Image Enhancement Algorithms to Improve Robustness and Accuracy of Pre-trained Neural Networks for Autonomous Driving

A THESIS

SUBMITTED TO THE FACULTY OF THE GRADUATE SCHOOL

OF THE UNIVERSITY OF MINNESOTA

BY

Himanshu Yogesh Joshi

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
MASTER OF SCIENCE

Dr. Sayan Biswas

January 2023

Acknowledgments

I value the encouragement and support of many who made this arduous journey possible. My profound appreciation is extended to Dr. Sayan Biswas, my adviser, for his patience, guidance, and endless hours of meetings and discussions that helped me accomplish this work. I am grateful for Dr. Biswas's helpful feedback and constant inspiration, which have assisted me in developing as a researcher and an engineer. I would also like to thank Dr. William Northrop and Dr. Rajesh Rajamani from my master's committee for their time and valuable feedback. I am forever grateful to Dr. Lu Zhan, my mentor at the 3P Lab, who helped me formulate the problem statement and brainstormed various ideas to initiate the research in the right direction. I am thankful for Dr. Zhan's continuous support and encouragement to explore multiple methods to solve the problem.

I am grateful to the 3P Lab for allowing me to work there and providing the resources I needed to complete my research with the highest quality of work. Additionally, I thank Phil Magney and Matthew Linder for mentoring me during and after my time at VSI Labs. I am thankful to VSI labs for giving me permission in writing to use the sample dataset from their repository for this research.

I want to thank my fellow lab mates, Akash, Binit, Gabe, Daipayan, Oluwasegun, Roudh, and Prakadeeswaran, for their assistance, constructive criticism during the lab meetings, and encouragement all year round. I want to thank my roommates Chaitanya, Saahil, and Siddhant for all their help, being there with me through the ups and downs of my time here, and always keeping me motivated. Additionally, I would like to thank Koustav and Abha for their valuable suggestions and support. I also want to thank my cousin Niranjan for his helpful and constructive discussions regarding this work.

I want to extend a special thanks to Srujan for all his time, support, and insightful feedback that helped this research accomplish its objectives. It was fun working with you, and I shall always cherish the unaccountable hours we spent together learning, doing projects, and trying to solve exciting challenges.

Dedication

This work is dedicated to my parents, my better half, Divya, and my family and friends back home and here in the States for their unconditional love and support.

Abstract

The need for autonomy is imperative in many critical technology areas. Vehicle autonomy can revolutionize our existing transportation system. Studies [1,2] show that by 2035 there will be nearly 21 million autonomous vehicles operating on the road. Autonomous vehicles must be equipped with reliable, durable, and safe driver-aid technology prior to being put into use on public roads. To ensure road safety, autonomous vehicles must concurrently perform a number of critical tasks in real-time, such as object detection, lane following, autonomous emergency braking, and so on [3]. Past studies [4-6] have demonstrated that pre-trained neural network (NN) algorithms are well capable of handling these tasks. Even though the advancements in NN pre-training using machine learning and artificial intelligence are praiseworthy, there are occasions when these networks malfunction or even fail, leading to minor to fatal safety incidents. Examples of NN failure algorithms are non-detection of an object, such as a pedestrian or tree, false detection of an object, and misdetection of lanes or other vehicles on the road. These failures can primarily be attributed to the quality and quantity of pre-training data, and pre-training methodology.

Most to nearly all the NN are trained using data comprising of excellent lighting conditions and clear weather, but in reality, the vehicle could encounter poor lighting or harsh weather conditions. Hence, this training data makes the machine learning results biased toward good-quality data [7,8]. In this thesis, we propose a generalized data-driven approach for improving the pre-training image datasets fed to NNs. The algorithms developed and tested in this work could substantially enhance the image bringing out the critical spatial information in the image for better NN performance. This image enhancement technique consists of two main components: image colorization and contrast enhancement. Image colorization is implemented to obtain a color corrected image from a grayscale image. The traditional global contrast enhancement algorithm is extended to dynamic local contrast improvement to boost local contrasts within an image frame that suffers from under/over-exposed lighting conditions. The dynamic contrast improvement algorithm is further optimized for the computation cost. The algorithm is lightweight and computationally inexpensive that can be implemented in real time to improve images locally at the regions of interest. To increase robustness, a variable contrast improvement factor is added to improvise the local dynamic contrast algorithm to enhance the local region of interest in the image with appropriate improvements based on pixel intensities. Since dynamic contrast improvement algorithm relies on image segmentation, it produces an

unsmooth, discontinuous global image. To resolve this issue, the image segments are smoothened with a gaussian distribution to obtain a continuous image.

In summary, a Smoothened Variable Local Dynamic Contrast Improvement (SVL-DCI) algorithm is developed and thoroughly tested in the present thesis that could run in real-time as a pre-training algorithm for neural networks. SVL-DCI could substantially improve NNs prediction capabilities. We implemented SVL-DCI on the 3P lab dataset of 2470 images, and observed competitive improvement in the performance of the investigated NNs for object recognition (MaskRCNN) [9] and lane detection (LaneNet) [10].

Table of Contents

List of Figures

List of Tables

List of Abbreviations

Abbreviation	Definition
α	Linear Contrast Improvement Factor
β	Brightness Factor
I[x,y]	Intensity Value corresponding to pixel (x, y) of an image
3P Lab	Plasma Power Propulsion Laboratory
ADAS	Advanced Driver Assistance System
CI	Contrast Improvement
COCO	Common Objects in Context
DCI	Dynamic Contrast Improvement
DL	Deep Learning
FN	False Negative
FP	False Positive
FPS	Frames Per Second
GAN	Generative Adversarial Network
L-DCI	Local-Dynamic Contrast Improvement
LiDAR	Light Detection and Ranging
MaskRCNN	Mask Region based Convolution Neural Network
NN	Neural Network
P	Precision
PPV	Positive Predictive Value
R	Recall
RGB	Red, Green, Blue
ROI	Region of Interest
SVL-DCI	Smoothened Variable Local-Dynamic Contrast Improvement
TN	True Negative
TP	True Positive
TPR	True Positive Rate
VL-DCI	Variable Local-Dynamic Contrast Improvement

Chapter 1

1. Introduction and Overview

A long-term objective of machine learning is to develop machine intelligence similar to human intellect. A machine with machine intelligence undergoes pre-training via different learning techniques to infer judgments. These judgments include many feedback-based or feedback-free decisions that ultimately contribute to the capability of the machine's performance in a specific complex environment. Machine learning consists of algorithms and sub-processes that can be harnessed individually or in unison to achieve goals that otherwise might require a lot of computational complexities [11]. Machine intelligence also provides a utility benchmark to multiple algorithms, can test their working functionalities in various environments, and draw conclusions for further learning. Machine intelligence can tremendously benefit various industrial sectors, including automotive, medical, consumer electronics, industrial automation, education, and many other critical technology areas [12-14]. Machine intelligence applications include but are not limited to areas that need understanding and decision-making to a greater extent. Examples of such applications include viewer content suggestions in on-demand content platforms, intelligent portfolio management systems for investors, analysis and predictions in sports, autonomous and intelligent transportation, etc.

In the last decade, technologies that rely on machine intelligence have rapidly penetrated the automotive sector, and the idea of self-driving autonomous cars is now becoming a reality. The number of advancements in this sector is unparalleled, and the number of algorithms and sub-processes developed using the amalgamation of software and hardware is prevalent to an extent where it undeniably can be called the future of transportation. The novel technologies under development just for specific applications in transportation take up one of the most significant chunks in the upcoming technological breakthroughs. Although the mass deployment of autonomous vehicles still has a substantial path ahead, significant research activities are bringing up new and frequent evolutions in vehicle autonomy. The growth in the automotive industry due to the introduction of autonomy has created exponential market value growth and has given impetus to applications like never before. Technologies that use LiDAR (Light Detection and Ranging), cameras, and radars for automatic lane detection, driver assist, and parking assist are now made even more accessible. Their sophistication is several folds better than before. The extensive use of machine intelligence, neural networks, deep networks, and

other algorithms has made it feasible to make intelligent collaborative transportation a reality, thus making autonomous vehicles 'see' and 'perceive' things with higher accuracy than human drivers. Autonomous vehicle systems execute thousands of processes simultaneously to make critical decisions with minimal errors. Various neural networks (NNs) and deep learning networks (DL) addressing image segmentation [15], multi-sensor fusion, depth estimation [16], object recognition, and other operations make autonomous driving systems capable of adapting to complex real-world scenarios.

Even though state-of-the-art object recognition and lane detection NNs outperform humans in complex driving situations [17,18], there are not flawless. The performance of NNs changes drastically with driving conditions and surrounding information. Even the most accurate neural network suffers from inconsistency and mis/non-detection with a varied degree of detection confidence. To achieve any progress toward vehicle autonomy, we must develop robust NNs capable of delivering accurate results under a variety of driving conditions. Nearly all current research is concentrated on modifying the NNs parameters, such as weights and biases, as well as its structure to improve the robustness of NNs. However, this approach quickly turns into an application-specific optimization process and takes significant effort, rarely reaching any unified solution. Even if the NNs are over-optimized to specific driving conditions or scenarios, it could result in false or non-detections for challenging scenarios. This inconsistency in NNs compelled us to think about other factors causing this issue. We found that the problem partly results from the disparity between the quality and quantity of training data. Although real-life driving conditions include low illumination (heavy cloud covering, night-time driving, etc.) or bad weather (rain, snow, etc.), the training data is heavily biased toward clear skies and ideal lighting conditions. Machine learning results are heavily biased toward high-quality training data. **Figure 1.1** illustrates the analogy of the training dataset to a normal distribution. The model performs worse when the data contains only low-quality images and performs better when the data contains only ideal-quality images.

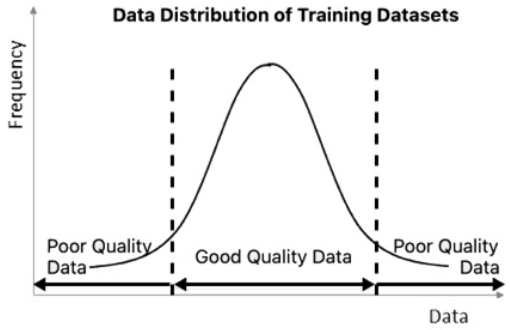


Figure 1.1: Imbalance of quality of data in training datasets

In the current work, a data-driven approach is adapted to improve the quality of the NNs training datasets, obtaining a robust result. Our recommended techniques could enhance the images and accentuate the spatial features between the frames for a NN to function more effectively. Our data-driven approach could improve the robustness of a wide range of NNs. The current research mainly investigates NNs for object recognition (MaskRCNN) [9] and lane detection (LaneNet) [10] as our baseline implementations. Currently, study utilizes two different datasets.

- Dataset 1: A dataset of around 2470 images was collected during the snowfall of 2021 on the streets of Saint Louis Park, MN. This source dataset is used for the object recognition task.
- Dataset 2: We are also utilizing VSI lab's dataset of 500 images as a source dataset for the lane detection task. Our work starts with implementing the mentioned two NNs, which serves as a baseline implementation.

A series of methodically constructed step-by-step image enhancement techniques were used to optimize the input/source dataset for our NN algorithm. First, the image enhancement initializes by checking the color correction of the image. We implement image colorization using Generative Adversarial Networks (GANs) [19]. We implement a traditional linear contrast improvement algorithm to generate enhanced image datasets. We extend the baseline image enhancement algorithm to dynamically decide if the image needs improvement based on the histogram representation (pixel intensities versus the number of pixels) of the image. This algorithm is referred to as Dynamic Contrast Improvement (DCI). After implementing DCI, we observed that the histogram representation of the image might not be the correct representation of the histograms of the local region of interest. The algorithm is improved by splitting the image into pre-determined regions and applying the DCI on all the regions separately. The DCI algorithm is enhanced to Local-Dynamic Contrast Improvement (L-DCI) algorithm. The scale of contrast enhancement is an input parameter consistent for all the regions of interest. A variability is introduced to the L-DCI by scaling the contrast improvement as needed by the corresponding regions. The Variable Local-Dynamic Contrast Improvement (VL-DCI) gives an image with distinguishable boundaries of the regions of interest. A Gaussian kernel is applied to the improvement scale to get a smoothened region of interest across the image.

Finally, a Smoothened Variable Local Dynamic Contrast Improvement (SVL-DCI) algorithm is used to obtain the improved dataset that can be used to get the expected detections from a pre-trained neural network. Results of this research highlight the quantified improvements in the performance of the given neural networks. The SVL-DCI could support the execution of the image enhancement in real-time.

Chapter 2

2. Background

2.1 Future of Mobility

Modes of mobility are rapidly changing and evolving. For ease in vehicle operability, the focus of vehicle development has shifted from the manual gearbox to automatic transmission. The manual transmission was a bottleneck in developing a driver-friendly environment, as the driver had to actively participate in engaging the clutch mechanism to change the vehicle speed. The automatic transmission eliminates the need for drivers' participation in speed change, making it automated and controllable. All the wheel speeds, the steering, the lighting system, and other complex mechanisms operate synchronously and can be controlled centrally. This control over the entire system has opened the doors for introducing various driver assistance functions in vehicles equipped with basic sensors such as cameras and short and long-range radars. These driver assistance systems, often known as Advance Drive Assistance Systems (ADAS) [20], handle critical tasks, including lane assistance, lane maintenance, obstacle avoidance, autonomous emergency braking, and more.

Over the last two decades, the scarcity of fossil fuels and the negative impact of a changing climate has encouraged the research and development of green and sustainable mobility [21]. The world is moving from carbon emissions towards green emissions. The automotive industry has entered a new sector of electric vehicles. Though much work is needed to improve the range of battery electric and hybrid vehicles, they are advantageous in handling, maintenance, and in the long term, reducing carbon footprint in the environment. The architecture of a battery electric or hybrid vehicle, given that every single process is controlled centrally, is suitable for implementing the ADAS technology to a greater extent. Ease in the large-scale deployment of ADAS in electric and hybrid cars has provoked researchers to further simplify driving by aiming for complete autonomy of the system. Autonomous driving involves taking signals from various sensors, processing them simultaneously, making crucial decisions in real time, and acting accordingly. Complete autonomy in vehicles is a difficult task to achieve as the vehicle constantly needs to interact with the dynamic environment, other vehicles in the scene, and pedestrians and maintain road safety by following the traffic lights and traffic signs.

Researchers are putting in efforts to develop flawless autonomous consumer vehicles and public transport vehicles. This mode of mobility would reduce traffic conjunctions on roads, improve the efficiency of the transport system, and reduce road accidents. An autonomous vehicle is a step closer to reducing human errors in transportation. The focus is on reducing machine errors by optimizing the performance of autonomous systems. As per SAE J3016 levels of driving automation, "there are total six levels of automation ranging from no driving automation (Level 0) to full driving automation (Level 5). Level 0: no automation; Level 1: automation of one control function, such as lane keep or autonomous speed control; Level 2: automation of two control functions; Level 3: confined self-driving but expect the driver to take control at whichever time with an adequate warning from the system; Level 4: drivers are not expected to take control at any point of the execution; and Level 5: fully autonomous driving with no human control [22-24]." **Figure 2.1** [25] shows the differences in all the levels of driving automation.

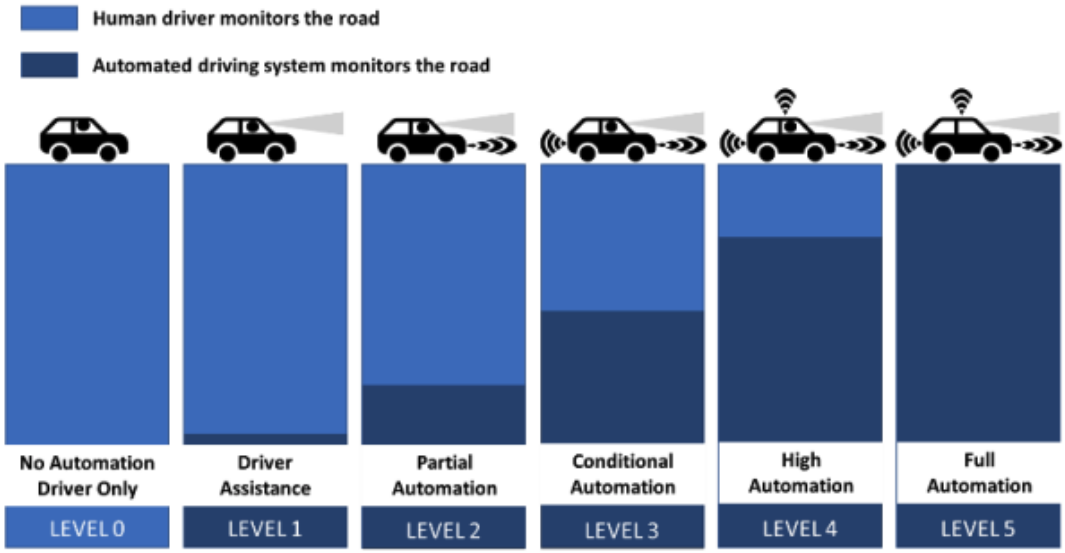


Figure 2.1**: The levels of driving automation** [25]

Tech giants such as Tesla, Waymo, Zoox, and Cruise are working on autonomous taxi services and vehicles for commercial use. They have achieved excellent results regarding the large-scale deployment of autonomous vehicles in real-world scenarios. The vehicle system must completely perceive and understand the environment to work autonomously in a dynamic environment. Most systems use a combination of vision cameras, LiDAR, radars, and thermal cameras to improve perception [26]. Similar to how the eyes are essential for humans, a vision sensor is vital for the vehicle to comprehend its surroundings. Visual information achieved using the most common vision sensor - a camera, misses depth information which is critical for

decision-making. This is where LiDAR sensors can bridge the information gap. LiDAR generates a 3D point cloud of the surroundings containing depth information. In conjunction with LiDAR, radars and thermal cameras are used to maintain the safety sphere by detecting other vehicles, objects, or pedestrians in the vehicle's proximity. A considerable amount of data is generated with all these sensors, and computationally heavy algorithms are implemented to make these systems work flawlessly.

