved by our proposed approach. It is important to note that the implementation of this model will be carried out in a ROS2 simulation environment.

The structure of this paper is outlined as follows: Section II presents an overview of the proposed method. In Section III, we provide a detailed explanation of the CNN-LSTM architectures incorporated into our proposed model. Section IV introduces the experimental system, dataset, evaluation metrics, and the corresponding results, followed by a comprehensive discussion. Finally, Section V concludes the paper, summarizing the key findings and contributions.

II. Proposed Method Overview

The system being developed within the ROS2 robot simulation environment comprises a vehicle model equipped with an RGB camera mounted on the front chassis. Our proposed synthetic neural network, which integrates VGG16 and LSTM networks, is utilized to estimate the steering angle based on input from the camera. The VGG16 model processes each frame of the camera image individually, extracting relevant features. These features are then fed into the LSTM network to capture temporal dependencies, as explained in the next section. The steering angle prediction is obtained from the output classifier following the LSTM layers. Upon completing the training process, the model will be saved and applied for testing on the vehicle model.

To train the proposed model network, we utilize the VGG16 architecture for feature extraction by including the current image (t) and the four preceding images from (t-1) to (t-4). This creates a sequence of 5 images captured within one 0.16s, which will serve as a sample sequence. The LSTM network will then analyze the temporal relationships among these images to estimate the steering angle based on contextual information. During training, the estimated steering angle at the time (t) will be compared with the corresponding ground-truth steering angle at the time (t), and the error will be used in the backpropagation algorithm [18] to update the model's parameters. During the training phase and after saving the model, the proposed system can be visualized through a block diagram, as shown in Fig. 1. This diagram outlines the flow of data and processes involved in the system's operation during both the training and running phases.

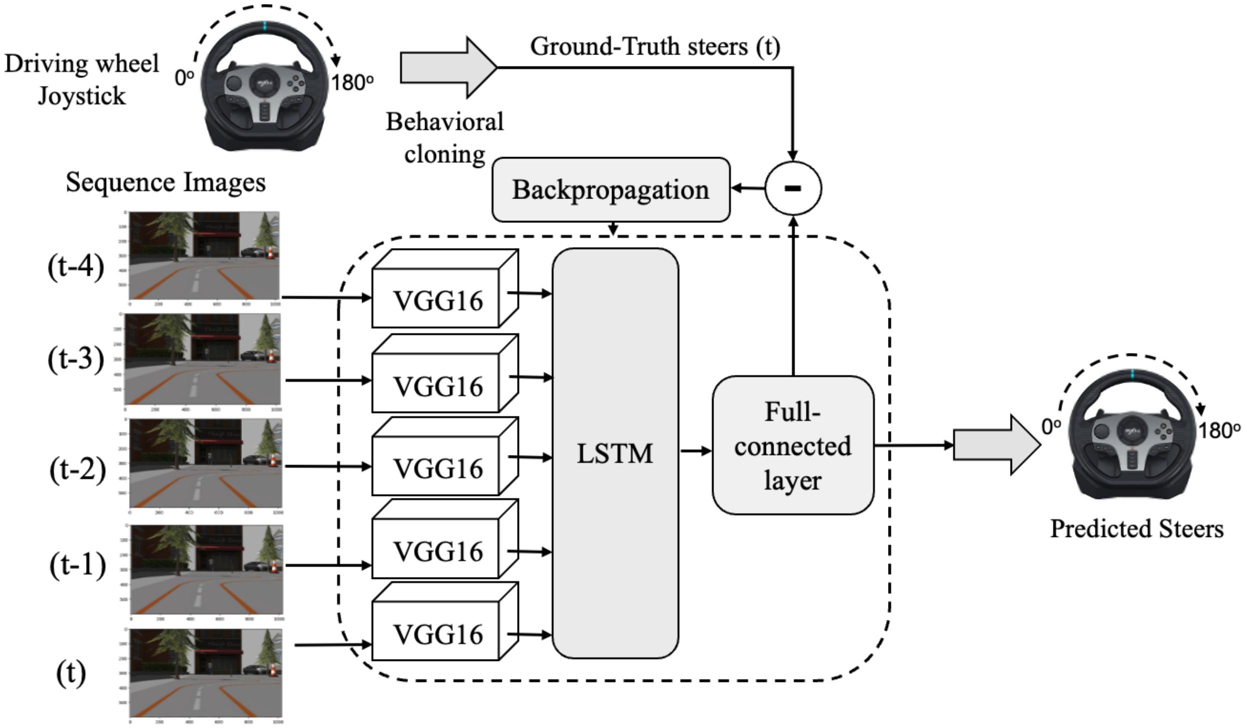


Fig. 1. Block diagram of the proposed system.

III. CNN-LSTM Architectures

In this section, we will introduce the CNN architecture and its integration with LSTM to extract relevant features and combine them over time. This integration allows us to process the input data in a sequential manner and pass it through a fully connected layer to obtain the predicted steering angle.

