Comparing Traditional Computer Vision Algorithms and Deep Convolutional Neural Networks as Self Driving Algorithms for Use in Dynamic Conditions

Shilpi Shah*
Dept. of Computer Science
Rutgers University
New Brunswick, NJ, USA
sps247@scarletmail.rutgers.edu

Brendan Franz*
Dept. of Computer Science
Harvard University
Cambridge, MA, USA
brendanfranz@college.harvard.edu

Travis Forgach*
Dept. of Computer Science and Engineering
University of Michigan
Ann Arbor, MI, USA
tforgach@umich.edu

Milan Jostes*
Dept. of Math and Computer Science
Lawrence Technological University
Southfield, MI, USA
mjostes@ltu.edu

Joshua Siegel
Dept. of Computer Science and Engineering
Michigan State University
East Lansing, MI, USA
jsiegel@msu.edu

Chan-Jin Chung
Dept. of Math and Computer Science
Lawrence Technological University
Southfield, MI, USA
cchung@ltu.edu

*All authors contributed equally.

***Abstract*—This research develops, compares, and analyzes both a traditional algorithm using computer vision and a deep learning model to deal with dynamic road conditions. In the final testing, the deep learning model completed the target of five laps for both the inner and outer lane, whereas the computer vision algorithm only completed almost three laps for the inner lane and slightly over four laps for the outer. After conducting statistical analysis on the results of our deep learning model by finding the p-value between the absolute error and squared error of the self driving algorithm in the outer lane and inner lane, we find that our results are statistically significant based on a two-tailed T test with unequal variances where the p-value for absolute error is 0.009, and 0.001 for squared error. Self-driving vehicles are not only complex, but they are growing in necessity—therefore, finding an optimal solution for lane detection in dynamic conditions is crucial to continue innovation.**

Keywords—Deep learning, self-drive, convolutional neural network, ROS, computer vision

I. Introduction

As self-driving vehicles become increasingly common in society due to various reasons such as environmental conscientiousness, rapidly advancing technology, and wide-spread availability, it becomes significantly more important for these vehicles to be able to respond appropriately to diverse events. Currently, most widely available self-drive technologies require human assistance to ensure robustness and safety of the vehicles. They are more suited to highway driving compared to driving in rural and narrow roads with frequent obstacles such as pedestrians and mailboxes. This paper considers the importance of the ability of self-driving vehicles to maintain accurate lane keeping in dynamic conditions, such as worn-out lines, harsh lighting, and inclement weather.

As the field of automated vehicles rapidly expands, researchers have developed five levels of classification known as the SAE J3016 Levels of Driving Automation [1]: Level 0: No Automation; Level 1: Assisted Driving; Level 2: Partial Automation; Level 3: Conditional Automation; Level 4: High Automation; Level 5: Full Automation.

We propose a model utilizing deep learning convolutional neural networks (CNNs). This would be classified as a level 2 vehicle. This CNN model is tested and trained on images and linked yaw messages compiled from rosbags, a file format available by Robotic Operating System (ROS) to store message data. After extracting images from the rosbag, the model resizes and preprocesses images to enhance accuracy, randomly splits given data into testing and training data, and displays input images following preprocessing and graphs depicting mean standard error loss rates and mean average error (MAE). When launched with a code file that connects the deep learning model to the vehicle, the vehicle is able to drive on its own by predicting the yaw rate at which to turn the vehicle.

To compare with a deep learning model, we also propose a traditional algorithm using computer vision—which is classified as a level 2 vehicle. The handcrafted algorithm uses Python and ROS libraries, as well as extensive image preprocessing, to be able to adapt to dynamically changing weather conditions. Following preprocessing, the algorithm extends dashed white

lines in their primary directions in order to create more clear lanes for the self-driving vehicle to follow.

In this paper, we benefit from deep learning models, particularly convolutional neural networks using TensorFlow, Keras, and sci-kit learn. The model splits the images into random testing and training data. Then, the model is tested and trained on that data. The model is constructed using TensorFlow and Keras layers and converts the data into tensors—a multidimensional array that is compatible with TensorFlow. TensorFlow, Keras, and sci-kit learn are model libraries in Python that are downloaded as extensions.