Even though the research for autonomous vehicles has reached a milestone of deployment of such systems in masses, various edge cases could cause a safety hazard that is still difficult to address. Researchers are exploring options for shared intelligence between the traffic on the road. Instead of relying on the processing capabilities of a single vehicle and trying to make it robust, multiple autonomous vehicles can work together to make the decisions for dynamic situations easier. This research leads to the connected and autonomous vehicles (CAV) domain [27]. If a vehicle has already processed the surrounding completely, it can transfer this learning to the other vehicles wirelessly connected. This reduced processing load would help the network of connected vehicles perform efficiently and robustly in the dynamic environment.

As the research community progresses towards autonomous driving and CAVs, the need for a supportive infrastructure becomes crucial. The infrastructure at present is supportive of human drivers. All the ADAS features, such as lane following, cruise control, and adaptive braking, are built on the existing infrastructure features such as types of lanes, the color of lanes markings, the position of traffic signals, and traffic signs. For systems to perform more efficiently, these features could be altered and improved so that a machine could perceive the surroundings more accurately and efficiently. Considering the lane marking example: If the car exits a tunnel, there is a sudden change of light in the scene from dark to extremely bright. During this change, the visual feed obtained from the camera mounted on the car does not capture any features from the scene, including the lane markers, as shown in **Figure 2.2 (a)**. There are few test efforts to improve the visibility of lane markers by adding a high contrast darker patches at the boundary of the lane marking, as shown in **Figure 2.2 (b)**. These high-contrast lane markings are expected to give better performance of lane detection in situations where the scene is overexposed with the light.

Figure 2.2: (a) Visual feed of vehicle exiting a tunnel, (b) A high contrast lane marking

The efforts of improving the infrastructure and the technology should be happening in a synchronized manner. Any developments in either of the domain complement the developments in the other domain considerably. A vehicle system can only reach complete autonomy if the perception sensing and control system is robust and precise enough to extract all necessary information from the operating surrounding [28]. The more details captured, the more the tasks could be processed quickly and accurately. Perception algorithms in autonomous vehicles are crucial as they would make the vehicle capable of exploring the various features in the dynamically changing environment. The algorithms utilize these features to make crucial decisions in various applications, such as object detection, lane keeping, and adaptive cruise control.

2.2 Perception in Autonomous Vehicles

An autonomous vehicle must operate in a dynamically changing environment and consider all the possible movements of the objects in the scene to decide the safe trajectory of motion. The vehicle simultaneously performs various tasks such as perception, mapping, localization, locomotion, and providing feedback to the system. In the entire processing cycle of a vehicle, the first step is to perceive the surroundings completely. Autonomous vehicles are equipped with various sensors, such as color cameras, LiDAR, radars, etc., to obtain specific types of information from individual sensors [29]. Different sensor fusion algorithms are utilized to obtain valuable and meaningful insight from all the available sensor data. In particular, this data is used by various perception algorithms to carry out tasks such as object detection, scene recognition, depth estimation, pedestrian detection, automatic emergency braking, etc.

The perception function mainly focuses on computer vision, as visual feed makes the collected data meaningful. Various computer vision algorithms were developed to extract features from an image, and these features are used for solving complex problems of object detection, face

recognition, etc. Features can be edges, corners, changes in intensity, and specific points in an image [30]. Traditional computer vision algorithms are used for feature extraction from an image. Binary thresholding, Sobel filtering, linear contrast improvement, normalization, etc., are examples of feature extraction techniques [31]. These techniques have been in use for a long time and are computationally efficient. However, there are limitations to the implementation of these algorithms. While these above-mentioned algorithms produce excellent results for simpler scenarios, they lack versatility and sophistication for a complex vehicle operating environment. The traditional computer vision algorithms are susceptible to thresholding parameters, and the results change drastically even with a slight change in these parameters. If the algorithm is tuned for a particular lighting condition in the scene, and the input scene conditions change, the algorithm will not give the desired output. For an autonomous vehicle, there are better solutions than having a sensitive system that needs tuning repeatedly. With the rise of machine learning and artificial intelligence, researchers started to explore new and generalized computer vision algorithms. Neural network-based machine learning algorithms hold significant promise to enable vehicle autonomy.

A neural network is a collection of algorithms that aims to identify underlying links in a set of data using a method that imitates how the human brain functions. Neural networks consist of an input layer, multiple hidden layers, and an output layer. The neural network's architecture is fundamental as it determines the accuracy and execution time of getting the output. The feature extraction techniques using traditional computer vision algorithms are used internally to convolve the image and generate meaningful information out of the image. The images are passed to deep neural networks, and these networks are referred to as deep convolution neural networks [32]. The neural networks are trained on massive amounts of data to detect and classify the features. The accuracy and robustness of neural networks depend on the structure of the network and the type of data used for training it. The quantification of measurable parameters for the neural network is discussed in Section 2.3.

2.3 Quantification Metrics for Neural Networks:

Various quantification metrics for neural networks are used in research, such as accuracy of detections, mean squared error, etc. Recall and precision of detections is a practical quantification matrix as it gives an idea about the number of correct predictions versus the number of false predictions.

Let us consider an example of a neural network that would classify an object as a *car* or *not a car*. The confusion matrix for this example is shown in **Figure 2.3**. A confusion matric or error matrix [33,34] is a special kind of contingency table comprised of two dimensions ("ground truth" and "prediction") with identical sets of "classes" (*car* and *not a car*) in both dimensions.

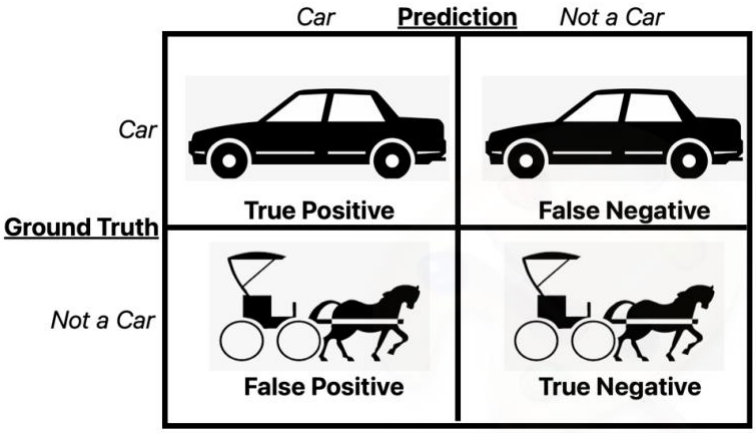


Figure 2.3**: Confusion matrix for a car classification model**

The recall of a neural network is defined as the number of true positives (TP) divided by the sum of true positives and false negatives (FN). The precision of a neural network is defined as the number of true positives divided by the sum of true positives and false positives (FP).

Recall = TP/ (TP+FN) **(2.1)**

Precision = TP/ (TP+FP) **(2.2)**

In the case of object detection models, the confusion matrix would be similar, as shown in **Table 2.1**.

TP	**True Positive**	**Actual object** in the ground truth, Model **did detect an object** in prediction
TN	**True Negative**	**No object** in the ground truth, Model **did *not* detect an object** in prediction
FP	**False Positive**	**No object** in the ground truth, Model **did detect an object** in prediction
FN	**False Negative**	**Actual object** in the ground truth, Model **did *not* detect an object** in prediction

Table 2.1**:Confusion matrix for an object detection model**

For an object detection model, having *higher recall* is more important than having *higher precision*. In the case of an autonomous vehicle, detecting FP might not be critical, but identifying any FN could be dangerous or even catastrophic, posing a collision danger since an object is present in reality, but the model does not predict it. Hence, the lower the FN, the higher the recall.

There are a few methods by which recall, and precision can be improved. First is improving the image given to the neural network, and second is optimizing the structure of the neural network. Image enhancement is targeted with a combination of implementing traditional computer vision algorithms and various image improvement neural networks. The neural network's structure is optimized by changing the number of layers and manipulating the weights and biases, leading to the activation of corresponding neurons in the output layer. In Section 2.4, we will discuss the literature review regarding improving the performance of the neural network by optimizing the structure and the image enhancement using image colorization GAN.

2.4 Literature Review

Traditional computer vision algorithms for image enhancement focus on image improvements that are often primitive/rudimentary and lack sophistication. These improvements change the fundamental properties of an image, such as contrast, brightness, and color saturation. Sometimes, these primitive improvements do not help achieve the expected neural network results. Color correction is an essential aspect of image enhancement. After rudimentary improvements, a color correction can be applied to an image to get the desired results. The NNs utilizing shapes and colors of the features to identify objects perform better with the enhanced images. NNs' structure and other controllable parameters can be optimized to improve performance.

Previous studies [35-38] have demonstrated that contrast improvement implemented judiciously can be a robust image enhancement technique. Contrast can be defined as a difference in the luminous intensity along the pixels in the image. The human visual system is more sensitive to contrast than absolute luminance. Better contrast images make the objects in them appear distinguishable from the background. Contrast Improvement includes altering the intensity of the pixel along with the brightness. Wongsritong et al., 1998 [35] implemented a contrast enhancement on an image using multipeak histogram equalization while preserving the image's brightness. The results show that the output image has the same brightness as the input image, but the contrast is enhanced. They validated their method on two images with extreme brightness

values and got promising results. Stark, 2000 [36] proposed an adaptive image contrast enhancement using generalizations of histogram equalization. This implementation gives various stages of contrast enhancements for a given input image, starting from an image with no change and finally with an adaptive equalization. Chen et al., 2003 [37] proposed and analyzed a minimum mean brightness error bi-histogram equalization technique in contrast enhancement. This study implemented the bi-histogram equalization on highly bright and dark images and got a normalized image as output by keeping the brightness of the image constant. Sun et al., 2005 [38] implemented a dynamic contrast enhancement based on histogram specification. This method is computationally efficient and robust; hence it could be utilized for real-time executions. All the discussed methods showed promising improvements in the contrast of the image and are computationally efficient as they are logically simple and computationally inexpensive. In this research, we utilize these implementations as our baseline. The primary issue with these methods is that they convert images from pixel to histogram space and lose the spatial information of the features in the image. We build a spatial relationship between the histograms in the proposed method by localizing these implementations throughout the image.

Besides contrast, color is another important aspect associated with an object and is utilized by the human eyes while processing information [39]. Computers also perceive the world similarly, and researchers are training machine learning models to achieve the processing complexity of a human brain. Images can be enhanced with a color correction. If the input image has an improper distribution of the RGB colors, then neural networks can help to color correct that image. Various GANs are used to colorize grayscale images. GANs are a class of machine learning frameworks developed by Goodfellow et al., 2014 [40] that utilize a generator and a discriminator. These two models are trained simultaneously with the backpropagation technique. GANs are used broadly to solve problems such as image colorization, classification, segmentation, etc. Creswell et al., 2018 [41] studied deep learning for visual understanding and presented various advantages and challenges of using GANs in computer vision. Zhang et al., 2016 [42] proposed a method to colorize an image by solving the problem as a classification problem and demonstrated the results on various image classes. Guo et al., 2020 [43] proposed a technique to colorize a grayscale image using a GAN. This method was validated on a dataset containing a top view of the land. In this research, we propose utilizing a pre-trained GAN for image colorization that is validated on real-world scenes containing various objects such as vehicles, bicycles, and traffic lights. The implementation proposed by Isola et al., 2016 [44] in Image-to-Image translation with conditional adversarial network showed promising results on our test datasets.

GANs have two networks, a generator, and a discriminator, that predict different tasks simultaneously. Let us consider an example of face generating GAN. **Figure 2.4** shows a block diagram of the structure of a GAN for this example. Unlike a neural network having one loss function, GANs have two separate loss functions, one for the generator and one for the discriminator.

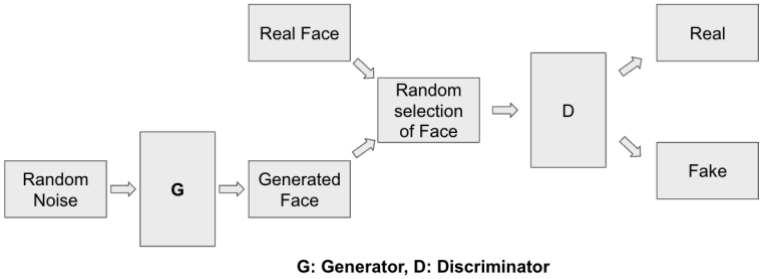


Figure 2.4: GAN structure

Initially, the generator starts generating random faces from noise, and the discriminator correctly classifies a true face versus a false face. At the start, the loss function of the generator creates more correction, and for the discriminator, the loss function deviates minutely. As the training progresses, the generator starts to create faces similar to real faces, and the discriminator starts to falsely predict the synthetically generated face as an actual face. At this moment, the loss function of the generator starts to give minute corrections, and that of the discriminator gives more corrections. This is the iteration where the training is completed, and the GAN is ready to implement on test data.

Most networks take the input image as a grayscale image for image colorization. Two representations are most commonly used to predict the color values for the output image, RGB and LAB representation. A given pixel has red, green, and blue intensities in RGB representation. In LAB representation, a given pixel has a lightness value, i.e., the grayscale intensity of the pixel, a channel value which is the green-red color intensity of that pixel, and b channel value which is the yellow-blue color intensity of that pixel. If the model uses RGB representation, it would need to predict three intensity values in the range [0,256], giving the possible combinations of 256x256x256, which is more than 16 million values. On the other hand, for the LAB representation, the number of predictions the model has to make is 256x256, which is about 65000, as the lightness values are already available from the input grayscale image. Most networks utilize the LAB representation to reduce the execution complexities and runtime. **Figure 2.5** shows the difference between an image's RGB representation and the LAB representation.

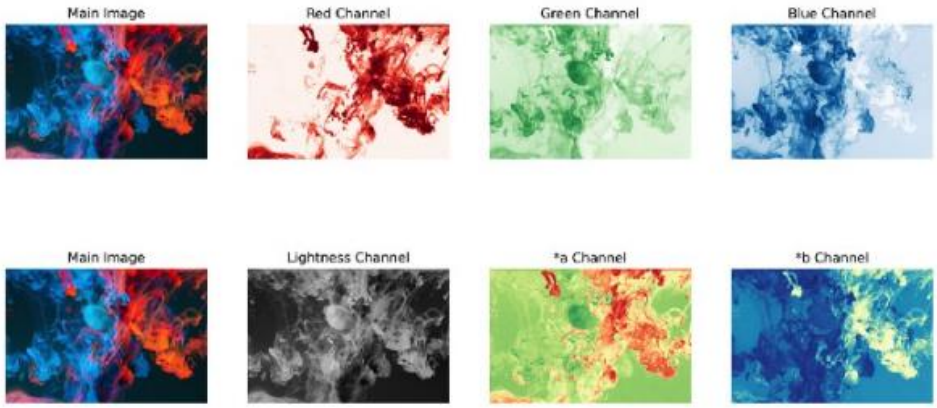


Figure 2.5: RGB representation (top), LAB representation (bottom) [45]

The contrast improvement and image colorization techniques will be utilized in this research as a baseline implementation for image enhancement. The other approach to improve the recall and precision of detections for neural networks is modifying the structure of the network, tweaking the weights and biases of the neurons, and the learning rate of the algorithm. Li et al., 2009 [46] proposed an improved backpropagation training algorithm for the neural network with a self-adaptive learning rate. This method helped to reduce the iteration time for the batch backpropagation resulting in better training performance and flexibility in modifying the network parameters. The process of initializing the weights and biases at the first iteration is often a random selection, and then they get adjusted and corrected as per the designed loss function. Pavelka and Prochazka, 2004 [47] proposed a comparison study between algorithms for initializing weights for the neural networks. This study presents a quantified outcome of various randomization techniques that can be used to initialize the weights for the network.