A. Nvidia CNN-LSTM Model

The Nvidia model, introduced by Nvidia [10], utilizes convolutional neural networks (CNNs) and is specifically engineered to predict the steering angle by processing raw pixel information obtained from a front-facing camera. This model takes advantage of the visual information captured by the camera to directly predict the appropriate steering angle for autonomous driving. By training on a large dataset of images and corresponding steering angles, the Nvidia model learns to extract relevant features from the images and make accurate predictions. We have separated the convolutional feature map of the Nvidia model. Then, we integrated it with LSTM, as shown in Fig. 2, to process the image data sequence by first extracting the features individually before reassembling them using LSTM.

B. MobileNetV2-LSTM Model

The MobileNetV2 model is a lightweight CNN architecture that focuses on efficient computation [19]-[20]. It utilizes depthwise separable convolutions, which divide the convolutional operation into separate depthwise and pointwise convolutions.

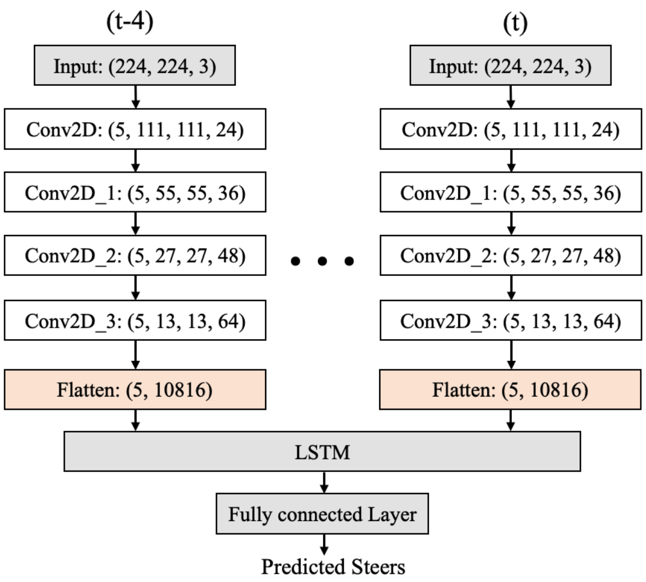

(t-4)
(t)
Input: (224, 224, 3)
Conv2D: (5, 111, 111, 24)
Conv2D_1: (5, 55, 55, 36)
Conv2D_2: (5, 27, 27, 48)
Conv2D_3: (5, 13, 13, 64)
Flatten: (5, 10816)
• • •
LSTM
Fully connected Layer
Predicted Steers

Fig. 2. Nvidia CNN-LSTM structure.

This approach reduces the number of parameters and computational complexity, making it suitable for resource-constrained environments such as mobile devices or embedded systems. The MobileNetV2 model is also trained on image sequences and corresponding steering angles to learn the relationship between visual inputs and steering control. However, we have replaced MobileNetV2 with the Nvidia CNN model to extract features from the input image sequence. The architecture of MobileNetV2 integrated with LSTM and a fully connected layer is depicted in Fig. 3. We will proceed with training on the dataset obtained from driving videos in the ROS2 environment to evaluate the results.

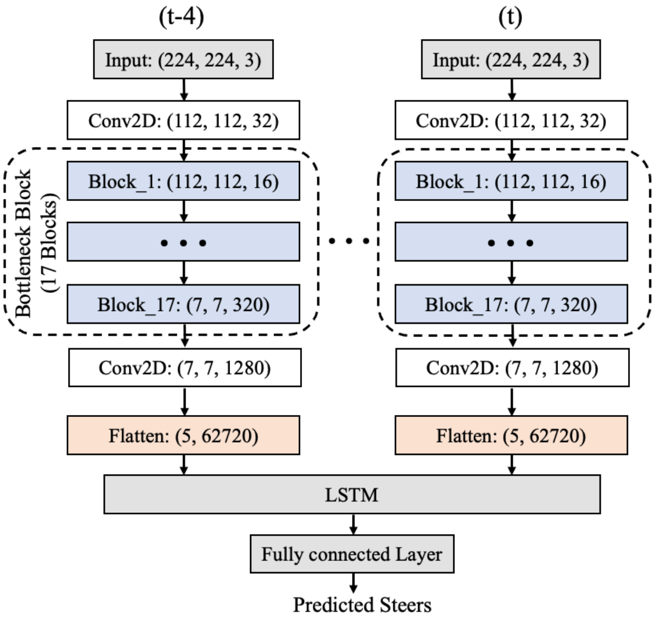


Fig. 3. MobileNetv2-LSTM structure.

C. Proposed VGG16-LSTM Model

The proposed VGG16-LSTM driving model, presented in Fig. 4, consists of two main components: the feature-extracting network and the steering angle prediction network. In this model, the input comprises the previous five frames, ranging from frame t-4 to frame t, which serve as inputs for the driving model.