We also benefit from a comparison between the handcrafted algorithm with the use of various image preprocessing techniques using the OpenCV library in Python, such as blurring, Gaussian transformations, and noise reduction. These techniques are utilized on visual data received from the camera of the vehicle, relying heavily on computer vision.

II. Literature Review

As deep learning has rapidly developed over recent years, there has been great success in artificial intelligence advancements and implementations in a broad range of fields, where self-driving is often a primary one. However, there remains much work to be done as many current models have various flaws and no model has successfully implemented level 5 self-driving.

A. Recurrent Neural Networks

Despite having been controversial for quite some time, highly automated and/or assisted driving has become increasingly popular. Some research demonstrates that recurrent neural networks (RNNs) are the optimal model for self-driving. This structure can be used to demonstrate steering wheel dynamics by predicting steering angle and steering rate based on applied steering torque using the first principles model to integrate the torque [2]. This network is created to improve a steering Proportional Integral Derivative (PID) controller [3], a type of controller used for automated steering, which computes the necessary torque input to be able to track steering trajectory [2].

Additionally, when paired with a Graph Neural Network (GNN), RNNs are able to predict vehicle trajectories by extracting previously acquired information using long short-term memory networks (LSTM) [4], which are enhanced RNNs with the ability to track information backwards in time [5].

B. Convolutional Neural Networks

While recurrent neural networks are often used, even more popular than these are convolutional neural networks (CNNs). The most frequent approach to deep learning self drive algorithms is a combination of these two algorithms, in what is called a recurrent convolutional neural network. The recurrent convolutional neural network is the most complex, but typically the most efficient solution.

CNNs are, however, still a powerful tool independently for vehicle automation. They are commonly used to train vehicles to drive autonomously using image preprocessing and computer vision. Therefore, making them a strong tool for isolating contextual information from training data [6]. Additionally, this makes CNNs extremely powerful in conjunction with end-to-end learning methods [7] in order to predict steering angles [8]. A commonly used CNN model is one that maps raw pixel data from cameras [9] and translates them directly into steering commands, as done by researchers at Cornell [10]. Furthermore, CNNs have shown great potential in not only steering command accuracy, but with overall driving techniques, including U-turns, lane changes, and parking by using image processing. In this research, the researchers rely on Hough transformations and other techniques, such as Canny edge detection, for image processing, which mainly account for straight lines and not curvatures- this approach is therefore better suited for highway and straight-road driving, over the intricacies of rural driving [11].

Since CNNs are often used with image processing, as aforementioned, there has also been work done on optimizing image processing using semantic segmentation [12], otherwise known as continuous visual signaling processing. This creates frames and analyzes pixel values in real-time. The benefit of semantic segmentation lies in enhanced situational awareness [12], using CNNs for object detection [9]. Semantic segmentation therefore differs from instance segmentation, another segmentation technique which detects each individual instance of an object in an image. Instance segmentation consists of both object detection and semantic segmentation [13].

C. Computer Vision

Previous research has also been completed on research pertaining to computer vision only algorithms. These algorithms used the Hough Transform and hyperbolic fitting as a means to interpret visual data from camera-recorded road data. The algorithm showed promise in processing images of roads with limited curves and straight lines, but there were noted constraints in its lane detection capabilities. These constraints were not only on curvy roads but also when there were adverse lighting conditions [14, 15, 16].

III. Experimental Design

A. Vehicle Details

The test vehicle we use in the experiment is a Polaris Gem e2 provided to Lawrence Technological University by MOBIS. Lawrence Technological University provided Dataspeed's drive-by-wire (DBW) system, a Mako G-319 camera, both 2D and 3D LIDARs, a GPS system, in-vehicle computers, and any other additional hardware.

B. Testing Environments

We began developing a self-driving vehicle by using a traditional, hand-crafted algorithm. Our code was first tested on a programmed simulation of a vehicle in a virtual environment that matches the test course we would be using. We used dynamic reconfigure to produce a toggle bar for threshold value, speed, and a checkbox to enable the self drive.

This work is supported by NSF grants 2150292 and 2150096.