Modifying the network's structure and manipulating the weights and biases of the activations is an application-specific task that needs multiple trial-and-error attempts. If the network is trained and tuned for one application, it can only be used for some applications. In this research, we propose a generalized data-driven technique for image enhancement that is an application for various neural networks addressing different problems. We verify our method on networks such as MaskRCNN for object detection and LaneNet for lane detection.

2.5 Dataset

Our lab – Plasma Power Propulsion Laboratory (3P Lab), is located in Minneapolis, Minnesota, where winter/snow driving conditions could last 4-5 months of the year. The weather here in winter is very harsh, with heavy snowfalls. The average annual snowfall in Minnesota varies from

36 to 70 inches of snow [48]. Most R&D efforts related to autonomous vehicles are concentrated in places with no or light snowfall, such as Arizona and California with nearly 2 to 12 inches of snow [49]. Our research aims to make the algorithms of autonomous vehicles robust to extreme weather conditions. We aim to implement the image enhancements resulting in an improvement of the neural network performance on a dataset collected in snow conditions.

In conjunction with VSI Labs, an autonomous vehicles solution company in Saint Louis Park, Minnesota, we collected data during winter driving conditions. We utilized their Ford Fusion Vehicle equipped with an RGB camera in the front to collect the visual feed of real-world scenarios on the streets of Saint Louis Park. This dataset is used to implement the image enhancement algorithm and validate the improvements in the results of the object detection model. This dataset consists of 2470 images in '.jpeg' format of 800x600 pixels. **Figure 2.6** shows a sample image from this dataset.

I worked as a summer intern at VSI Labs this summer from May-August. We were able to get a sample dataset of 500 images from the VSI Labs data repository (with VSI Labs' written permission) to implement the image enhancement algorithm and validate the implementation results on the lane detection model. This lane detection dataset consists of RGB images in a '.jpeg' format of 1600x900 pixels. **Figure 2.6** shows a sample image from this dataset as well. Both the images shown in this figure are resized to the same dimension for representation purposes.

Figure 2.6: Sample images, object detection dataset (left), lane detection dataset (right)

These two datasets are used to implement the proposed method, and part of these are used as a testing dataset to validate the improvements in the neural network's performance. The object detection and lane detection dataset are used to implement the baseline image enhancement algorithm and the respective neural network models on these datasets to generate quantitative results to support the proposed method. Further developments in the contrast improvement algorithm followed by the neural network implementation are done on the object detection dataset as it contains more images to validate the results.

2.6 Need for a Data-Driven Approach

We implemented the MaskRCNN and LaneNet models on the datasets that are discussed in the earlier section. These neural networks are pre-trained on the COCO (Common Objects in Context) and LaneNet datasets. We utilized the same input parameters for our implementation, as we want to generalize the use of neural networks and make it independent of the quality of data they are implemented on. The neural network performs well on a good-quality image where the lighting conditions are bright, and the weather conditions are not harsh. However, in real-life scenarios, the lighting conditions are darker, and the weather can be harsh. **Figure 2.7** shows the images from the COCO dataset used for training the MaskRCNN and the LaneNet dataset used for training the lane detection model. **Figure 2.8** shows the images that the system encounters in the real-world.

Figure 2.7: Sample images from training dataset, COCO dataset (top), LaneNet dataset (bottom)

Figure 2.8: Sample images from real-world scenarios, 3P Lab dataset (top), VSI Labs dataset (bottom)

It is clear from the above figures that the neural networks are biased toward good-quality data. If the system encounters a poor-quality image, the networks perform inconsistently and lack robustness. We examined the implementation of MaskRCNN and the LaneNet models on the collected datasets and confirmed the inconsistent robustness in the detections of the neural networks. **Figure 2.9** and **Figure 2.10** shows the implementation results of MaskRCNN on the object detection dataset. **Figure 2.11** shows the implementation of LaneNet on the lane detection dataset.

Figure 2.9: False negatives for MaskRCNN shown in dashed rectangles

Figure 2.10: False positives for MaskRCNN shown in dashed rectangles

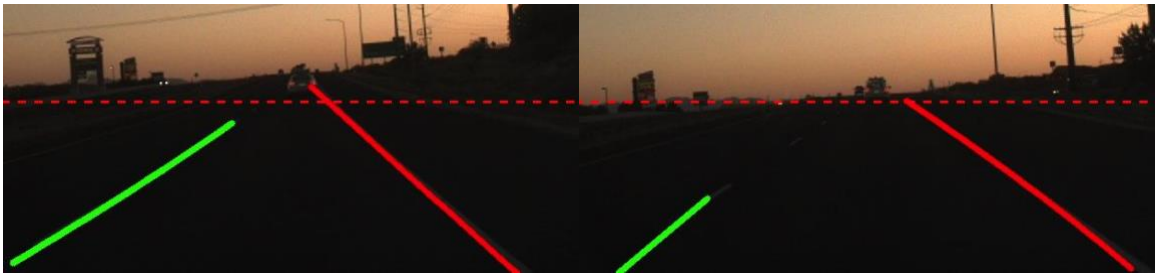


Figure 2.11: Inconsistent horizon for detected lanes marked with a horizontal dashed line

This shows the imbalance in the quality of images in the training dataset. It concludes the need for a data-driven approach to improve the robustness of the neural networks. Before being given to the network, the dataset needs to be improved to match the quality of the training dataset in order to achieve the expected results from the neural network. This research proposes a data-driven technique to improve the quality of the datasets and aims to achieve a real-time execution so that this method can serve as a pre-processing framework to the neural network.

Chapter 3

3. Methodology

3.1 Overview

The proposed image enhancement algorithm adds a data-driven framework to the traditional approach of generating the outputs from various neural networks. **Figure 3.1** shows the traditional approach of feeding the input images to the neural network and predicting the output.

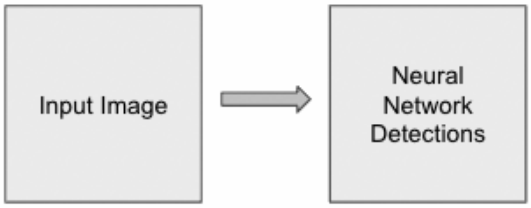


Figure 3.1: Traditional method to implement the neural network

We have studied the implementation of the traditional method and observed the inconsistency in the robustness of the results of the neural networks. This inconsistency is quantified in terms of false positives and false negatives for object detection and the horizon level for lane detection, as shown in **Figure 2.9**, **Figure 2.10,** and **Figure 2.11**. **Figure 3.2** shows the proposed method by adding an image enhancement framework between the input and output blocks.

Figure 3.2: A proposed method for improving the robustness of neural networks

The image enhancement framework consists of two steps. The First process deals with the color correction of the image. The second addresses the contrast improvement of the image. Neural networks focus on the features in the image and can be affected by color, size, shape, and various other factors of the features. **Figure 3.3** shows the two steps involved in implementing the image enhancement framework.

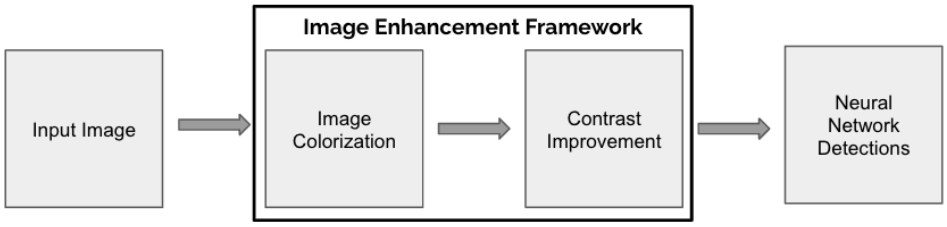


Figure 3.3: Two-step image enhancement framework

The image colorization block is crucial if the image does not have a proper Red-Green-Blue (RGB) distribution. The features that are supposed to get detected by the neural network might be adversely affected. The histogram representation of the image can gauge the correct distribution of the RGB channels. For any RGB image, the color histogram representing each color should be distinguishable. If any of the color histograms of channels overlap, then the algorithm concludes that the image is not colorized properly and needs color correction. Once the image is color corrected, it is passed to the second block, where the contrast improvement algorithm is implemented. **Figure 3.4** shows the complete implementation of the image enhancement technique.

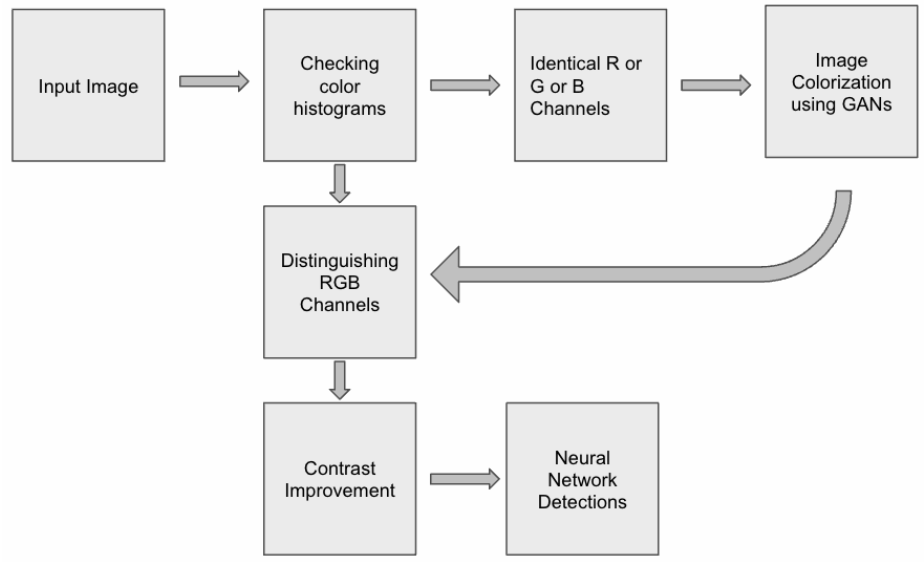


Figure 3.4: Details of the implementation of the image enhancement technique

The details about the setup, experiments, and other implementations for the two blocks in the image enhancement framework are explained in Section 3.2 and Section 3.3.

3.2 Image Colorization

The image dataset that the 3P lab collected is used to generate faulty RGB images synthetically. The blue and red channels are made to overlap with each other and generate the incorrect distribution of color histograms. **Figure 3.5** shows the faulty RGB image and color histograms of corresponding images. These images appear grayscale, but they are RGB images, as every channel has a histogram distribution of pixel intensities versus the number of pixels.

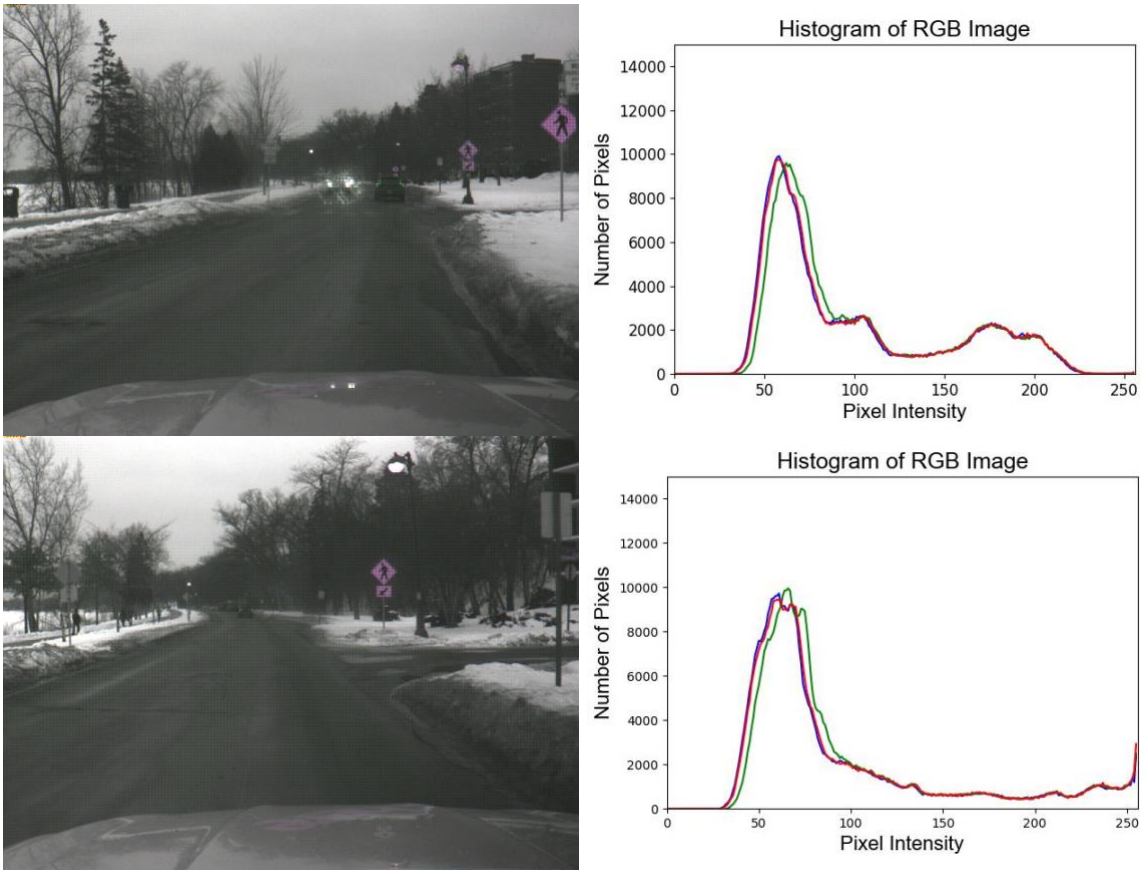


Figure 3.5: Faulty RGB image (left) and corresponding color histogram (right)

The color histogram of the image can be manually studied, and a decision can be made whether or not the image needs color correction. If the image needs color correction, image colorization using GAN could be implemented. **Figure 3.6** shows the output of the GAN implementation. The image is color corrected, and the result can be verified from the color histogram of the image. The blue and red channel distribution has become distinct.

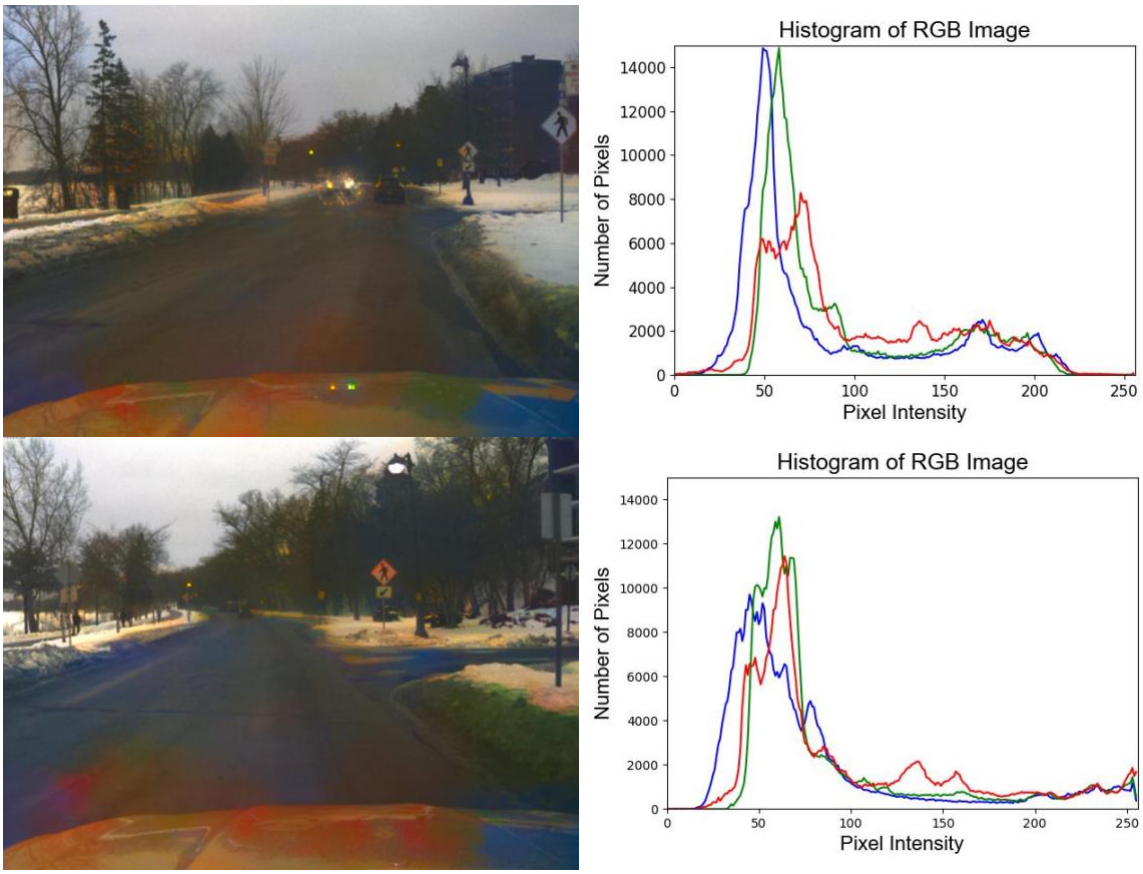


Figure 3.6: Image colorization using GAN

To verify the GAN implementation, we also studied the histogram of the original image used to generate the synthetically faulty RGB image. The histogram distributions are distinct but demonstrate a similar trend of pixel intensities. Note the first peak – peak closer to pixel 50-75 signifies the overrepresentation of darker pixels. **Figure 3.7** shows the source image and the corresponding color histogram of the images.