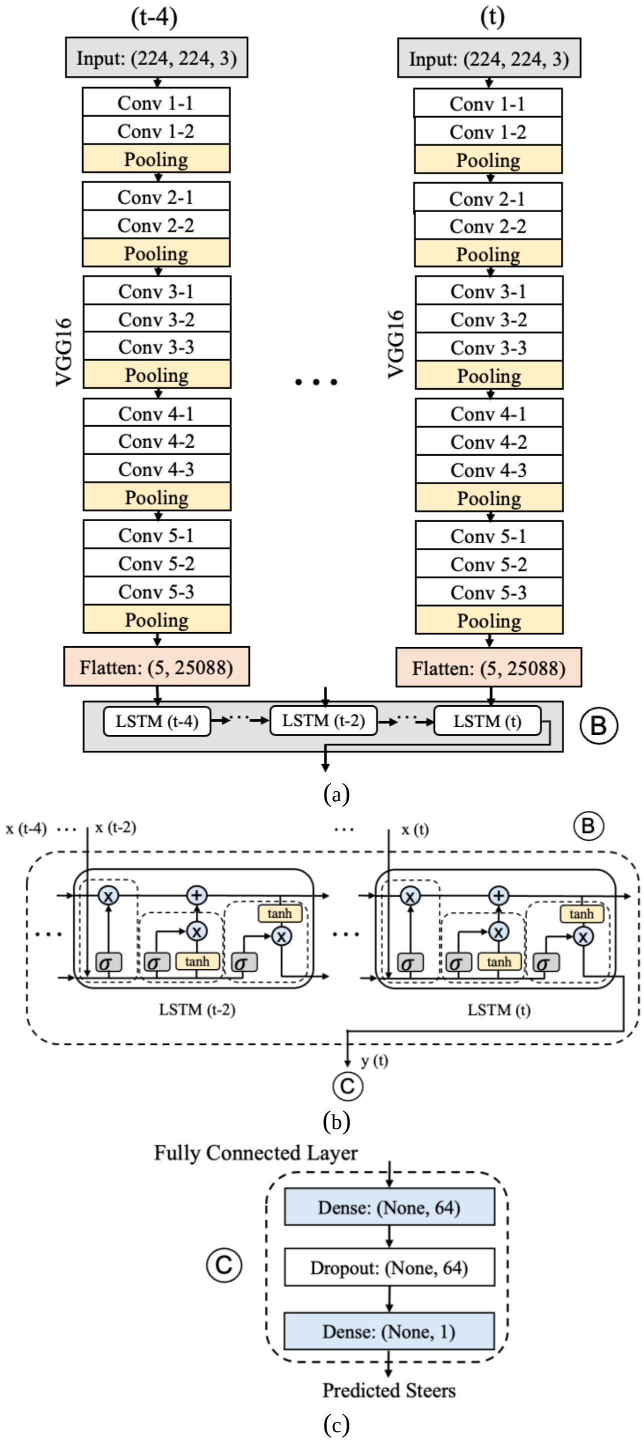


Fig. 4. Proposed VGG16-LSTM structure.

In this study, the VGG16 architecture [21]-[22] is utilized as the feature extraction component. VGG16 is composed of several convolutional layers and pooling layers, which are employed to extract relevant features from images.

The network receives a sequence of five images, each having dimensions of 224x224x3, as input. These images are processed through the feature extraction layers of VGG16, resulting in a feature map with a predetermined size. Following that, a Flatten layer is utilized to convert the output of VGG16 into a vector shape. This vector will be fed into an LSTM network to store temporal information. The LSTM network will handle sequential data and retain previous information for estimating the steering angle.

In the LSTM model with multiple inputs and one output, the network takes in 5 input vectors corresponding to x(t-4), x(t-3), x(t-2), x(t-1), and x(t) as shown in Fig. 4(b). These vectors represent past temporal information. The LSTM model is designed to process and analyze sequential data. To accomplish this, the LSTM model utilizes activation functions, specifically the sigmoid function and the hyperbolic tangent (tanh) function. The sigmoid function is used for the input, forget, and output gates, ensuring controlled information flow within the model. On the other hand, the tanh function facilitates the storage and updating of continuous-valued information within the memory cell.

The output y(t) of the LSTM model is further processed by passing it through the final layers, which consist of two fully connected layers and one dropout layer as shown in Fig. 4(c). These layers contribute to refining the predicted steering angle estimation. The fully connected layers serve as a mapping function, transforming the input from the LSTM into a suitable output format for the desired steering angle estimation.

Every neuron is interconnected with all the neurons in the preceding layer, enabling the learning of intricate relationships and patterns. In the following chapter, we will proceed with the training and compare the accuracy of these models.

IV. Experimental Results

A. Experimental Setting

The virtual environment, illustrated in Fig. 5, is constructed and designed using the Gazebo/ROS2 software. Within this simulation world, a donkey car and a two-lane map are present, serving as the training and testing environments for self-driving mode. Human experts control the donkey car through joystick input, and the captured images are saved and used for training the proposed models. The training process utilized a dataset comprising 10,000 images and was executed on a computer equipped with macOS Ventura 13.4, an ARM-based M2 CPU, a 10-core GPU, and 32 GB RAM. Our algorithm was implemented in Python 3.10, utilizing the Tensorflow 2.12.0 and Keras 2.12.0 libraries. The optimizer used in this study is ADAM [23]. For the experiments conducted, the initial learning rate is set to 0.001.