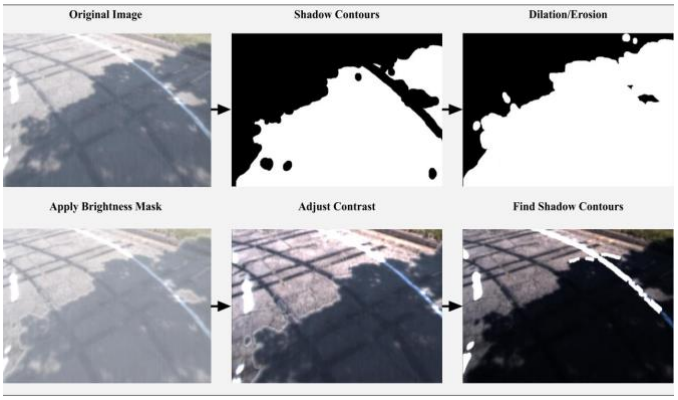
After succeeding in following the lanes of the test course on the simulator, we then updated our code to connect to the car using its DBW system. This was done by publishing a vehicle enable message before publishing a twist message to the vehicle. Once the code was connected to the vehicle, we removed the implementation of the simulation map and replaced it with images from the camera.

C. Traditional Self Driving Algorithm for Lane Keeping

The traditional algorithm was developed to identify lanes using contour detection. This algorithm drives at a constant, predetermined speed. The images undergo preprocessing that first crops the images to have a smaller region of interest. The algorithm will identify yellow lines and draw contours over them to make them more unique from the white lane lines. This masking allows the image to be converted to grayscale for improved contrast of shadows. The program then identifies the shadows on the input image through the use of contours. It then dilates to encompass the lines into the shadow, followed by an erosion to remove the excess edges. The shadow's contour is then used to apply a mask that brightens the shadow slightly. This new image is then processed by a contrast and gamma correction algorithm that helps highlight whites in the image. This process described is illustrated within Fig. 1.

After the preprocessing has finished, the algorithm identifies contours of a certain size to use as lane lines. To account for the poor road conditions and the cracked paint, the program dilates the contours in the image in order to merge shapes that are close together. This allows many of the cracked lines to be identified as a singular shape, rather than a collection of many small dots. The program then identifies which of the contours should be defined as the actual lane. It does this by sorting the contours by their total area. It then takes some of the largest contours and extends them by identifying its center point and relative slope. It also identifies whether the line will be aligned with the expected lane, or if it may be a false line. If the predicted line would be perpendicular to the anticipated lane, the algorithm will not draw the contour.

The algorithm identifies the four largest contours and predicts which line to follow based upon the previous position of these lines. This stops the program from identifying many of the outside variables such as crosswalks and sidewalks. It identifies the center of these lane lines and attempts to align the center of the vehicle with this center point.



This work is supported by NSF grants 2150292 and 2150096.

Fig. 1. Six images depicting shadow removal during image pre-processing. The steps move from left to right, top to bottom (original image, shadow contours, dilation/erosion, apply brightness mask, adjust contrast, find shadow contours).

D. Deep Learning Self Driving Model for Lane Keeping

The deep learning model is a CNN that takes in an image as the input. When training the CNN, the model is given an image and a yaw rate using a CSV file. The image is used to predict a yaw rate and the given yaw rate serves as the label for comparison with the prediction. The CNN itself is comprised of three convolutional layers using Convolutional2D and two dense layers. We then bring in a single flattened layer to convert the two-dimensional array of images into a one-dimensional array.

Before training the model, the region of interest is redefined such that the top five-eighths of the image is removed. Then, the RGB image is converted to a grayscale image and the image is resized to 100 pixels by 100 pixels. All these steps serve the purpose of reducing noise and to eliminate irrelevant parameters that the CNN might use to base its yaw rate prediction on.

After it completes the image preprocessing, the model then uses the scikit-learn model API that was imported to randomly split the data into training and testing data with a 95:5 respective ratio. Once this is complete, the model is trained on the processed data, it is saved as the new trained model. This algorithm drives at a constant, predetermined speed.

The model used in the experiment was trained on data collected in numerous weather conditions. The data included sunny weather conditions with shadows, cloudy weather, rain conditions with water on the front windshield, and sunny weather with overexposed lighting as depicted in Fig. 2.