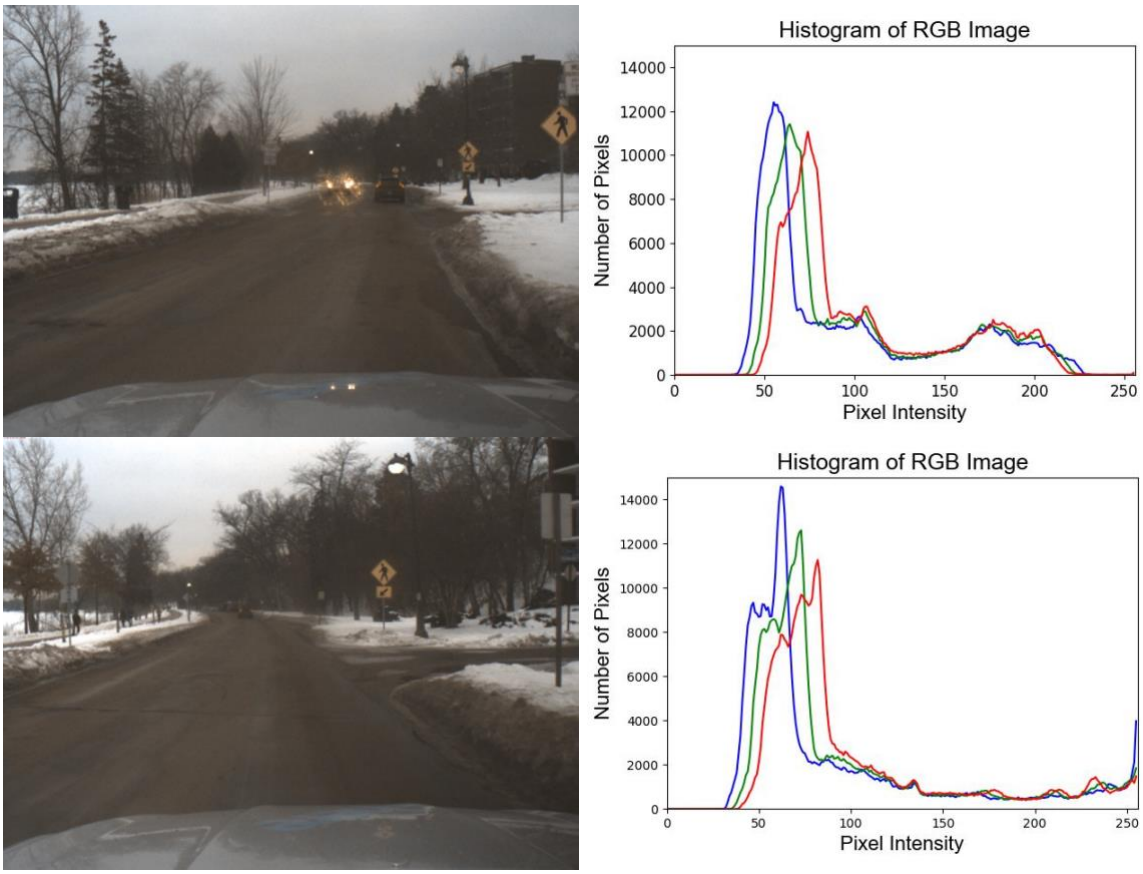


Figure 3.7: Source RGB image (left) and corresponding color histogram (right)

The color correction step makes the RGB histogram of the image smooth and regularly spaced. This improved color distribution helps to decide that the image is ready for further enhancement. After getting a color-corrected RGB image, it is passed onto the next step of the image processing framework, i.e., contrast improvement.

3.3 Contrast Improvement (CI)

Image enhancement with linear contrast improvement is the baseline implementation for this research. We found that, with contrast enhancement, the image's features are highlighted, which helps the neural networks for detection. We implemented the MaskRCNN and LaneNet on the images obtained from the contrast improvement and noticed an increase in recall and precision.

Consider an RGB image of size MxN pixels. Given the pixel at the location (x,y), the intensity of that pixel, I, is a vector of intensity values for R, G, and B channels given by **equation 3.1**:

$$I(x,y) = [I_R\ I_G\ I_B] \quad \textbf{(3.1)}$$

Linear Contrast Improvement function, f(I), changes the intensity of the pixel at the location (x,y), as given by **equation 3.2** [50]:

$$I'(x,y) = \alpha * I(x,y) + \beta \quad \textbf{(3.2)}$$

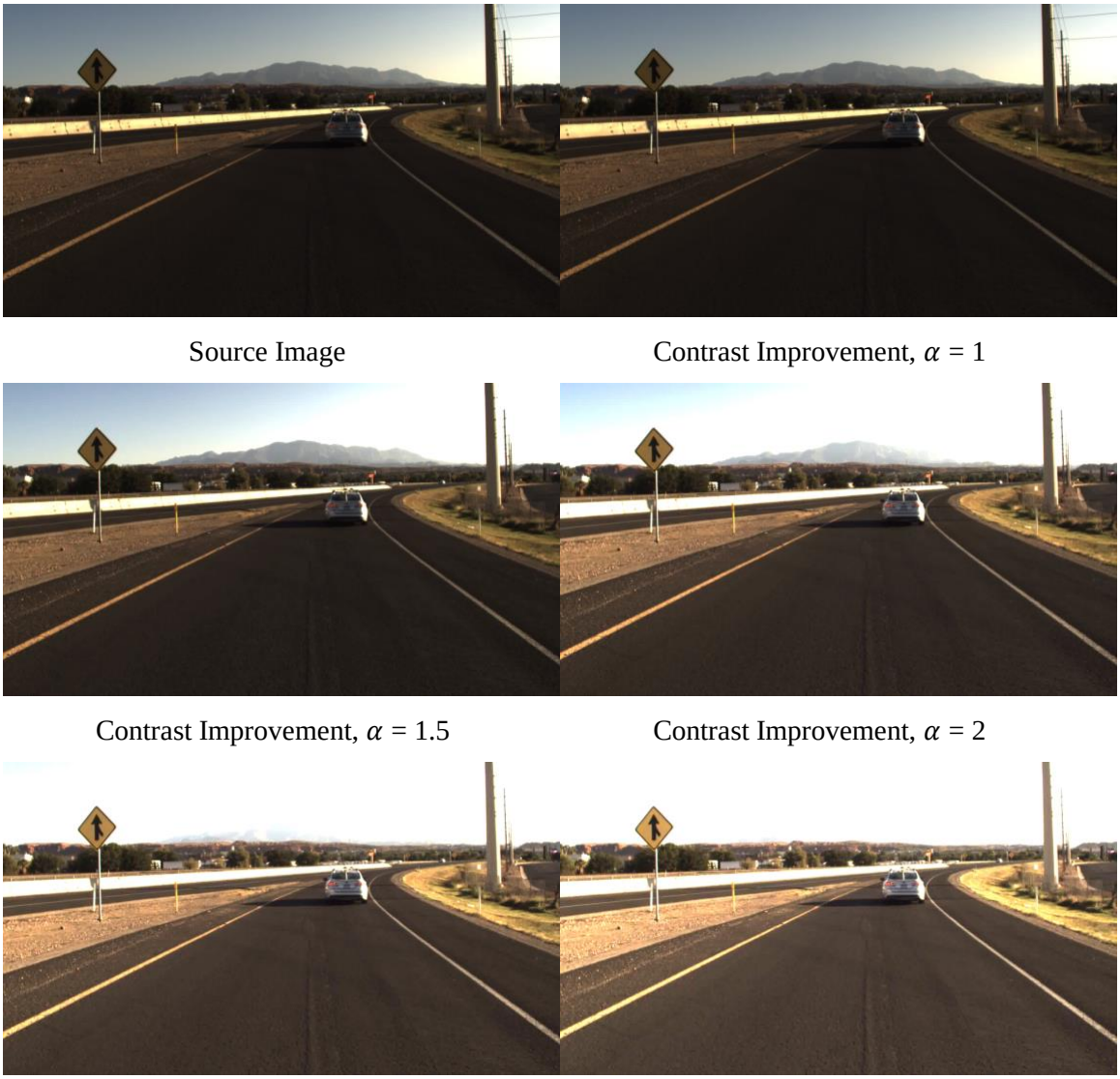
Where α is the factor to improve the contrast and β is the factor in improving the brightness. Contrast is about change in intensities. Hence, we will only consider α, as it is a scaling factor to intensity, and would keep β=0, as we are interested in improving the contrast of the image and not the brightness of all the pixels. The equation for linear contrast improvement becomes:

$$I'(x,y) = \alpha * I(x,y) \quad \textbf{(3.3)}$$

All the pixels in an image could take any intensity value I(x,y) out of possible 256 values as [0:255]. While we scale the intensity, it is important to clip the values at extreme ends. The linear contrast improvement algorithm for all the pixels in the image is as shown below:

*I'(x,y) = α*I(x,y)*
***IF** I'(x,y) < 0:*
    I'(x,y) = 0
***ELSE IF** I'(x,y) > 255:*
    I'(x,y) = 255

This algorithm would limit the I'(x,y) values in the range [0:255]. The value of α can be varied in the range of [0,3]. $\alpha < 1$ would reduce the contrast; hence, we would use $\alpha \in [1,3]$. **Figure 3.8** shows some qualitative results with various values of α tested on the lane detection dataset. The figure below shows that the higher the value, the better the feature visibility in the image is. The image with α 2.5 and 3 highlights the lane markers better than all the other α values. To verify the selection of α value, contrast improvement is implemented on the object detection dataset, as it contains a snow scene. **Figure 3.9** shows contrast enhancement implementation on an image from the object detection dataset.



Source Image | Contrast Improvement, α = 1

Contrast Improvement, α = 1.5 | Contrast Improvement, α = 2

Contrast Improvement, α = 2.5 | Contrast Improvement, α = 3

Figure 3.8: Source image and various contrast improved images with corresponding α values

Contrast Improvement, α = 2.5 Contrast Improvement, α = 3

Figure 3.9: Contrast Improvement on object detection dataset

It is clear from the figure above that, with α of 3, the image containing the lights or snow is washed out. To avoid the loss of features, α of 2.5 is finalized. To support this selection of α as 2.5, we randomly selected 1% of the images from the object detection dataset. We varied the α value and generated corresponding contrast-improved image datasets. We implemented the MaskRCNN on these datasets and obtained the quantified recall and precision results. The quantification supports the selection of α value as 2.5 for all the implementations further. **Table 3.1** shows the quantified results for various α values. **Table 3.2** shows the percentage improvement in recall and precision from the baseline implementation.

Alpha Values	Recall	Precision
1 (Baseline Implementation)	**0.52**	**0.86**
1.5	**0.58**	**0.86**
2	**0.59**	**0.88**
2.5	**0.61**	**0.88**
3	**0.58**	**0.88**

Table 3.1**: Quantification of recall and precision for various α values**

Alpha Values	Recall	Precision
1.5	**10.0%**	**0.4%**
2	**13.2%**	**2.4%**
2.5	**16.6%**	**2.7%**
3	**10.7%**	**2.4%**

Table 3.2**: Percentage improvement from the baseline implementation in recall and precision**

The above results are plotted on a graph, and the smooth curve of recall peaks at α of 2.5, as shown in **Figure 3.10**. This peak indicates that the neural network performs better at this value of α as the objects and the features in the scene are highlighted due to higher contrast. The performance decreases beyond this α value, as the image starts to wash out due to overexposure, and the details of the features are lost.

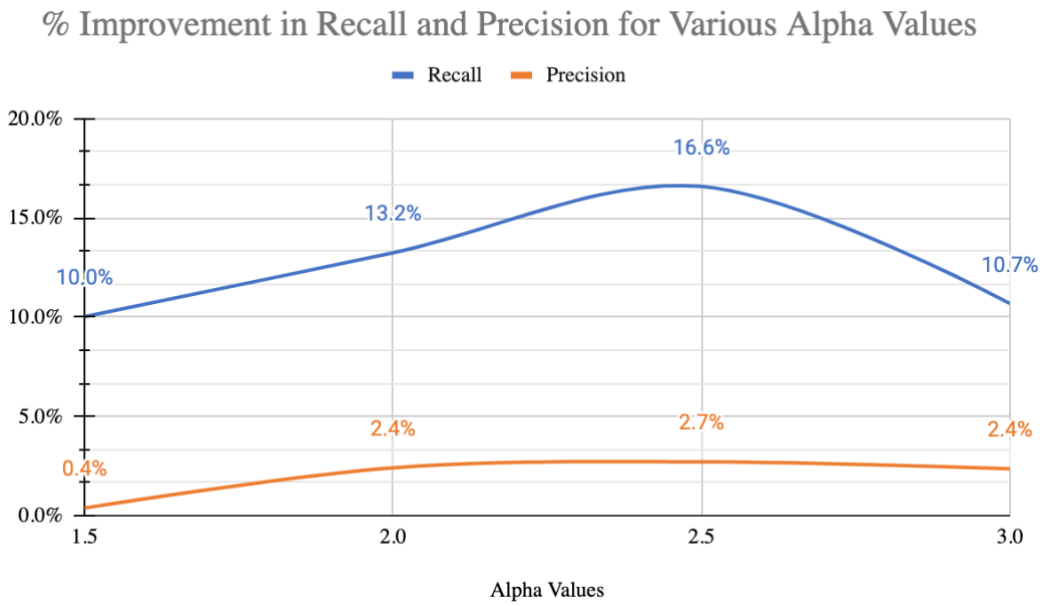


Figure 3.10**: Plot for percentage improvement from baseline implementation in recall and precision**

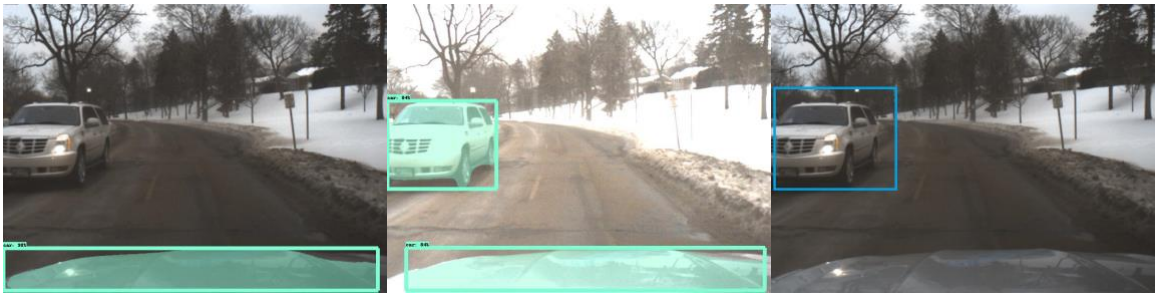
With the concluded value of α, the contrast-improved dataset is created and referred to as the baseline implementation hereafter. Both the neural networks are implemented on respective datasets, and improvements in the detections were observed. The object detection neural network results are quantified with the recall and precision of detections. **Figure 3.11** shows the improvement in the results obtained from the implementation of MaskRCNN. The lane detection neural network results are quantified with the detected horizon level. **Figure 3.12** shows the improved horizon level results obtained from the implementation of LaneNet.