The VGG16-LSTM model has a total of 45,994,177 parameters, including both trainable and non-trainable ones. Out of these, 40,128,641 parameters are trainable, representing weights and biases that will be updated during training, while 5,865,536 parameters are non-trainable and remain fixed. By freezing the last 8 layers of the VGG16 base with a "trainable" attribute set to False, their weights and biases are preserved, leveraging pre-trained knowledge from the ImageNet dataset. The architecture and parameters of our proposed steering angle prediction model, which was based on transfer learning using VGG16-LSTM, are shown in Fig. 6.

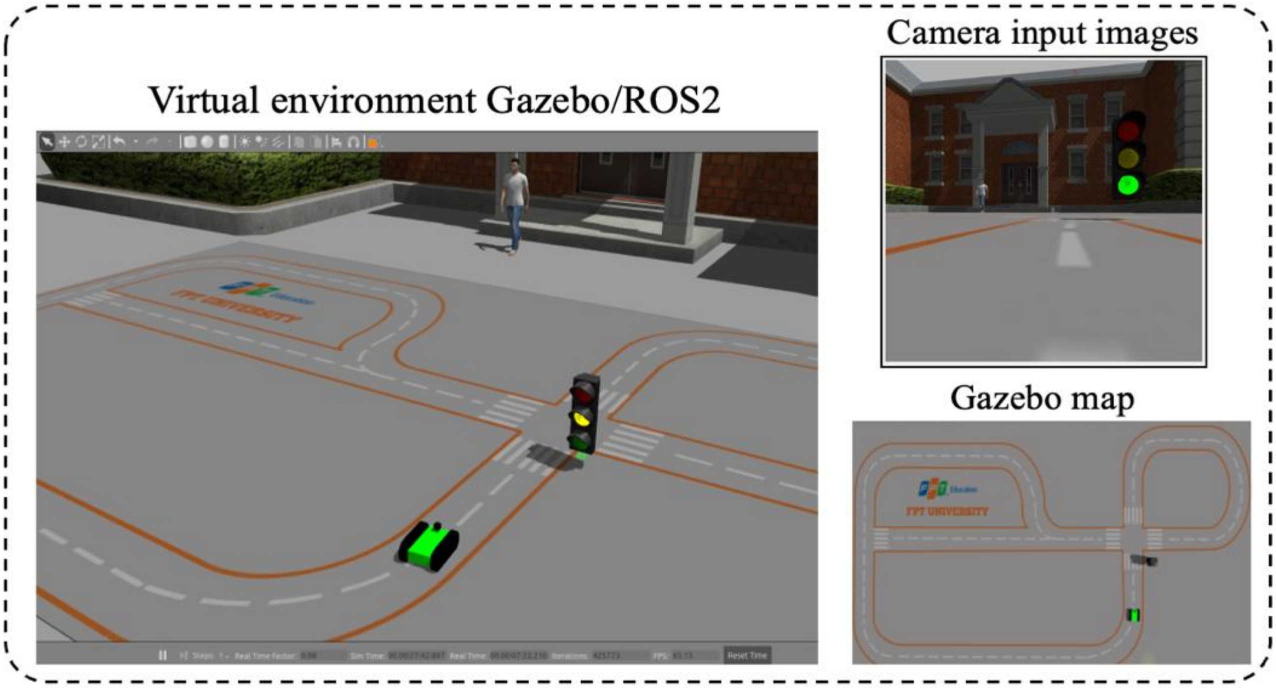
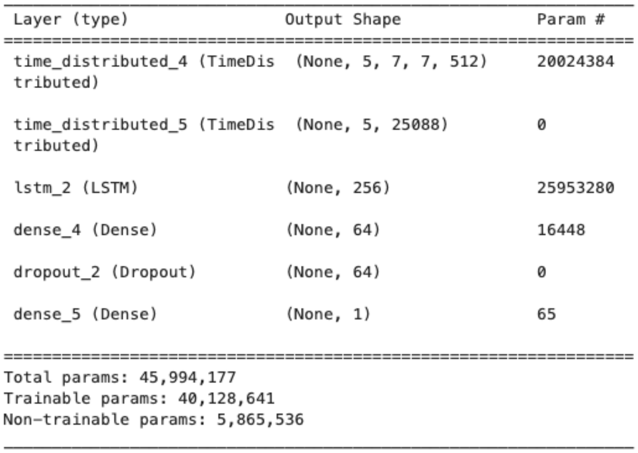


Fig. 5. Simulated environment created using Gazebo/ROS2.



```
_________________________________________________________________
Layer (type)                Output Shape              Param #
=================================================================
time_distributed_4 (TimeDis  (None, 5, 7, 7, 512)     20024384
tributed)

time_distributed_5 (TimeDis  (None, 5, 25088)         0
tributed)

lstm_2 (LSTM)               (None, 256)               25953280

dense_4 (Dense)             (None, 64)                16448

dropout_2 (Dropout)         (None, 64)                0

dense_5 (Dense)             (None, 1)                 65

=================================================================
Total params: 45,994,177
Trainable params: 40,128,641
Non-trainable params: 5,865,536
_________________________________________________________________
```

Fig. 6. Proposed model's architecture based on VGG16-LSTM.