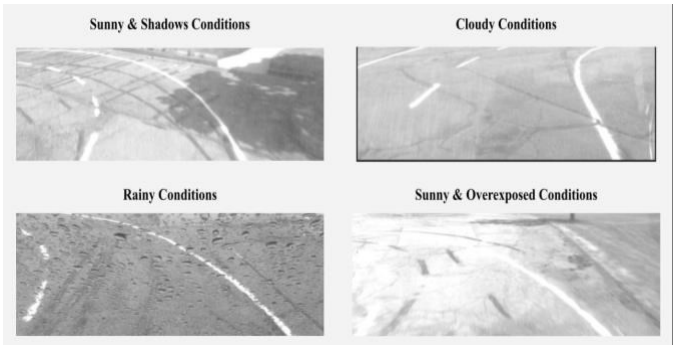


Fig. 2. Four images depicting different weather conditions the CNN was trained on (top left: sunny & shadow conditions; top right: cloudy conditions; bottom left: rainy conditions; bottom right: sunny & overexposed conditions).

IV. Performance Analysis

A. Statistical Analysis of Deep Learning Results

We evaluated the performance of the deep learning model by finding the statistical significance through calculating a p-value and outcome independence through chi-square analysis. We choose to find the p-value as it is a frequently agreed upon statistical analysis test used to accept or reject results of an experimental design. The p-value can assess the confidence level of the deep learning model. Additionally, we use a

Pearson chi-squared analysis test to further assess the accuracy of our prediction model, and further affirm the confidence of the model.

To begin, we calculate the p-value of both the absolute error and squared error for the outer lane and inner lane using a two-tailed T test with unequal variances. We opted to use a two-tailed T test to analyze if the absolute error or squared error is either greater than or lower than the outer lap in comparison to the inner lap. Additionally, we choose to use unequal variances rather than equal variances since the sample size is greater than 30, making it substantially large, and the standard deviation of both the outer lap and inner lap error margins is unknown, therefore assumed as unequal. For absolute error, our model yields a 0.009 p-value when analyzed for over 220 data points, making it statistically significant. Additionally using another 220 data points from the same data extraction process for the squared error, our model yields a 0.001 p-value with a two-tailed T test with unequal variances, once again making it statistically significant by a considerable margin. The results of the T test depict that there is over a 99% chance that there is a relationship between the absolute error or squared error of the deep learning model, and the performance of the self-driving vehicle in either the outer or inner lap.

Furthermore, we perform a Pearson chi-squared test to determine the relationship between 200 data points of predicted values in contrast to actual values. The predicted and actual values are based upon the twist messages connected to each image and the predicted twist value for yaw rate that the model creates. The null hypothesis is that there is no relationship between the predicted values and actual values. We then calculate the chi-squared value for the data points, followed by calculating the critical value with a significance level of 0.05 and 1 degree of freedom, as there are two columns of actual value and predicted values, and two rows of whether they are within a thousandth decimal point of each other or not. The chi-squared value is 543.8, and the critical value is 240.99. Since the chi-squared value is greater than the critical value, we can reject the null hypothesis and therefore prove that there is a significant relationship between the predicted and actual values of the model. This, therefore, indicates that the vehicle is able to consistently maintain accurate lane keeping for a statistical perspective.

B. Performance and Comparison of Models

To analyze the performance of both our traditional and deep-learning models, we analyzed the number of laps around the track the model successfully completes, the distance the model travels if the model fails, and the time it takes to complete all of the laps—which is used to calculate the average time taken per lap. We analyzed the performance over a series of demonstrations, seeing varying results of success between the deep learning and computer vision models.

In the final demonstration, we established the goal number of laps for each model to complete on both the inner and outer lane to be five. The deep learning model succeeded in completing five laps for both the inner and outer lane as depicted in Table I, but the computer vision algorithm failed in both lanes as depicted in Table II. The deep learning model was able to complete the goal number of laps for both the inner and outer lane. The computer vision algorithm, on the other hand, completed just short of three laps for the inner lane and slightly over four laps for the outer lane.

In terms of comparing the time taken to complete laps, we divided the total time taken with the total number of laps in order to find the average time taken by lap. The deep learning model, on average, completed a lap in the inner lane in approximately 1 minute and 22 seconds. It was able to complete a lap in the outer lane in approximately 1.5 minutes. In comparison, the computer vision model completed a lap in the inner lane in approximately 2 minutes on average. It completed a lap in the outer lane in 1 minute and 38 seconds.