TP = 2, FN = 6, FP = 0 | TP = 3, FN = 5, FP = 0 | Number of Detections = 8

R = 0.25, P = 1 | R = 0.38, P = 1

TP = 2, FN = 2, FP = 1 | TP = 2, FN = 2, FP = 0 | Number of Detections = 4

R = 0.50, P = 0.67 | R = 0.50, P = 1



TP = 0, FN = 0, FP = 0 | TP = 1, FN = 0, FP = 0 | Number of Detections = 1

R = 0, P = 0 | R = 1, P = 1

TP = 1, FN = 6, FP = 0 | TP = 4, FN = 3, FP = 0 | Number of Detections = 7

R = 0.14, P = 1 | R = 0.57, P = 1

Figure 3.11: Quantification of detections by MaskRCNN on source dataset (left), and contrast improved dataset (middle), and the ground truth detections(right)

Horizon for left lane = 4/5 Horizon for left lane = 5/5

Horizon for right lane = 5/5 Horizon for right lane = 5/5

Horizon for left lane = 5/5 Horizon for left lane = 5/5

Horizon for right lane = 0 Horizon for right lane = 5/5

Figure 3.12: Quantification of detections by LaneNet on source dataset (left), contrast improved dataset (right)

The implementation above shows promising results regarding improving the robustness of the neural networks. The contrast improvement algorithm can be improved further, as the image might already have good exposure to light. Applying a contrast improvement algorithm to that image might lead to a washed-out image. Also, the headlights of the cars can cause a glare in the image, and it would increase if the contrast improvement were applied to such an image. **Figure 3.13** shows a scene having snow and the contrast improvement resulting in a washed-out image. Similarly, **Figure 3.14** shows a vehicle with headlights turned on, and the contrast improved the image, losing the vehicle features due to increased glare around the headlights.

Figure 3.13: Issues with contrast improvement, source image (left), contrast improved image (right)

Figure 3.14: Issues with contrast improvement, source image (left), contrast improved image (right)

An algorithm that accounts for the current pixel intensities of the image before applying contrast enhancement will be a robust image enhancement technique. The algorithm would be able to decide if the contrast improvement is needed or not. Chapter 4 explains the evolution and developments in contrast improvement algorithms that are robust and can work on various datasets.

Chapter 4

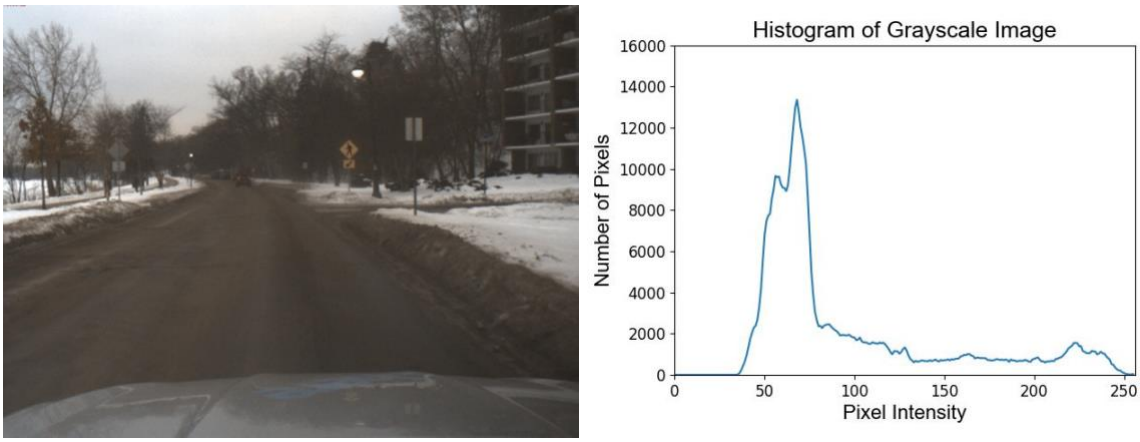
4. Algorithm Improvements

4.1 Dynamic Contrast Improvement (DCI)

Dynamic decision-making is added to the contrast improvement algorithm to address the issue of limiting the contrast enhancement in already overexposed frames. This algorithm is referred to as Dynamic Contrast Improvement Algorithm (DCI). The image representation is converted from pixel space to histogram space. The histogram is a plot of intensity values of the pixels ranging from 0 to 255 versus the number of pixels belonging to that particular pixel intensity. Histogram representation of an image gives an idea about the overall pixel intensity distribution in the image [51]. The peak of this histogram represents the overall intensity of the image. The higher the peak, the higher the intensity values of most pixels. We added a step of thresholding the peak of the histogram to dynamically decide if the image needs a contrast enhancement or not in the contrast improvement algorithm, which became the foundation of DCI. The DCI algorithm is as follows:

$intensity_{peak}$ = *intensity corresponding to the maximum number of pixels in the histogram*
$intensity_{threshold}$ = *peak threshold (input parameter)*
IF $intenstiy_{peak} < intensity_{threshold}$
    $I'(x,y) = \alpha * I(x,y)$
    IF $I'(x,y) < 0$:
        $I'(x,y) = 0$
    ELSE IF $I'(x,y) > 255$:
        $I'(x,y) = 255$
ELSE
    $I'(x,y) = I(x,y)$

After implementing the DCI, the peak of the histogram shifts towards the brighter pixel values, i.e., 255. **Figure 4.1** shows the images with the histogram of corresponding grayscale images. The shifting of the peak is visible after the DCI implementation.



Source Image

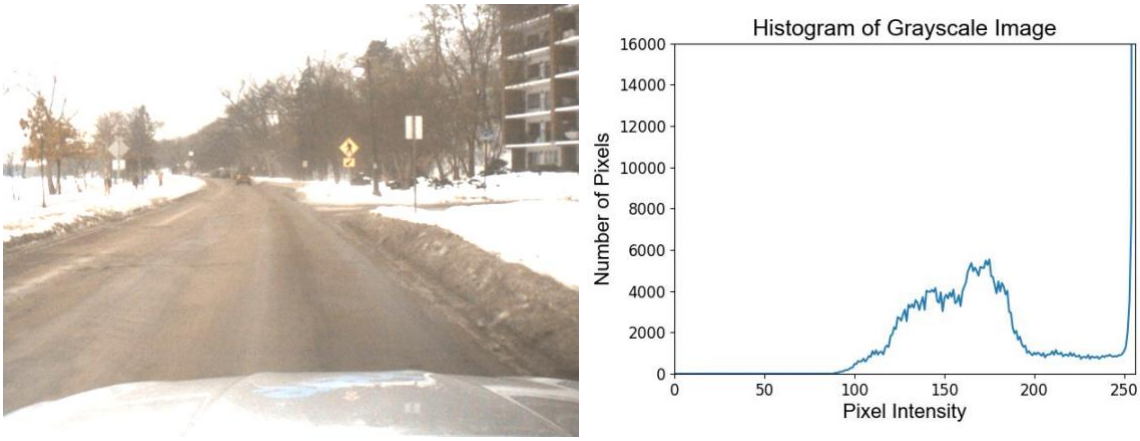
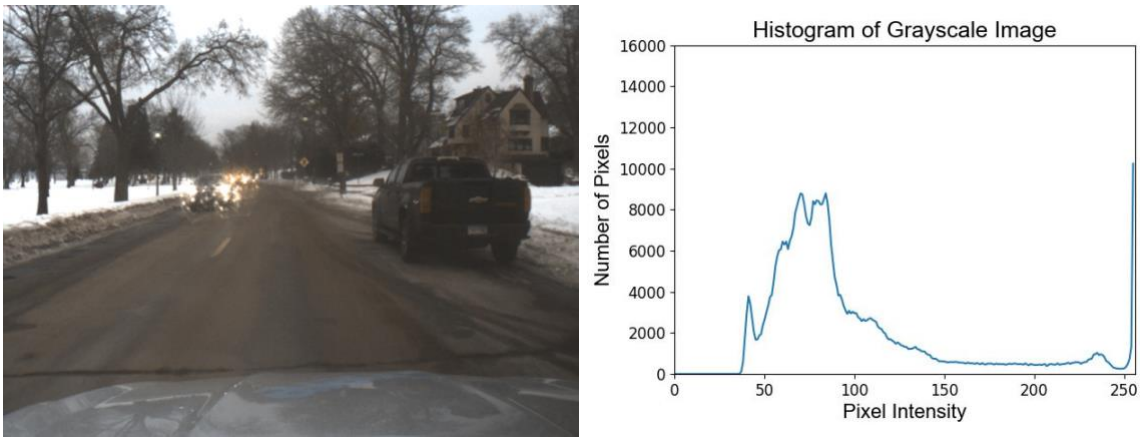


Image after DCI implementation

Figure 4.1: Shifting of intensity peak, RGB image (left), histogram of grayscale image (right)

If the intensity peak of the histogram of the image is already beyond the threshold parameter, then the DCI would interpret that the image is already bright and would not enhance the image further. DCI helps improve the scenes with poor lighting conditions but keeps the well-light images. This helps in improving the dataset quality generically. **Figure 4.2** shows the implementation of DCI when the intensity peak of the histogram is beyond the threshold. DCI decides not to enhance the image further and prevent the overexposure of the image that is caused due to the contrast enhancement algorithm.



Source Image

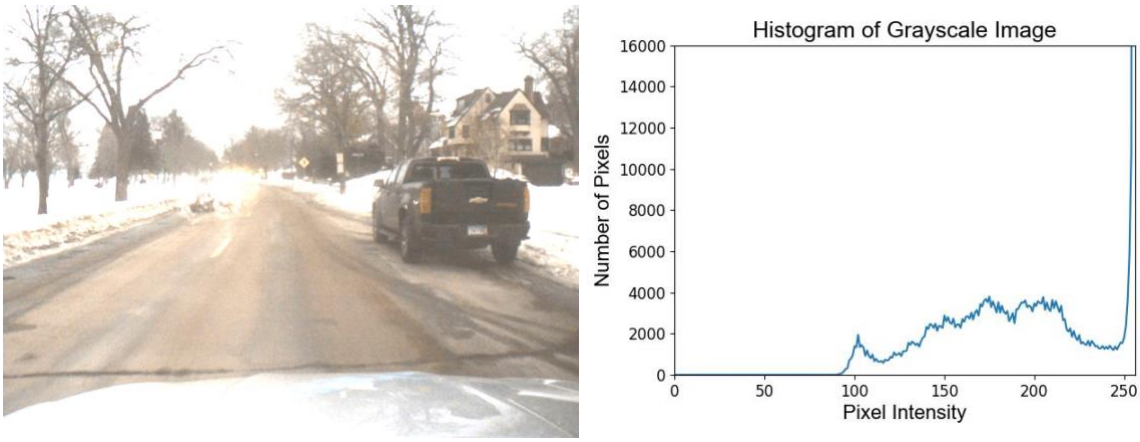


Image after Contrast Improvement implementation

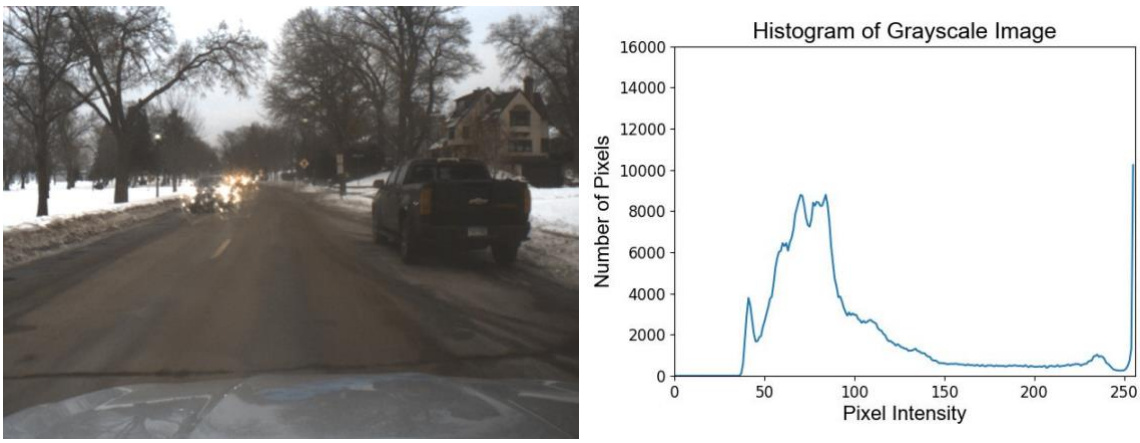


Image after DCI implementation

Figure 4.2: Advantage of DCI over Contrast Improvement, RGB image (left), histogram of grayscale image (right)

When an image is converted from pixel space to histogram space, the information about the local region of interest (ROI) and spatial cues are lost. The histogram gives information about

the overall pixel intensity distribution in the image and does not consider the local spatial cues between the features of interest. DCI can falsely interpret the histogram in some instances where,

a. The scene includes other bright features, but the ROI is darker, giving a histogram peak towards the bright side. The image remains unchanged after DCI implementation.
b. The scene includes other dark features, but the ROI is bright, giving a histogram peak towards the dark side. The image is enhanced and results in wash-out after DCI implementation.

Figure 4.3 shows an example of a scene where the road being the region of interest, is darker, and DCI implementation decides to enhance the image based on the thresholding. The image is getting washed out, as the enhancement was needed only for the ROI and not the other surrounding features.

Figure 4.3: Source Image (left), Image after DCI implementation (right)

The ideal enhancement would be focusing only on the ROI rather than improving the contrast of other features in the image. This is possible if we isolate the ROI from the rest of the image. **Figure 4.4** shows the DCI implementation on a pre-defined ROI from the image. This allows the image not to lose other important features in the process of image enhancement and get the improved contrast for the ROI simultaneously.

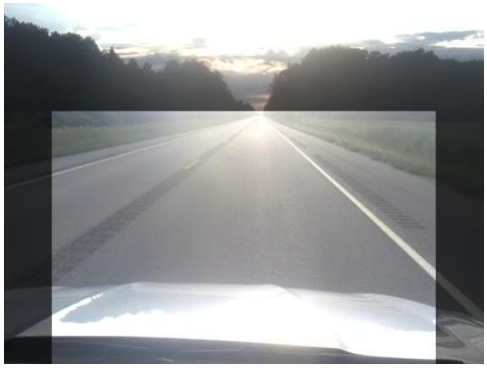


Figure 4.4: DCI implementation on pre-defined region of interest

The DCI algorithm needs to consider the region of interest for better and generic image enhancement. It is challenging to compute the ROI for different applications. For lane detection, the road is the ROI; for object detection, the surrounding is also included in the ROI. This task of extracting ROIs from an image is known as image segmentation. Various neural networks give image segmentation outputs from an image [52]. **Figure 4.5** shows the image segmentation from the MaskRCNN model [9] focusing on object detection. Similarly, various pre-trained models give image segmentation for roads and other features.

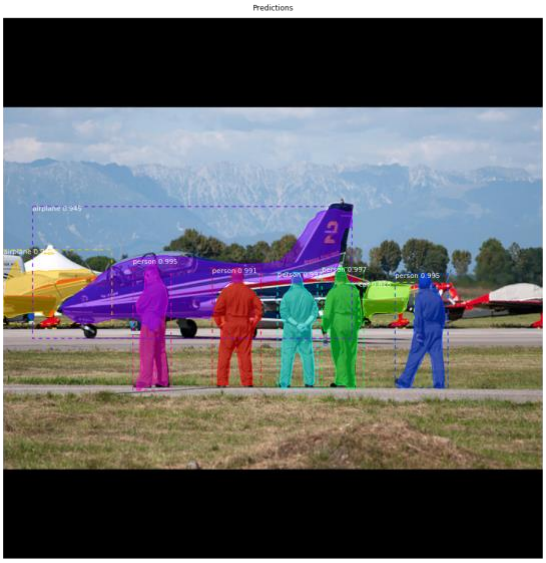


Figure 4.5: Segmentation output for MaskRCNN model

As this research focuses on improving the robustness of neural networks with a data-driven approach, we utilize the traditional image processing technique to focus on the ROIs in an image. We are keeping the image enhancement framework independent from the neural networks. Hence, we introduce contrast improvement to various ROIs of the image locally. The development of extending the DCI to various local patches of ROIs in the image is a step closer to the generalization of data-driven image enhancement.

4.2 Local-Dynamic Contrast Improvement (L-DCI)

The limitations of DCI leading to the washing out of the image due to overexposure are essential to address. The goal is to focus on small regions of interest and apply DCI to those regions. This localized image enhancement algorithm is called Local-Dynamic Contrast Improvement (L-DCI). The entire image is split into a pre-defined number of regions of a selected size, and each part is

treated as a separate ROI. Each ROI is converted from pixel to histogram space, and DCI is implemented on every ROI separately. The L-DCI algorithm is as follows:

$intensity_{threshold}$ = *peak threshold (input parameter)*
ROI = number of ROI windows in an image (input parameter)
FOR *every ROI in an image*
    $intensity_{peak}$ = *intensity corresponding to the maximum number of pixels in the histogram*
    IF $intenstiy_{peak} < intensity_{threshold}$
        $I'(x,y) = \alpha * I(x,y)$
        IF $I'(x,y) < 0$:
            $I'(x,y) = 0$
        ELSE IF $I'(x,y) > 255$:
            $I'(x,y) = 255$
    ELSE
        $I'(x,y) = I(x,y)$
NEXT *ROI*

Here, apart from the $intensity_{threshold}$,, the number of ROI windows in an image is also a controllable input parameter. **Figure 4.6** shows the implementation of DCI versus L-DCI on an RGB image. For this implementation, the number of ROIs taken is 100.