B. *Dataset and Evaluation Metrics*

1) Dataset: The vehicle is equipped with a front-facing camera that captures images, and the ROS2 controller records both the steering angle from the joystick and the corresponding camera images. The data is collected at a rate of 30 frames per second, resulting in five consecutive frames captured within a duration of 0.16 seconds. For this particular study, the dataset used consists of 10,000 images, which is equivalent to 2000 sequences of images. Each sequence contains five consecutive images. The dataset is continuously collected along with the corresponding steering angle information, which is obtained through human perception.

To access the dataset used in this study, you can visit the following link: (https://www.kaggle.com/datasets/ngochoangtran1992/steering-angle-prediction). The input images have a size of 1024x600 before being fed into the training model, and they undergo normalization to align with the input size of VGG16. Fig. 7 illustrates a sequence of five images, showing them before and after normalization, which prepares them for training.

2) Evaluation metrics: During the training process for estimating steering angles using a VGG16 model combined with LSTM, the Mean Squared Error (MSE) equation (4) is commonly used as the loss function and evaluation metric. The MSE measures the average squared difference between the predicted steering angles and the ground truth values provided by human perception.

$$MSE(\hat{y}[t], y[t]) = \frac{1}{N}\sum_{i=1}^{N}(\hat{y}[t]_i - y[t]_i)^2 \tag{4}$$

where $\hat{y}[t]_i$ and $y[t]_i$ are the predicted and true steering angles at the current time and the i^{th} sequence.

By minimizing the MSE loss, the model aims to accurately estimate the steering angles, ultimately improving the alignment between the predicted and actual values, as perceived by humans.

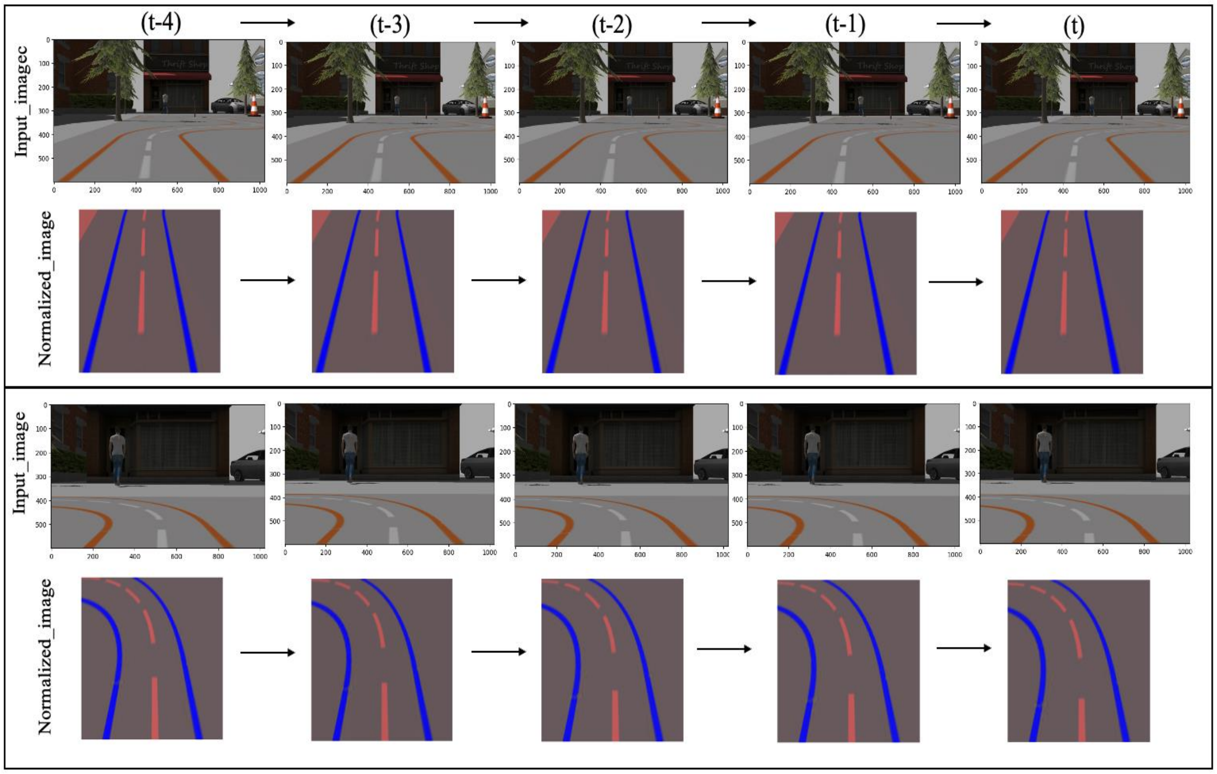


Fig. 7. Sequences of five images before and after normalization of our datasets.

3) Results: The results and comparison of four models, namely Nvidia-CNN, CNN-LSTM, MobileNetv2-LSTM, and the Proposed method (VGG16-LSTM), were evaluated based on their loss values and validation loss values. The Nvidia-CNN model achieved a loss value of 314.03 and a validation loss value of 1268.70. The CNN-LSTM model obtained a loss value of 198.95 and a validation loss value of 479.37. The MobileNetv2-LSTM model demonstrated a loss value of 164.37 and a validation loss value of 244.32. Lastly, the Proposed method (VGG16-LSTM) outperformed the other models with a loss value of 65.07 and a validation loss value of 198.08. These results indicate that the Proposed method (VGG16-LSTM) achieved the lowest loss values, both in training and validation. This suggests that the VGG16-LSTM model performs better in estimating the steering angles compared to the other models, as it exhibits significantly lower loss values. The decrease in loss values indicates a stronger correlation between the predicted and actual steering angles, demonstrating enhanced precision and effectiveness in estimating steering angles. The comparison results of the models are depicted in Table I, and Fig. 8 and Fig. 9 visualize the training outcomes and accuracy assessment of these models throughout 20 epochs.