Therefore, as visible in Table I and Table II, the deep learning model completed more laps and completed them faster than the computer vision algorithm due to an ability to better detect road lines and turn more accurately based on those lines.

TABLE I. DEEP LEARNING RESULTS

Track Section & Direction	***Laps Completed***	***Time***
Inner Lane & Clockwise	5	6:51
Outer Lane & Counterclockwise	5	7:39

TABLE II. COMPUTER VISION RESULTS

Track Section & Direction	***Laps Completed***	***Time***
Inner Lane & Clockwise	2.9	5:32
Outer Lane & Counterclockwise	4.25	7:04

V. CONCLUSION

Self driving algorithms hold a great deal of significance and potential benefits to society. So, finding an efficient and accurate solution for lane detection in adverse and dynamic conditions is critical. Our traditional handcrafted model was able to complete 4.25 laps and 2.9 laps in the outer and inner lanes respectively and was able to complete such laps at a lower speed compared to our deep learning-based algorithm. Additionally, the computer vision algorithm struggled to adapt to changing lighting conditions during test runs which led it to running off course and resulted in it being unable to complete the target lap goal of 5.

Furthermore, our deep learning model was able to reach the target lap goal for both the inner and outer lanes. The deep learning model was able to deal with changing cloud coverage and moving shadows during testing. Therefore, between the two models that were tested, the deep learning model was more robust for lane keeping and lane detection in various lighting and weather conditions. Since these algorithms are fairly rudimentary, there remains quite a deal of work to be done to reach the maximum protentional of self drive algorithms and to achieve level 5 self drive status. To reach

This work is supported by NSF grants 2150292 and 2150096.

level 5, the car would need the ability to make its own decisions as to destination changes or methods of responding to adverse and dynamic events.

To continue to improve the deep learning model, more data collection and training would be needed. Specifically, data could be collected at higher speeds to allow for strong functionality with a wider range of speeds. In addition, this would allow for the algorithm to drive at various speeds instead of relying on the data collectors to predetermine the speeds at which the model drives. This would need much more training data to become operational, but based on the results of our current model, this seems promising. Furthermore, to guarantee the preservation of functionality at all speeds, it would be advantageous to implement a RNN that makes decisions on both the current image it is evaluating, but also the previous yaw rates that it had predicted. This would create a model that would then know to turn more if it seems to be understeering previous and turn less if it was previously oversteering. For our traditional algorithm, it would be advantageous to improve upon the thresholding measures we had in place in order to deal with the rapidly changing cloud cover and brightness we experienced in testing. Specifically, the thresholding improvements would include an automatically changing threshold value range depending on the weather and lighting conditions. Also, improvements to the vehicle's hardware may be necessary, such as a wide-angle lens to be able to pick up the lanes while in turns. With these improvements, the performance of both models would be promising and could lead to much more stable lane following.

ACKNOWLEDGMENTS

Our work was supported by the National Science Foundation under Grant Nos. 2150292 and 2150096. We thank our mentors, Joe DeRose and Nick Paul, and our graduate assistants, Ryan Kaddis, and Justin Dombecki, for guidance throughout this research project.

REFRENCES

[1] Szikora, P., & Madarász, N. (2017). Self driving cars - the human side. *IEEE 14th International Scientific Conference on Informatics*, *17*(31), 383–387. Retrieved from https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=8327279.

[2] Garimella, G., Funke, J., Wang, C., & Kobilarov, M. (2017). Neural Network Modeling for Steering Control of an Autonomous Vehicle. *IEE/RSJ International Conference on Intelligent Robots and Systems*, *978-1-5386-2682–5*(17), 2609–2615. Retrieved from https://asco.lcsr.jhu.edu/wp-content/uploads/2018/10/neural_network_modeling_steering_control.pdf.

[3] Sang, N., & Chen, L. (2019). Design of an active front steering system for a vehicle using an active disturbance rejection control method. *Sage Journals*, *103*(1). https://doi.org/10.1177/0036850419883565.