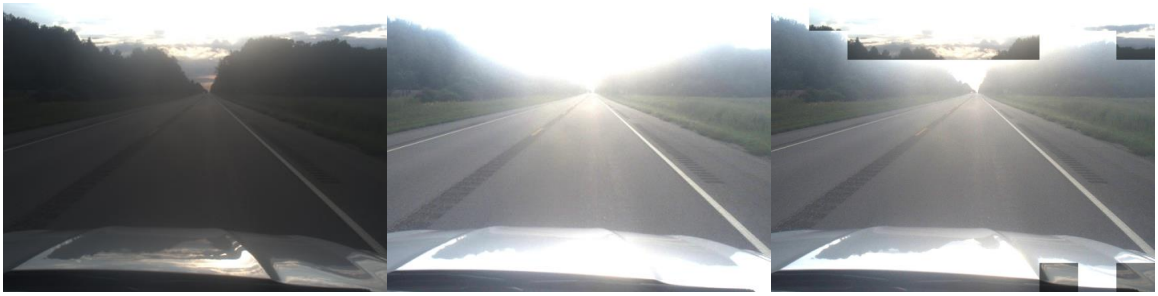


Figure 4.6: Implementation of L-DCI, Source Image (left), DCI (middle), L-DCI (right)

It is clear from the figure above that the more ROIs, the better the localized dynamic contrast enhancement for the image. However, the more the number of ROIs, the more the time complexity of the algorithm. **Figure 4.7 s**hows the implementation of L-DCI with various numbers of ROI windows to weigh the performance versus runtime of execution.

ROI: 16 (left), 25 (right)

ROI: 49 (left), 64 (right)

ROI: 100 (left), 121 (right)

Figure 4.7: Implementation of L-DCI for various number of ROIs

We performed a time complexity analysis to compute the supported frames per second (FPS) by the L-DCI algorithm for the given number of ROI windows. We observed that the supported FPS drops drastically as the number of ROIs increases. Currently, the camera hardware

used on autonomous driving cars captures a visual feed at 30 FPS. Many state-of-the-art object detection and classification algorithms can achieve FPS support comparable to the camera FPS [53,54]. For our research, we extended the expected camera FPS support to 40 and optimized the L-DCI algorithm to meet that targeted FPS by deciding the appropriate number of ROIs. **Figure 4.8** shows the FPS comparison for various ROI blocks in the L-DCI implementation.

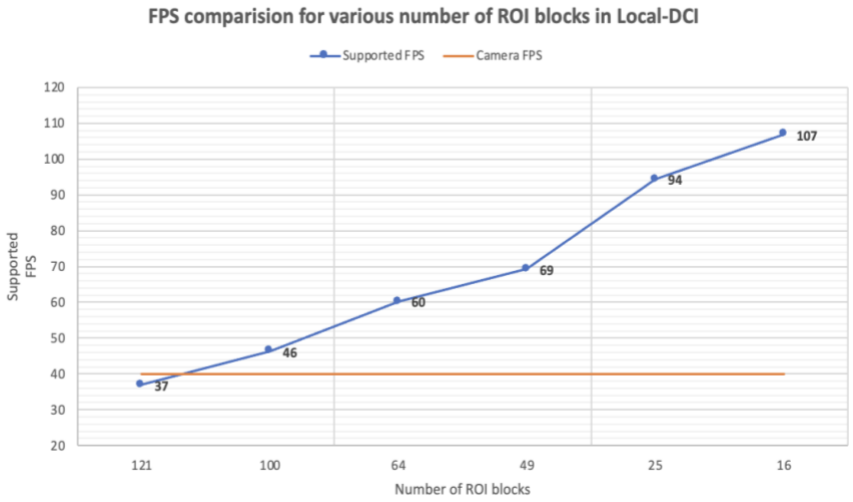


Figure 4.8: Runtime comparison of L-DCI implementation for various number of ROIs

The figure above shows that the FPS support drops below the camera FPS for 121 ROIs. To achieve the target of running the L-DCI in real-time, we decided to keep the number of ROIs at 100 for further implementations. L-DCI with 100 ROIs supports the execution of the algorithm at 46 FPS.

As shown in **Figure 4.6**, the L-DCI algorithm implements the DCI for every region of interest. The DCI algorithm does not consider the absolute values of current intensity levels present in the image. It only decides if the image needs an enhancement based on the threshold intensity. There might be a scenario where the ROI needs a contrast improvement, but it is very minute. Suppose a constant enhancement factor is applied to all the ROIs qualified for improvement. In that case, it might improve a few ROIs and lead to washing out of a few as their need for improvement was different from the constant enhancement factor. This leads to a need for a variable contrast improvement factor in the L-DCI algorithm.

4.3 Variable Local-Dynamic Contrast Improvement (VL-DCI)

The local improvements in an image give a better image than the DCI implementation. The ROIs are enhanced with a constant contrast improvement factor in L-DCI. A scale of variability is added to the enhancement factor before implementing the L-DCI on all the ROIs of the image.

This leads to Variable-Local-Dynamic Contrast Improvement Algorithm (VL-DCI). The ROIs would be enhanced with a scale based on the ratio of the pre-defined threshold intensity and the peak intensities of the histograms. The obtained improvement scale needs to be clipped to a maximum value of 3, as it is the maximum value of α. The algorithm for VL-DCI is as follows:

$intensity_{threshold}$ = peak threshold
ROI = number of ROI windows in an image (input parameter)
***FOR** every ROI in an image*
    $intensity_{peak}$ = intensity corresponding to the maximum number of pixels in the histogram
    scale = $intensity_{threshold}$ / $intensity_{peak}$
    ***IF** scale > 3*
        α = 3
    ELSE
        α = scale
    ***IF** $intenstiy_{peak}$ < $intensity_{threshold}$*
        *I'(x,y) = α * I(x,y)*
        ***IF** I'(x,y) < 0:*
            I'(x,y) = 0
        ***ELSE IF** I'(x,y) > 255:*
            I'(x,y) = 255
    ELSE
        I'(x,y) = I(x,y)
***NEXT** ROI*

Adding the scale factor allows the input parameter of $intensity_{threshold}$ to be kept constant for various datasets. The scale would take care of appropriate enhancements as it considers the i$ntensity_{peak}$ of the ROI. VL-DCI has one input parameter: the number of ROI windows in an image. By the time complexity analysis presented in the earlier section, it can be set to 100 ROIs for better results and achieving real-time execution. **Figure 4.9** shows the implementation of VL-DCI on an image split into 100 ROI windows. Four ROIs were selected randomly to visualize the histograms of intensities and the scale of enhancements.

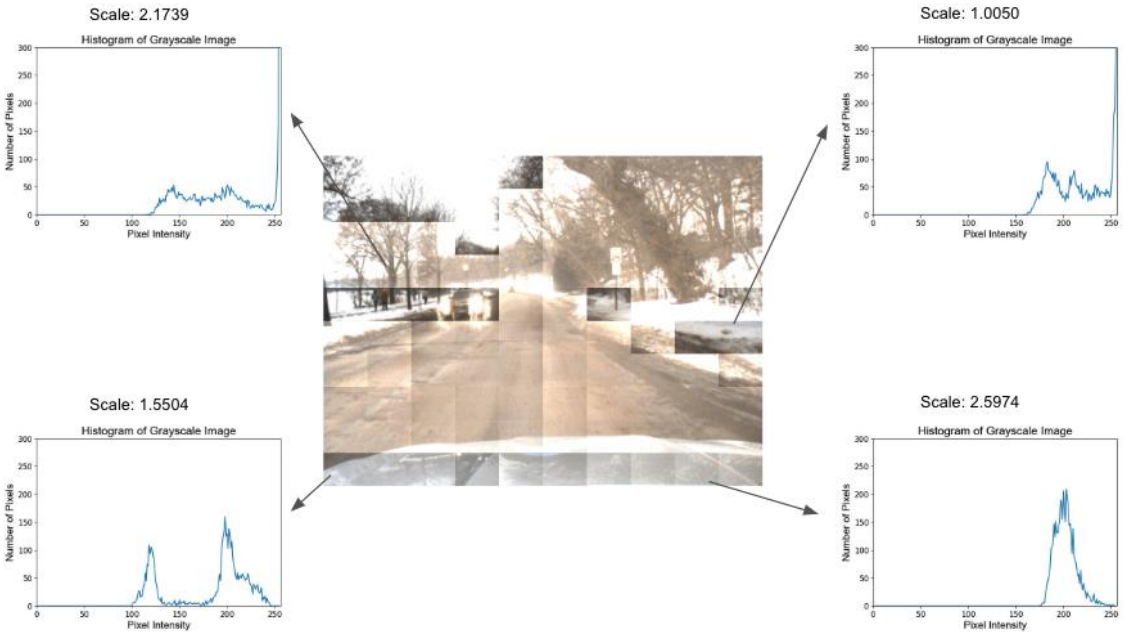


Figure 4.9**: Implementation of VL-DCI**

Figure 4.10 shows the output images from DCI, L-DCI, and VL-DCI implementations. It is evident that the poor contrast regions of the source image are improved, and the over-contrast regions are with reduced contrast in the VL-DCI implementation.

Figure 4.10: Source image (left-top), DCI (right-top), L-DCI (left-bottom), VL-DCI (right-bottom)

As VL-DCI takes the current intensity peak of the ROI into consideration, the scale plays an important role in reducing the contrast in the overexposed patches of the images. If the image is highly overexposed and the features are missing in the source image, it will not regenerate the features. However, if the overexposure is such that the features are visible in the source image, then VL-DCI would reduce the contrast and make those feature highlight. **Figure 4.11** shows VL-DCI resolving the vehicle headlight glare issue that was prominent in the CI implementation.

Figure 4.11: VL-DCI algorithm addressing the headlight glare issue, Source image (top), CI (middle), VL-DCI (bottom)

The object detection neural network was implemented on the dataset obtained from VL-DCI implementation. **Figure 4.12** shows the improvements in the recall and precision of the detections in VL-DCI from DCI and source implementations.

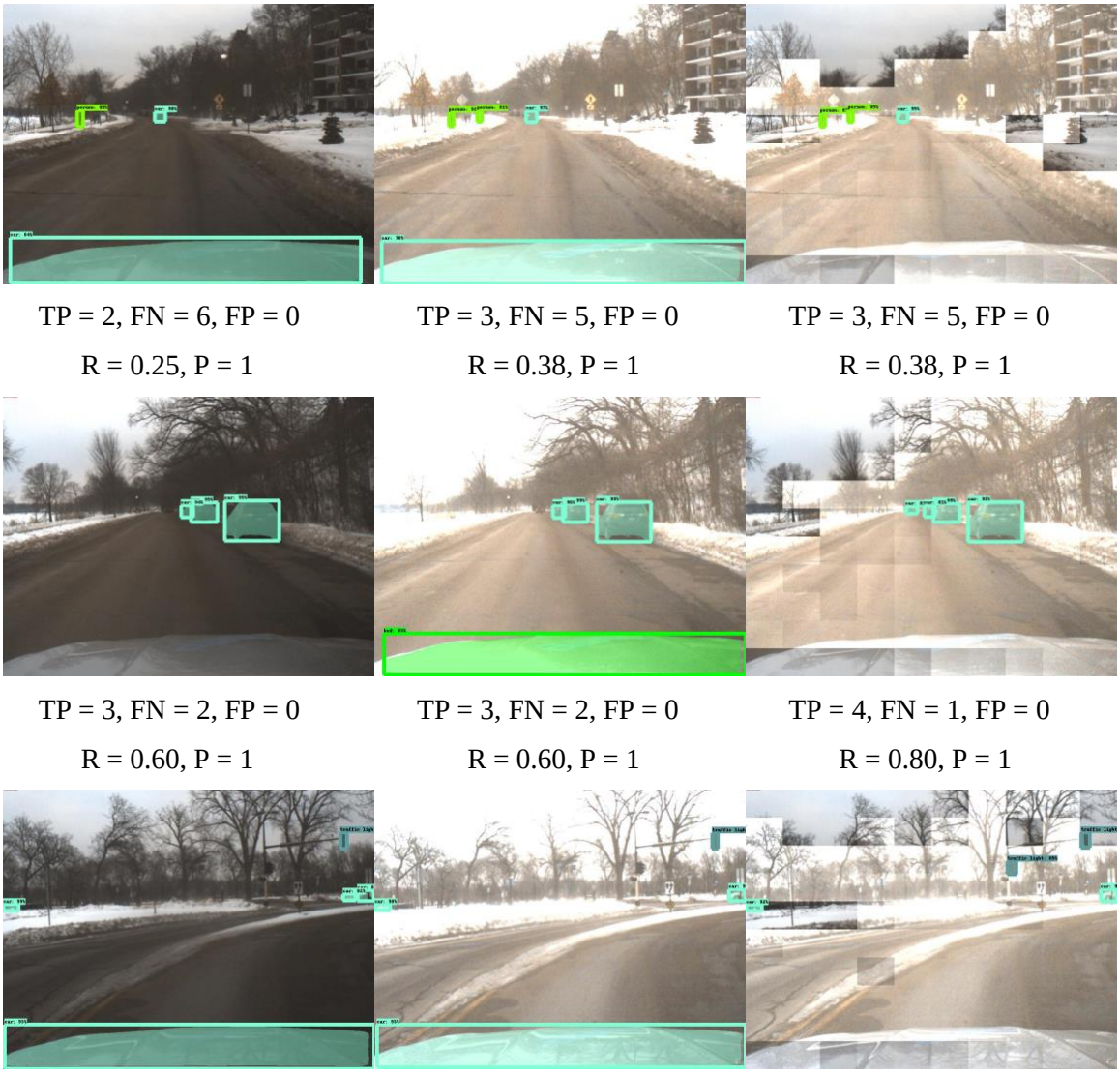


Figure 4.12: Improvement in recall and precision with VL-DCI implementation, Source image (left), DCI (middle), VL-DCI (right)

Due to the scaling of enhancement to all the ROIs in an image, the output after implementing VL-DCI is patchy. The edges appear brighter due to sudden changes in the contrast values of the adjacent ROIs. This leads to neural networks missing the detections in some cases. **Figure 4.13** shows images where the NN could not detect the vehicles in the scene.

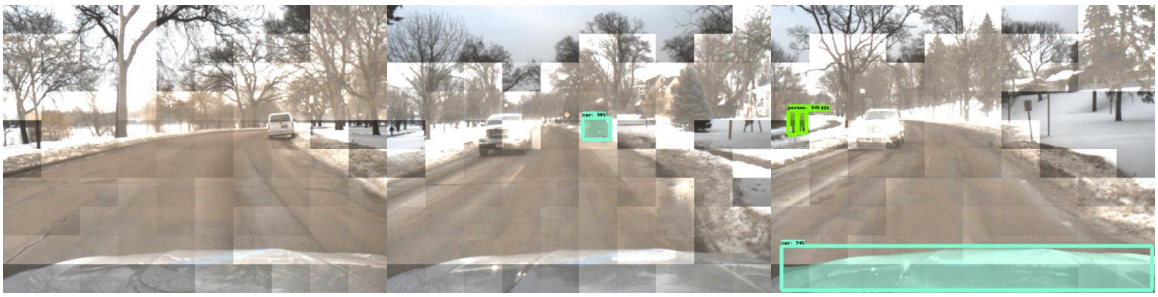


Figure 4.13: Issues of appearing edges of ROIs with VL-DCI implementation

Smoothening of the output image generated by the VL-DCI algorithm is needed to reduce the sudden differences in the contrast between the adjacent ROIs. It can be done by smoothening the ROIs to get a continuous image, improving the neural network detections.

4.4 Smoothened Variable Local-Dynamic Contrast Improvement (SVL-DCI)

The VL-DCI implementation gave rise to an image with highlighted borders shared between the ROIs having a different scale of improvement. The object detection neural network considers spatial information to detect an object confidently. If the object is split into different ROIs, as shown in **Figure 4.13**, the same object has a different level of contrast at different locations. This makes the NNs task difficult. A smooth transition between the scales of improvements for adjacent ROIs is essential. We extended the VL-DCI algorithm to smoothen the changes in the intensities at the boundaries of ROIs. This algorithm is called Smoothened Variable Local Dynamic Contrast Improvement (SVL-DCI).