TABLE I. EXPERIMENTAL RESULTS OF THE PROPOSED MODEL AND OTHER COMPARATIVE METHODS

Model	*Nvidia-CNN*	*CNN-LSTM*	*MobileNetV2-LSTM*	*Proposed Model*
Loss	314.03	198.95	164.37	65.07
Val_Loss	1268.70	470.37	244.32	198.08

Fig. 10 shows the comparison between the steering angle predictions of various models with the proposed approach. It is readily apparent that the proposed method's steering angle predictions exhibit an accuracy level of approximately 95% when compared to the ground truth values. Achieving this level of accuracy involves implementing a method that utilizes a series of input images and incorporates information from previous frames to estimate the steering angle.. The VGG16 and LSTM networks are utilized to extract significant features, ensuring the highest possible precision in the predictions.

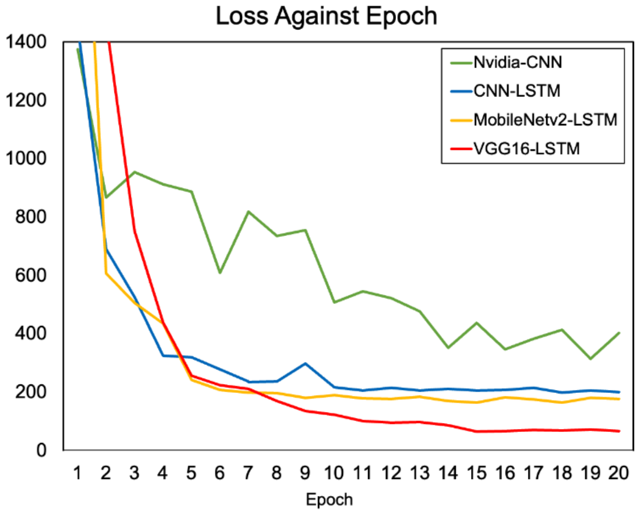


Fig. 8. Comparison of loss values between models architecture.

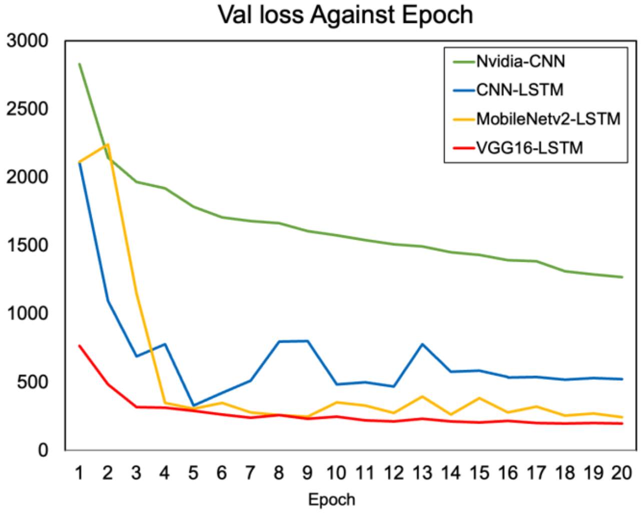
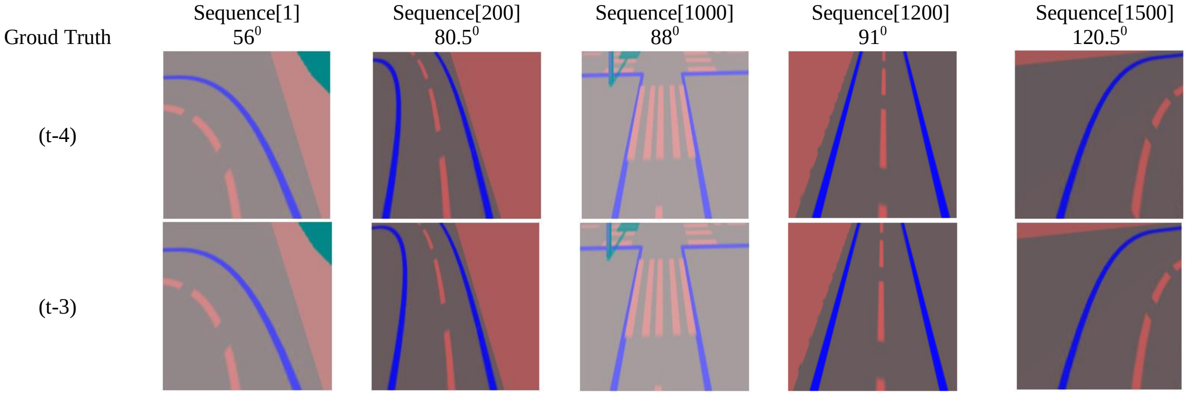


Fig. 9. Comparison of validation loss values between models architecture.