[4] Mo, X., Xing, Y., & Lv, C. (2021). Graph and Recurrent Neural Network-based Vehicle Trajectory Prediction For Highway Driving. *Computing Research Repository (CoRR)*, *2107*(03663). Retrieved from https://doi.org/10.48550/arXiv.2107.03663.

[5] Siwei, L., Cheng, H., Rui, W., Chao, Z., & Jing, Y. (2019). A maneuvering tracking method based on LSTM and CS model. *IEEE International Conference on Signal, Information and Data Processing (ICSIDP)*, 1–4. https://doi.org/10.1109/ICSIDP47821.2019.9173187.

[6] Ramos, S., Gehrig, S., Pingerra, P., Franke, U., & Rother, C. (2017). Detecting unexpected obstacles for self-driving cars: Fusing deep learning and geometric modeling. *2017 IEEE Intelligent Vehicles Symposium (IV)*, *2017*(7995849), 1025–1032. https://doi.org/10.1109/IVS.2017.7995849.

[7] Chen, S., Zhang, S., Shang, J., Chen, B., & Zheng, N. (2019). Brain-Inspired Cognitive Model with Attention for Self-Driving Cars. *IEEE Transactions on Cognitive and Developmental Systems*, *11*(1), 13–25. Retrieved from https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=7954050.

[8] Du, S., Guo, H., & Simpson, A. (2020). Self-Driving Car Steering Angle Prediction Based on Image Recognition. *Computing Research Repository (CoRR)*, *1912*(05440), 1–9. Retrieved from https://doi.org/10.48550/arXiv.1912.05440.

[9] Kulkarni, R., Dhavaliker, S., & Bangar, S. (2018). Traffic Light Detection and Recognition for Self Driving Cars Using Deep Learning. *2018 IEEE Fourth International Conference on Computing Communication Control and Automation (ICCUBEA)*, *2018*(8697819), 1–4. https://doi.org/10.1109/ICCUBEA.2018.8697819.

[10] Bojarski, M., Del Testa, D., Dworakowski, D., Firner, B., Flepp, B., Goyal, P., Jackel, L. D., Monfort, M., Muller, U., Zhang, J., Zhang, X., Zhao, J., & Zieba, K. (2016, April 25). *End to end learning for self-driving cars*. arXiv.org. https://arxiv.org/abs/1604.07316.

[11] Garg, N. (2022). Self-Driving Car to Drive Autonomously using Image Processing and Deep Learning. *International Journal of Research in Engineering, Science, and Management (IJRESM)*, *5*(1), 125–132. Retrieved from https://journal.ijresm.com/index.php/ijresm/article/view/1692/1632.

[12] Sellat, Q. et al. (2022). Intelligent Semantic Segmentation for Self-Driving Vehicles Using Deep Learning. *Hindawi Computational Intelligence and Neuroscience*, *2022*(6390260). Retrieved from https://doi.org/10.1155/2022/6390260.

[13] Sharma, R., Saqib, M., Lin, C.T. & Blumenstein, M. A Survey on Object Instance Segmentation. *SN COMPUT. SCI.* **3**, 499 (2022). https://doi.org/10.1007/s42979-022-01407-3.

[14] Assidiq, A.A.; Khalifa, O.O.; Islam, M.R.; Khan, S. (2008). Real time lane detection for autonomous vehicles. *2008 International Conference on Computer and Communication Engineering, Kuala Lumpur, Malaysia*, 13–15 May 2008; pp. 82–88. Retrieved from https://ieeexplore.ieee.org/abstract/document/4580573.

[15] Rao S, Quezada A, Rodriguez S, Chinolla C, Chung C-J, Siegel J. Developing, Analyzing, and Evaluating Vehicular Lane Keeping Algorithms Using Electric Vehicles. Vehicles. 2022; 4(4):1012-1041. https://doi.org/10.3390/vehicles4040055.

[16] R. Kaddis, E. Stading, A. Bhuptani, H. Song, C. -J. Chung and J. Siegel, "Developing, Analyzing, and Evaluating Self-Drive Algorithms Using Electric Vehicles on a Test Course," 2022 IEEE 19th International Conference on Mobile Ad Hoc and Smart Systems (MASS), 2022, pp. 687-692. https://doi.org/10.1109/MASS56207.2022.00101.