We smoothened the image obtained after the implementation of VL-DCI. For smoothening an image, there are various algorithms used, such as the Gaussian filter [55], fast global image smoothening [56], and sigmoid function [57]. We used the sigmoid function as given by **equation 4.1**, as using the Gaussian filter on the image was leading to blurring the features, which was not intended.

$$S(x) = 1 / (1 + e^{x}) \quad \textbf{(4.1)}$$

Using the sigmoid function, a 2D normal distribution was obtained that would give the scale values in the range [0,3]. **Figure 4.14** shows a sigmoid distribution for a kernel size of 200x200.

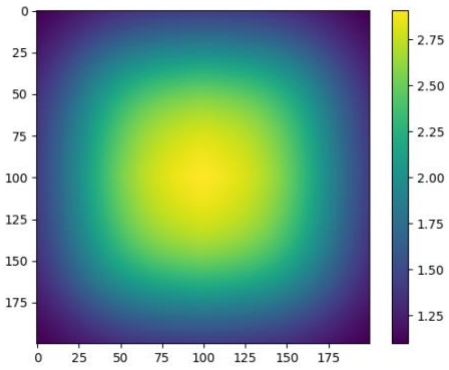


Figure 4.14: Sigmoid distribution

This sigmoid distribution was further modified with the kernel size and applied to every ROI in the image to smoothen the edges. The results are shown in F**igure 4.15**.

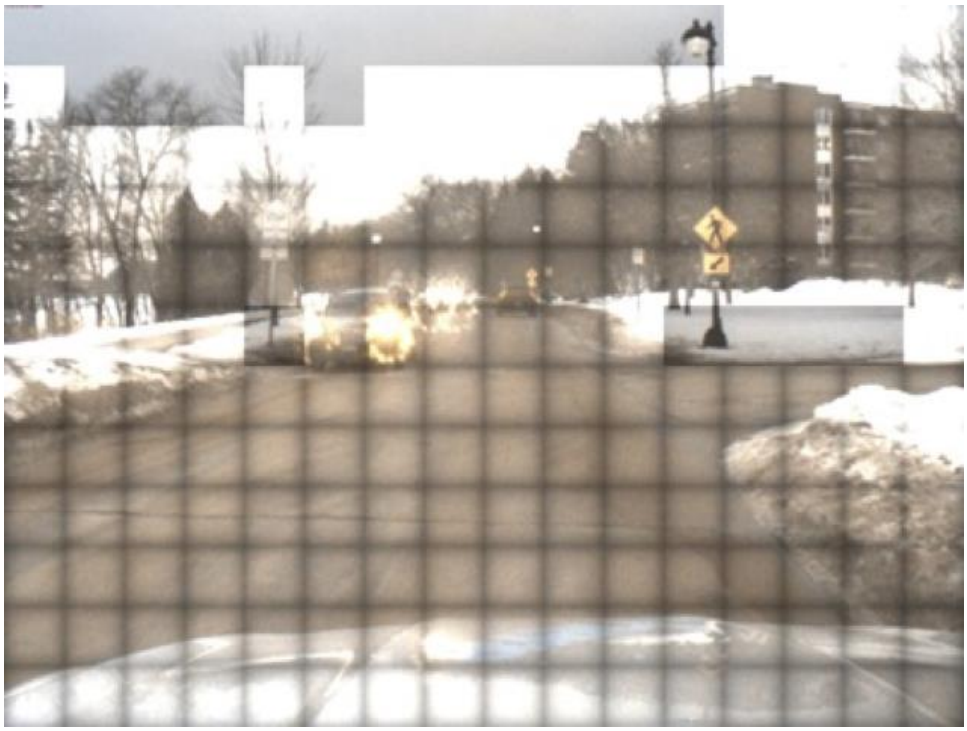


Figure 4.15: Attempt to smoothen the edges of ROIs

The output image shown in the figure above was not expected. The main issue with the current algorithm is that the image is generated with VL-DCI and then smoothened with the sigmoid after. The algorithm is modified to generate a smoothened image by utilizing matrix multiplication. When the image is split into various ROIs, and the scale is computed based on the histogram of the ROI, a scale matrix of the size of the image is created that has all the scale values associated with the corresponding pixels. Instead of applying this scale matrix to the image

and then smoothening the output image, this scale matrix is smoothened with a 2D gaussian. The smoothened scale matrix is now applied to the input image. This gives an image with the contrast enhancements as VL-DCI while not losing any features as the image is not blurred. The SVL-DCI algorithm is as follows:

$intensity_{threshold}$ = peak threshold
ROI = number of ROI windows in an image (input parameter)
scale_matrix = matrix of image size
***FOR** every ROI in an image*
    $intensity_{peak}$ = intensity corresponding to the maximum number of pixels in the histogram
    scale = $intensity_{threshold}$ / $intensity_{peak}$
    ***IF** scale > 3*
        α = 3
    ELSE
        α = scale
    ***FOR** every pixel in an ROI*
        scale_matrix = α
***NEXT** ROI*
smoothened_scale_matrix = 2D_Gaussian(scale_matrix)
*smoothened_image = image * smoothened_scale_matrix*

The scale matrix obtained before smoothening is shown in **Figure 4.16**. After applying 2D gaussian to this matrix, the edges between the ROI blocks smoothen out and give a smoothened scale matrix, as shown in **Figure 4.16**.

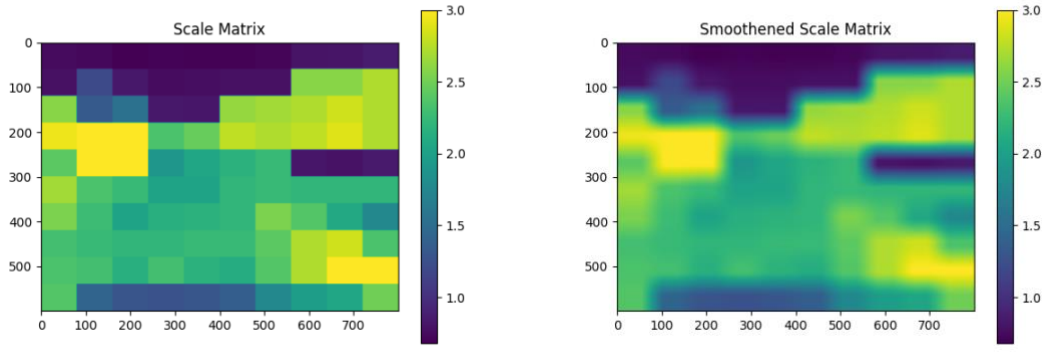


Figure 4.16: Smoothening of scale matrix

When applied to the input image, this smoothened scale matrix gives a smoothened image enhancement. **Figure 4.17** shows the gradual image enhancement with SVL-DCI versus the VL-DCI implementation.

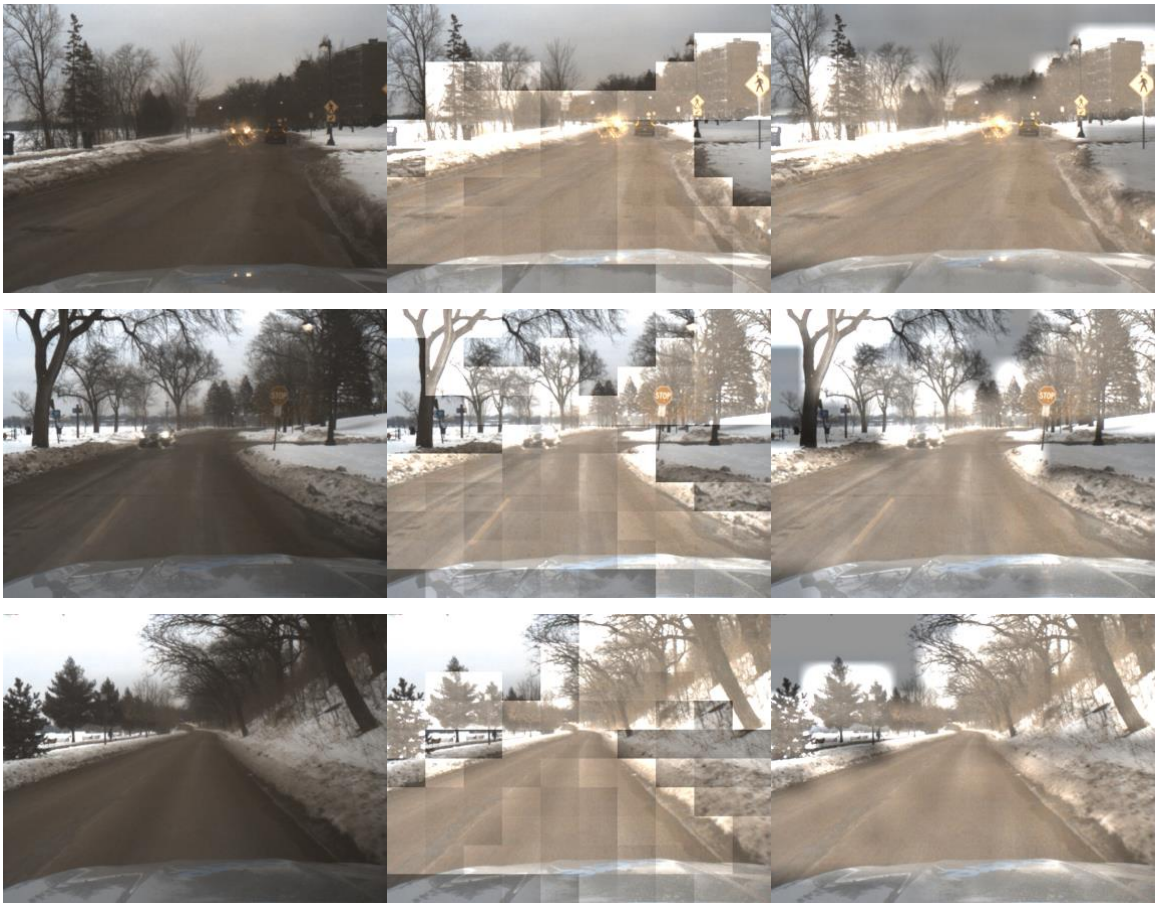


Figure 4.17: Image enhancement using SVL-DCI, source image (left), VL-DCI (middle), SVL-DCI (right)

We implemented the MaskRCNN on the SVL-DCI dataset and obtained improvements in the detections. The issue of the objects not getting detected due to the split between multiple ROIs is addressed with the implementation of SVL-DCI. **Figure 4.18** shows the improvement in the detection of SVL-DCI versus VL-DCI.

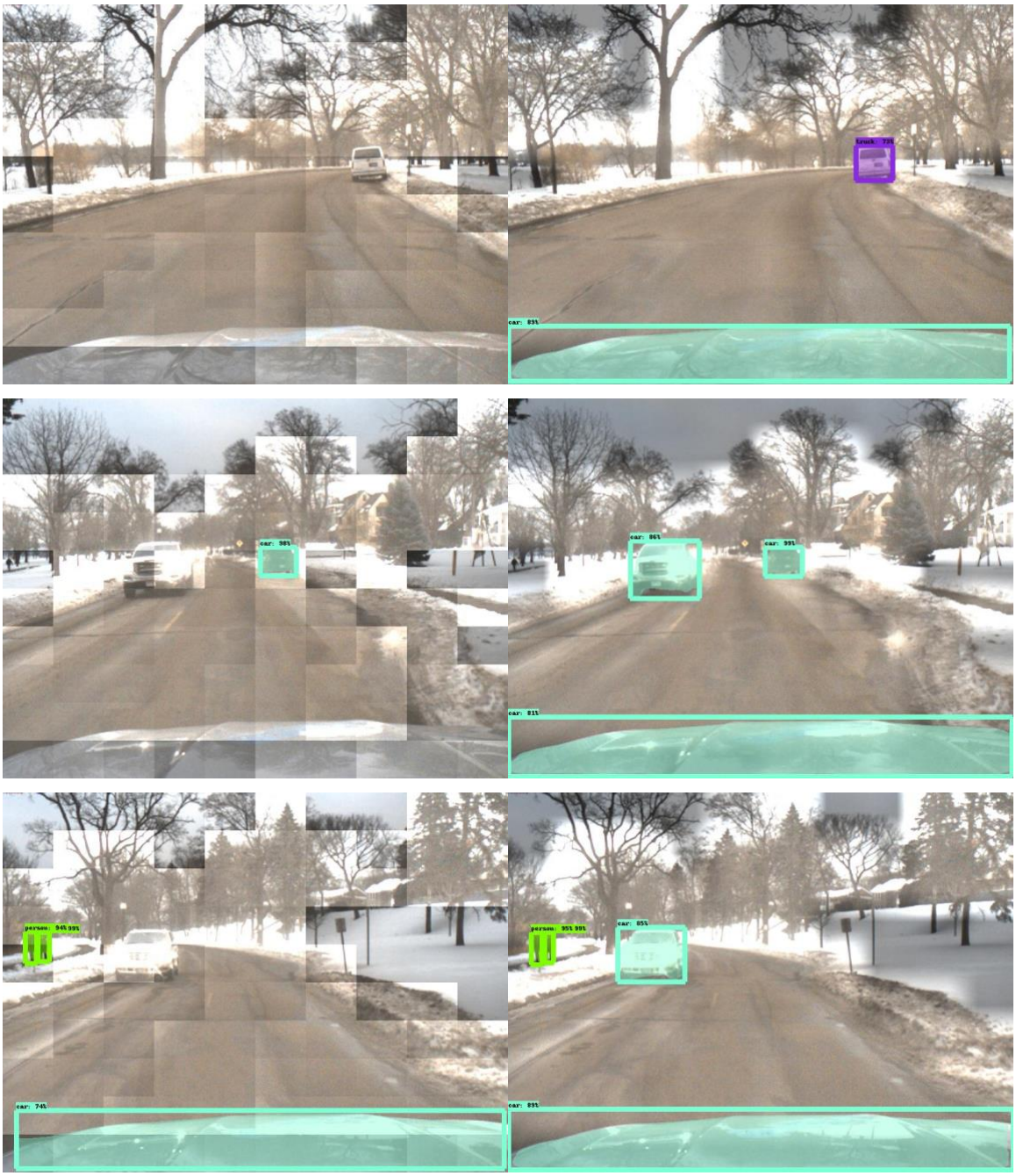


Figure 4.18: Improvement in object detection, VL-DCI (left), SVL-DCI (right)

The SVL-DCI algorithm is a generalized data-driven algorithm to improve image contrast and neural network robustness. The quantification of recall and precision of detections for the MaskRCNN is discussed in the results section. The summarized result explains how various algorithms have improved detection performance for the neural network.

Chapter 5

5. Results and Discussion

Various neural networks [58,59] use a 10% sampling of the entire dataset to generate the performance metrics of the networks. In this research, we collected a random sample of 15% of the total dataset for the object detection model to validate the proposed method. This consists of a total of 363 images out of 2470 images. The MaskRCNN model classifies the object into various classes. For our study, we focused on three classes Vehicles, Pedestrians, and Road Safety, as these classes are the most crucial for autonomous vehicles. The vehicle class considers all the various types of cars, trucks, and vans. The road safety class considers traffic lights, traffic signs, and stop signs. A manual ground truth labeling of the test data is performed for these classes.

Neural networks are examined based on various performance measures depending on the application. Taha and Hanbury, 2015 [60] have explained a few important metrics for evaluating image segmentation models. In this research, we validate the networks on three performance measures: recall and precision, accuracy, and F-score. True Positive Rate (TPR), also called recall, measures the fraction of retrieved relevant instances. Positive Predictive Value (PPV), called precision, measures the fraction of relevant instances among the retrieved instances. **Equations 2.1 and 2.2** show the formula to compute recall and precision. The accuracy of the neural network is defined as the number of correct predictions divided by the total number of predictions. **Equation 5.1** shows the formula to compute the accuracy of the neural network. F-score is the metric of performance measure that considers a weighted factor that determines the importance of recall over precision or vice versa. **Equation 5.2** shows the formula to compute the F-score of the neural network. Ye et al., 2012 [61] explained how to obtain the F-score with various values. To add more weight to precision, $\beta < 1$, and to add more weight to recall, $\beta > 1$. In our study, recall is more critical than precision. Hence, we consider $\beta = 2$, giving recall twice the importance as the precision [62].

$$\textbf{\textit{accuracy}} = (TP + TN) / (TP + TN + FP + FN) \quad \textbf{(5.1)}$$

$$\textbf{\textit{F-score}} = [(\beta^2 + 1) * \textbf{\textit{precision*recall}}] / [(\beta^2 * \textbf{\textit{precision}}) + \textbf{\textit{recall}}] \quad \textbf{(5.2)}$$

In the end, to validate the time complexities of the algorithms, we compare the runtime of execution for various developments in the contrast improvement algorithm.