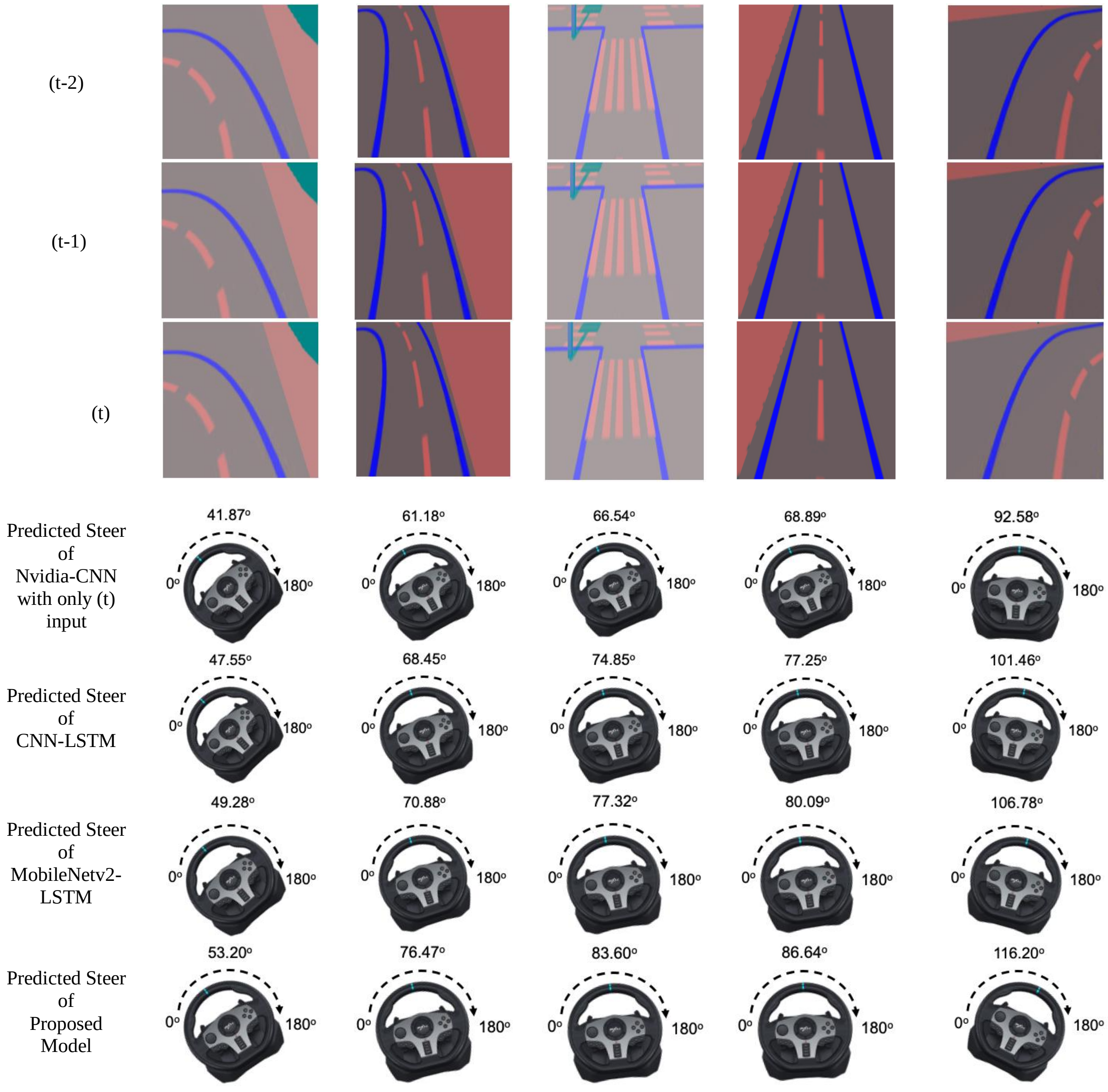

(t-2)
(t-1)
(t)
Predicted Steer of Nvidia-CNN with only (t) input
41.87°
61.18°
66.54°
68.89°
92.58°
Predicted Steer of CNN-LSTM
47.55°
68.45°
74.85°
77.25°
101.46°
Predicted Steer of MobileNetv2-LSTM
49.28°
70.88°
77.32°
80.09°
106.78°
Predicted Steer of Proposed Model
53.20°
76.47°
83.60°
86.64°
116.20°
0°
180°

Fig. 10. Compare the predicted steering angle of the models with the proposed method.

V. Conclusion

By combining VGG16 CNN and LSTM, our proposed approach successfully captures the temporal aspects of visual information and the dynamics of control. This integration sets it apart from other models, as it achieves an exceptional accuracy rate of approximately 95% when predicting steering angles. The results obtained in the ROS2 simulation environment are highly promising, suggesting significant potential for practical applications. This advancement represents a substantial improvement in the precision and dependability of autopilot systems, enhancing their ability to navigate real-life scenarios with greater accuracy and reliability.

