1 Introduction

The last decade has seen massive growth in the field of Autonomous Driving primarily due to a) proliferation of graphical processing unit(GPU), b) several projects like Google(Waymo) [1], Berkeley-DeepDrive [2], Apollo [3], have made their datasets open-source making it easier for people to work on these data and achieve better performance gains.

Training a deep neural network(DNN) forms the core in making a car autonomous. By using supervised learning, one can achieve reliable results as it gives greater control at each stage of training. The data-driven approach collects data in advance, labels it appropriately. It can, then, be fed to the DNN using supervised learning algorithms, to train the best model possible.

Ever since Alexnet in 2012 [4], the convolutional neural network(CNN) and deep learning(DL) are preferred choices to analyse images. However, it is well known that the camera sensors are susceptible even to a slight change in weather conditions. Sensors like radar [5], LIDAR [6], ultrasonic, depth camera give additional depth information for obstacle detection. These then are fused with the camera images to make data fusion possible.

Even though there are some public data available, it is still not enough data to reliably train a DNN. Then there is the cost of building an autonomous car. Fortunately, the last years have seen growth in reliable simulators which has helped massively to collect data to help understand this field of research. To name a few simulators that are being actively used – LGSVL [7], Nvidia Drive [8], Carla [9], CarMaker [10]. In this thesis, the LGSVL simulator is used.

The LGSVL simulator allows the usage of different sensors with minimal effort. The data from different sensors are published through websocket. So to capture these data, we need an interface/protocol which can understand the sending data's message types and enable the receiving node/programme to store them. However, the data from each sensor arrive at different rates. Hence it is necessary to collect and synchronise them in the order of their arrival before storing, to not lose their integrity and thereby prevent corrupting the dataset. Robotic operating system(ROS) [11] and its functionalities fulfil this purpose. It seamlessly transfers simulator's published data by subscribing to the publishing topic and allows the subscribing node to synchronise as necessary for storage.

So, now the data that resembles real-world is stored locally for later analysis and research.

1.1 Motivation

Though autonomous driving is one of the favourite research areas in mobility, the cost associated with integrating all the necessary sensors is still considered to be a major obstacle. Then comes the part where the environment has to be preprocessed in the vehicle to give the user an optimal prediction. The motivation for this thesis is to use a simulator, thereby avoiding a huge cost, do the necessary experiments and decide if using a simulator does help in achieving the greater goal of driving in the real-world.

As briefly mentioned, the high cost of associated sensors such as LIDAR [12], has put off many smaller research groups from implementing into their work. By using a simulator, again at a low cost,

to see if adequate experiments can be conducted on how different constellation of sensors work, how different modalities interact with each other and what impact does it have on the overall performance of the DNN.

Finally, implementing an end-to-end system which simulates real-world behaviour. This can then be applied to future research to make it more robust.

1.2 Goal

The desired goals of this thesis are listed below -

- Building an autonomous driving framework -
  - ROS - Use ROS2 to synchronise the data received from the simulator through a rosbrige, use functionalities such as slop and cache to order the data according to their received time in order not to scramble the information. During the evaluation, use the same functionalities to send command controls to the simulator.
  - Rosbridge - use a bridge transport protocol that connects the ros on the receiving end to the simulator on the sending end or vice versa.
  - Docker - setup a work environment that is independent of hardware or operating system which allows easy running of the commands for data collection and evaluation.
- Implement an end-to-end neural network architecture which learns to drive by training from image pixels to steering(control) commands. Besides, apply state of the art DL techniques.
- Implement a system that can efficiently collect and label data.
- Implement and analyse different constellation of sensors with different data fusion techiniques.

1.3 Related Work

In 2012, Alexnet [4], used CNNs to do object classification which then in Computer Vision, became the dominated approach for classification. Both Chen *et al.* [13] and Bojarski *et al.* [14] extended this approach of using CNN and demonstrated that in addition to classification, CNN can extract features from images. They then went onto demonstrate that through an end-to-end network, which self-optimised itself based on its inputs, predicted steering angles that keep the car in the lane of a road.

In a different field, but using CNN, Sergey Levine *et al.* [15] in 2016 corroborated that it was indeed possible to extract features with CNN and predict motor control actions in *object picking robots*.

Then, Xu *et al.* [16] in the same year with CNN-LSTM architecture showed that using the previous ego-motion states helped predict future ego-motion states. Using CNNs in an end-to-end architecture raised some questions on how it reached its decisions. So in 2017 both [17], [18] did visual analysis after the CNN layers to better understand the module's functionality. However, steering control alone can be used for vehicle control. It was then necessary besides steering, acceleration and deacceleration to be predicted for smoother control. Both acceleration and deacceleration are dependent on the user's driving style, lane's speed limit and traffic etc. Yand *et al.* [19] used CNN-LSTM architecture and provided the LSTM with feedback speed to determine the velocity of the EGO vehicle.