References

[1] H. N. Tran, and L. Quach, "Adaptive Lane Keeping Assist for an Autonomous Vehicle based on Steering Fuzzy-PID Control in ROS," International Journal of Advanced Computer Science and Applications; West Yorkshire Vol. 13, Iss. 10, 2022.

[2] M. Montemerlo, J. Becker, S. Bhat, et al., "Junior: The Stanford Entry in the Urban Challenge", Journal of Field Robotics, 25(9):569-597, 2008.

[3] J. Leonard, J. How, S. Teller, et al., "A Perception-Driven Autonomous Urban Vehicle", Journal of Field Robotics, 25(10):727-774, 2008.

[4] A. Gurghian, T. Koduri, S. V. Bailur, et al., "Deeplanes: End-to-End Lane Position Estimation using Deep Neural Networks", In IEEE International Conference on Computer Vision and Pattern Recognition (CVPR), pp 38-45, 2016.

[5] H. T. Vo, H. N. Tran, and L. Quach, "An Approach to Hyperparameter Tuning in Transfer Learning for Driver Drowsiness Detection Based on Bayesian Optimization and Random Search" International Journal of Advanced Computer Science and Applications(IJACSA), 14(4), 2023.

[6] P. H. Phan, A. Q. Nguyen, L. Quach, and H. N. Tran. 2023. “Robust Autonomous Driving Control using Auto-Encoder and End-to-End Deep Learning under Rainy Conditions”. In Proceedings of the 2023 8th International Conference on Intelligent Information Technology (ICIIT '23). Association for Computing Machinery, New York, NY, USA, 271–278.

[7] H. K. Hua, K. H. N., L. Quach, and H. N. Tran. 2023. “Traffic Lights Detection and Recognition Method using Deep Learning with Improved YOLOv5 for Autonomous Vehicle in ROS2”. In Proceedings of the 2023 8th International Conference on Intelligent Information Technology (ICIIT '23). Association for Computing Machinery, New York, NY, USA, 117–122.

[8] J. Janai, F. Gney, A. Behl, et al., “Computer Vision for Autonomous Vehicles: Problems, Datasets, and State-of-the-Art”, arXiv preprint, arXiv:1704.05519, 2017.

[9] V. D. Nguyen, T. D. Trinh and H. N. Tran, "A Robust Triangular Sigmoid Pattern-Based Obstacle Detection Algorithm in Resource-Limited Devices," in IEEE Transactions on Intelligent Transportation Systems, vol. 24, no. 6, pp. 5936-5945, June 2023.

[10] D. A. Pomerleau, “Alvinn: An Autonomous Land Vehicle in a Neural Network”, In Advances in Neural Information Processing Systems, pp 305-313, 1989.

[11] C. Chen, A. Seff, A. Kornhauser, et al., “Deepdriving: Learning Affordance for Direct Perception in Autonomous Driving”, In IEEE International Conference on Computer Vision (ICCV, pp 2722-2730), 2015.

[12] M. Bojarski, P. Yeres, A. Choromanska, et al., “Explaining How a Deep Neural Network Trained with End-to-End Learning Steers a Car”, arXiv preprint, arXiv:1704.07911, 2017.

[13] M. Bojarski, D. Testa, D. Dworakowski, et al., “End to End Learning for Self-Driving Cars”, arXiv preprint, arXiv:1604.07316, 2016.

[14] U. M. Gidado, H. Chiroma, N. Aljojo, S. Abubakar, and S. I. Popoola, “A survey on deep learning for steering angle prediction in autonomous vehicles,” IEEE Access, vol. VIII, pp. 163797-163817, 2020.

[15] H. M. Eraqi, M. N. Moustafa, J. Honer, “End-to-End Deep Learning for Steering Autonomous Vehicles Considering Temporal Dependencies”, arXiv preprint, arXiv:1710.03804, 2017.

[16] H. Xu, Y. Gao, F. Yu, et al., “End-to-end Learning of Driving Models from Large-scale Video Datasets”, In IEEE International Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp 2174-2182.

[17] 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 IEEE International Conference on Pattern Recognition (ICPR), 2018, pp 2289-2294.

[18] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by backpropagating errors. Cognitive modeling, 5(3):1, 1988.

[19] M. Gupta, V. Upadhyay, P. Kumar, and F. Al-Tụman, “Deep Learning Implementation of Autonomous Driving using Ensemble-M in Simulated Environment”, Research Square, May 2021.

[20] S. Du, H. Guo, and A. Simpson, “Self-Driving Car Steering Angle Prediction Based on Image Recognition”, ArXiv, vol. abs/1912.05440., 2019.

[21] A. Oussama and T. Mohamed, “A Literature Review of Steering Angle Prediction Algorithms for Self-driving Cars”, Int. Conf. on Advanced Intelligent Systems for Sustainable Development, vol 1105, Feb. 2020.

[22] J. Sokipriala, “Prediction of Steering Angle for Autonomous VehiclesUsing Pre-Trained Neural Network”, European Journal of Engineering and Technology Research, August 2021.

[23] D. P. Kingma, and J. Ba, “Adam: A Method for Stochastic Optimization”, in 3rd International Conference for Learning Representations, 2015.