Though the control commands gave better vehicle control, it was necessary to perceive the environment for collision avoidance. The RGB colour camera sensors don't provide depth information which is vital for collision avoidance. Hence, it was necessary to use other sensors with different modalities

and fuse that information with RGB to predict an optimal output. Liu *et al.* [20] provided important rules in fusing data. They said that it was essential to pick out only key information and discard others to avoid additional noise, creeping into the dataset. They also described the techniques involved in data fusion – early/late fusion, hybrid fusion, model ensemble and joint training. Park *et al.* [21] gave us methods to enhance the features by using feature amplification or multiplicative fusion. Zhou *et al.* [22] detailed how fusing data into CNN affects the overall performance.

However, when the fused dataset gave a performance boost, it performed worse compared to individual modality. The combined fused model overfit more than its counterparts. This was observed to be the fundamental drawback of gradient descent in backpropagation. This paper [23] introduced a technique called *gradient blending* to counteract this problem.

Xiao *et al.*[24] used all the fusion techniques mentioned above and an imitation based end-to-end network[25]. It was found that RGB images with depth information(provided through a different modality) could indeed result in better performing end-to-end network model.

1.4 Contribution

References

[1] "Waymo." https://waymo.com/

[2] "Berkeley-deepdrive." https://bdd-data.berkeley.edu/

[3] "Apollo." http://apolloscape.auto/

[4] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems 25*, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2012, pp. 1097–1105. http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf

[5] "Radar introduction." https://www.explainthatstuff.com/radar.html

[6] "Lidar introduction." https://www.geospatialworld.net/blogs/what-is-lidar-technology-and-how-does-it-work/

[7] "Lgsvl simulator." https://www.lgsvlsimulator.com/

[8] "Nvidia simulator." https://www.nvidia.com/en-us/self-driving-cars/drive-constellation/

[9] "Carla simulator." https://carla.org/

[10] "Carmaker simulator." https://ipg-automotive.com/products-services/simulation-software/carmaker/

[11] "Robotic operating system." https://index.ros.org/doc/ros2/

[12] "Verge report on lidar." https://www.theverge.com/2020/1/7/21055011/lidar-sensor-self-driving-mainstream-mass-market-velodyne-ces-2020

[13] Z. Chen and X. Huang, "End-to-end learning for lane keeping of self-driving cars," 06 2017, pp. 1856–1860.

[14] M. Bojarski, D. D. Testa, D. Dworakowski, B. Firner, B. Flepp, P. Goyal, L. D. Jackel, M. Monfort, U. Muller, J. Zhang, X. Zhang, J. Zhao, and K. Zieba, "End to end learning for self-driving cars," 2016.

[15] S. Levine, P. Pastor, A. Krizhevsky, and D. Quillen, "Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection," *The International Journal of Robotics Research*, 03 2016.

[16] H. Xu, Y. Gao, F. Yu, and T. Darrell, "End-to-end learning of driving models from large-scale video datasets," *CoRR*, vol. abs/1612.01079, 2016. http://arxiv.org/abs/1612.01079

[17] J. Kim and J. F. Canny, "Interpretable learning for self-driving cars by visualizing causal attention," *CoRR*, vol. abs/1703.10631, 2017. http://arxiv.org/abs/1703.10631

[18] M. Bojarski, P. Yeres, A. Choromanska, K. Choromanski, B. Firner, L. D. Jackel, and U. Muller, "Explaining how a deep neural network trained with end-to-end learning steers a car," *CoRR*, vol. abs/1704.07911, 2017. http://arxiv.org/abs/1704.07911

[19] Z. Yang, Y. Zhang, J. Yu, J. Cai, and J. Luo, "End-to-end multi-modal multi-task vehicle control for self-driving cars with visual perception," 01 2018.

[20] K. Liu, Y. Li, N. Xu, and P. Natarajan, "Learn to combine modalities in multimodal deep learning," 2018.

[21] E. Park, X. Han, T. L. Berg, and A. C. Berg, "Combining multiple sources of knowledge in deep cnns for action recognition." in *WACV*. IEEE Computer Society, 2016, pp. 1–8. http://dblp.uni-trier.de/db/conf/wacv/wacv2016.html#ParkHBB16

[22] Y. Zhou and K. Hauser, "Incorporating side-channel information into convolutional neural networks for robotic tasks," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, 2017, pp. 2177–2183.

[23] W. Wang, D. Tran, and M. Feiszli, "What makes training multi-modal classification networks hard?" in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 12 695–12 705.

[24] Y. Xiao, F. Codevilla, A. Gurram, O. Urfalioglu, and A. M. López, "Multimodal end-to-end autonomous driving," *CoRR*, vol. abs/1906.03199, 2019. http://arxiv.org/abs/1906.03199

[25] F. Codevilla, M. Mller, A. Lopez, V. Koltun, and A. Dosovitskiy, "End-to-end driving via conditional imitation learning," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2018, pp. 4693–4700.