5.1 Performance Comparison: Recall and Precision

We observed a promising improvement in the recall and precision of detections. **Table 5.1** shows the improved recall and precision values for various implementations. **Table 5.2** shows the percentage improvement in the performance of the MaskRCNN with contrast improvement algorithm versus VL-DCI versus SVL-DCI from the baseline implementation.

Algorithm	Recall	Precision
Baseline	**0.47**	**0.64**
Contrast Improvement	**0.50**	**0.65**
VL-DCI	**0.51**	**0.66**
SVL-DCI	**0.52**	**0.66**

Table 5.1: Improvement in the recall and precision

Algorithm	Recall	Precision
Contrast Improvement	**7.2%**	**2.6%**
VL-DCI	**8.1%**	**3.8%**
SVL-DCI	**10.4%**	**4.1%**

Table 5.2: Percentage improvement from the baseline implementation in the recall and precision

Figure 5.1 shows a graphical representation of the percentage improvement in the performance of neural network detections in baseline implementation with various algorithms. This improvement in recall and precision is critically important in autonomous driving as it represents an increase in the number of correct predictions as well as a reduction of FN and FP.

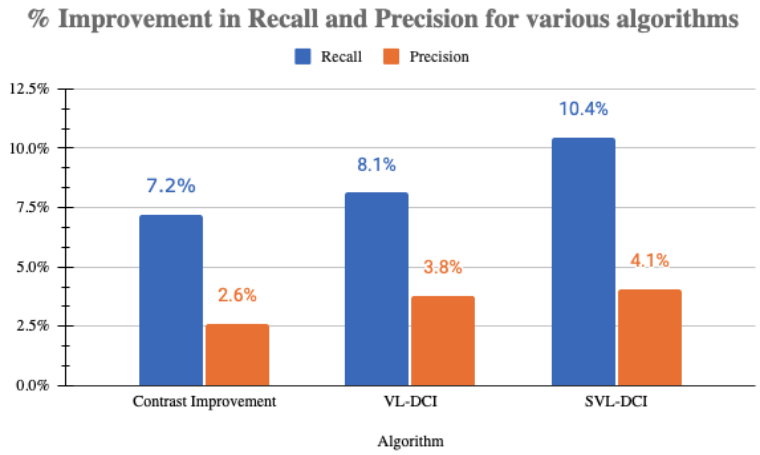


Figure 5.1: Percentage improvement from the baseline implementation in the recall and precision

5.2 Performance Comparison: Accuracy

Accuracy is the most frequently used measure of the performance of the neural network. It is a measure of correctly predicted classes. We noticed an encouraging rise in detection accuracy. **Table 5.3** shows the improved accuracy values for various algorithms. **Table 5.4** shows the percentage improvement in the accuracy of the neural network with contrast improvement algorithm versus VL-DCI versus SVL-DCI from the baseline implementation.

Algorithm	Accuracy
Baseline	**0.45**
Contrast Improvement	**0.48**
VL-DCI	**0.48**
SVL-DCI	**0.49**

Table 5.3: Improvement in the accuracy

Algorithm	Accuracy
Contrast Improvement	**6.4%**
VL-DCI	**7.2%**
SVL-DCI	**9.0%**

Table 5.4: Percentage improvement from the baseline implementation in the accuracy

Figure 5.2 depicts graphically the percentage increase in neural network detection accuracy over baseline implementation using different techniques. The gradual increase in the accuracy of the neural network signifies that logically complex image enhancement algorithms can improve the images better.

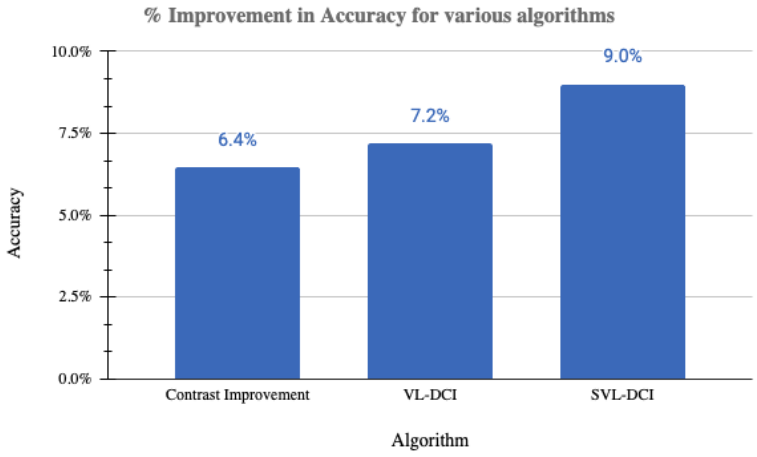


Figure 5.2: Percentage improvement from the baseline implementation in the accuracy

5.3 Performance Comparison: F-score

The F-score is a performance metric for neural networks that accounts for a weighted component that establishes the relative relevance of recall and precision. It measures classes that were incorrectly predicted. **Table 5.5** shows the improved F-score values for various implementations. **Table 5.6** shows the percentage improvement in the F-score of the network with contrast improvement algorithm versus VL-DCI versus SVL-DCI from the baseline implementation.

Algorithm	F-score
Baseline	**0.49**
Contrast Improvement	**0.52**
VL-DCI	**0.52**
SVL-DCI	**0.53**

Table 5.5: Improvement in the F-score

Algorithm	F-score
Contrast Improvement	**6.8%**
VL-DCI	**7.3%**
SVL-DCI	**9.5%**

Table 5.6: Percentage improvement from the baseline implementation in the F-score

Figure 5.3 depicts graphically the percentage increase in neural network F-score over baseline implementation using different techniques. F-score is critical, particularly in the autonomous driving application, as can be designed to give more weight to the recall over precision and the gradual increase in F-score indicates the improvement in recall.

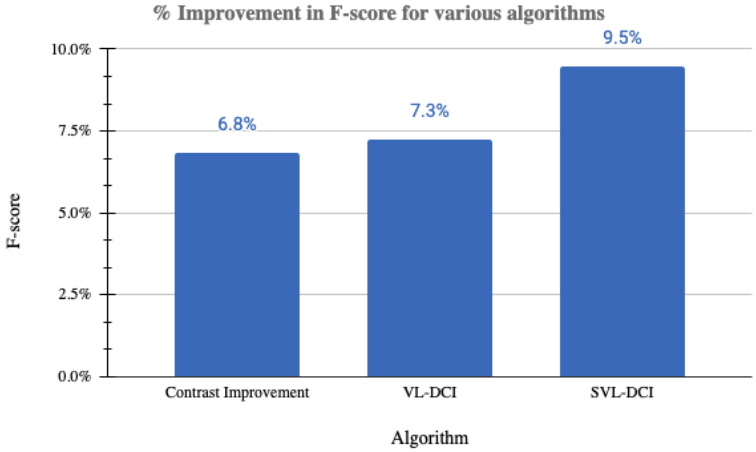


Figure 5.3: Percentage improvement from the baseline implementation in the F-score

These results of improved performance of the neural network are promising and support the hypothesis that a generalized data-driven approach can be utilized to improve the robustness and accuracy of neural networks. The SVL-DCI algorithm can be implemented on various datasets for applications such as object detection, object classification, image segmentation, pose estimation, depth perception, and other image enhancement models.

5.4 Runtime Comparison

The baseline implementation of linear contrast improvement is very naive and computationally inexpensive. As we made progress and developments in the image enhancement algorithm, the logical complexities of the algorithm increased and increased in the execution time. The histogram generation in DCI adds computational processing and increases the execution time further than the linear contrast improvement algorithm. The localization of the DCI by splitting the image into multiple ROIs is the most time-consuming step in the L-DCI algorithm, as the DCI is performed in a loop for a number of iterations equal to the number of ROIs. The optimal number of ROIs to get a real-time execution is 100. This runtime study is performed on a MacBook Pro with an Apple M1 Pro chip. System details are as follows:

Software: macOS Ventura (13.0.1), Memory: 16 GB, Hardware: 10-core CPU, 16-core GPU

To quantify the time of execution for the algorithms, we utilized frames per second (FPS) as the quantification matrix. The FPS drops sharply as we progress from contrast improvement to dynamic contrast enhancement and later to local-dynamic contrast improvement. As this research focuses on a data-driven algorithm that could perform a real-time execution, the targeted FPS support is 40. **Figure 5.4** shows the FPS comparison for various algorithms and the targeted camera FPS.

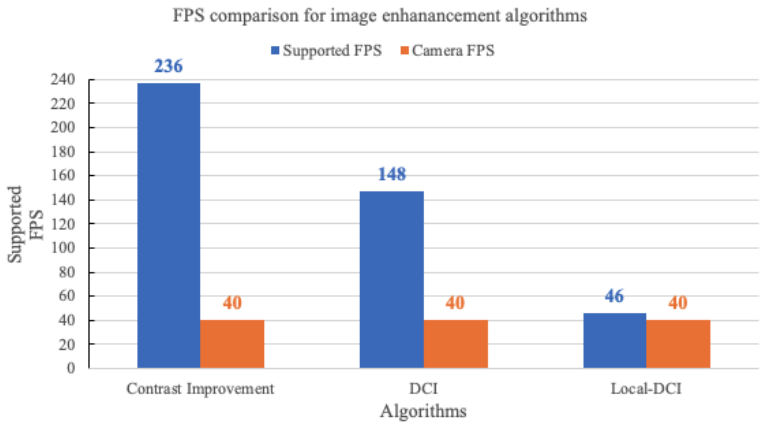


Figure 5.4: Runtime comparison in terms of FPS support for various algorithms

The figure above shows that the L-DCI algorithm supports the execution at 46 FPS. We also computed the runtime of executions for VL-DCI and SVL-DCI. The major bottleneck in the execution is splitting the image into various ROIs. Hence there was no FPS drop in the VL-DCI and SVL-DCI executions to the L-DCI implementation.

Chapter 6

6. Conclusion

Autonomous driving technology is constantly evolving, and perception is essential to this technology. Computer vision and image processing have found various applications, such as object detection, object recognition, and image enhancements, in improving the perception of autonomous vehicles. Traditional computer vision algorithms utilizing various image filtering techniques and convolution operations have limitations as the scenes in the real world are dynamically changing. Moreover, these classical algorithms are computationally expensive. Neural networks can address these limitations and perform on a wide variety of applications with considerable success. Neural networks are trained on a vast dataset, and the quality of that data decides the training weights and biases for the network. The training datasets are most of the time captured in ideal scenarios, such as a bright scene with clear weather. Vehicles operating in the real world often must deal with low light conditions and adverse weather conditions such as rain and snow. Neural network algorithms lose detection efficiency and accuracy in such scenarios. It is critical to improve the NNs' robustness and accuracy to aim to enable vehicle autonomy.

The method proposed in this research is a data-driven approach to improve the performance of neural networks. This approach focuses on dynamic image enhancement by keeping the implementation independent of neural networks. It would serve as a preprocessing framework for the dataset before passing it to the neural network. We studied various factors that affected the performance of neural networks and observed that the color and contrast of the features in an image are significant. The overall framework consists of two processing blocks: a) Image colorization and b) Image enhancement.

The image colorization block is utilized to check and correct the RGB color distribution of the image if required. The histogram of pixel intensities versus the number of pixels belonging to that intensity is used to inspect the intensity distribution of every color channel of the image. For any image, the RGB histograms should be separated from each other. The algorithm decides that the image needs a color correction if any of these overlaps. A pre-trained GAN is used to colorize the image. The input RGB image is converted to LAB space, and the GAN predicts the values of the A and B channels to generate a color correction. Once the image is color corrected, it is passed to the image enhancement block.

The image enhancement algorithm has evolved from the baseline implementation of linear contrast improvement by adding logical complexities to the approach, making it robust and dynamic. The dynamic contrast improvement algorithm considers the lighting conditions to decide whether the image needs an enhancement. The histogram of grayscale pixel intensities versus the number of pixels belonging to that intensity is plotted. Manual thresholding is applied to the histogram distribution to classify images into brighter and darker classes. Brighter images are passed through the framework without changes, and the darker images are enhanced with linear contrast improvement. The DCI is further extended to give localized enhancements to the image. An image is split into pre-defined ROIs, and DCI is applied to all the ROIs individually. This local dynamic contrast improvement helps the image enhancement to better localize throughout the image. A variable factor of scale is added to the linear contrast improvement to change the contrast of the ROIs as required. This implementation of VL-DCI gives an output image with bright edges between ROIs. A 2D gaussian is applied to the scale matrix and then convolved with the input image. The output of SVL-DCI is a smoothened enhanced image.

We implemented MaskRCNN on the datasets generated by DCI, VL-DCI, and SVL-DCI. We found promising results in the performance of the neural network. The improvement in recall from the baseline implementation is 7.2% for DCI, 8.1% for VL-DCI, and 10.1% for SVL-DCI. The improvement in precision from the baseline implementation is 2.6% for DCI, 3.8% for VL-DCI, and 4.1% for SVL-DCI. The accuracy and F-scores are also improved in a similar proportion to the baseline implementation. We quantified the execution runtime in terms of supported FPS for the various developments in the image enhancement algorithm. The baseline implementation of contrast improvement supports 236 FPS, DCI supports 148 FPS, and L-DCI supports the execution at 46 FPS.

This research concludes that SVL-DCI can be implemented on various datasets to improve the pre-trained neural network performance. SVL-DCI is a generalized image enhancement algorithm that can be utilized for various applications such as object detection, lane detection, pose estimation, and semantic segmentation.

Chapter 7

7. Future Work

7.1 Integration of Color Correction in the End-to-End Implementation

The image enhancement framework mentioned above currently considers the two aspects, namely, image colorization and contrast enhancement, separately. The user has to manually decide whether the color correction is needed by checking the color histograms of the image. Once the decision is made, the pre-trained GAN is used to colorize the image. This color-corrected image is then manually passed to the contrast enhancement block of the framework. **Figure 7.1** shows the current method in a flowchart.

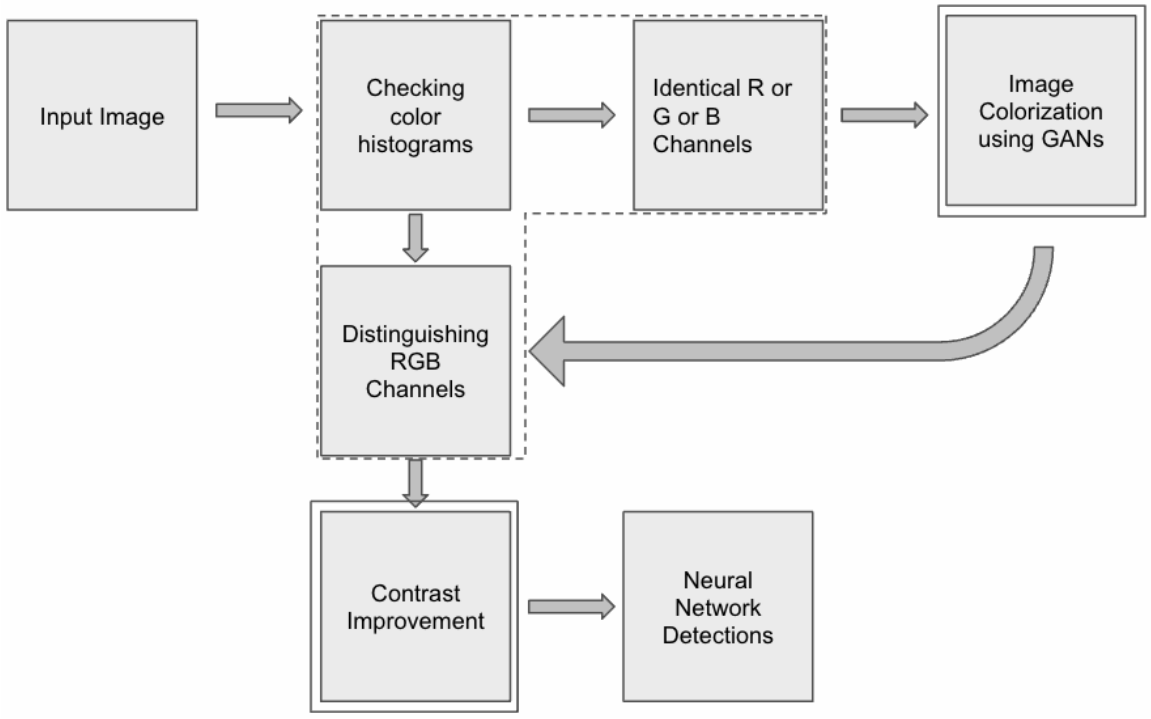


Figure 7.1: Manual implementation (blocks grouped with dashed line), end-to-end implementation (blocks with solid line)

The contrast enhancement algorithm can be further modified to take an input image and generate the color histograms. The percentage difference between the distribution of the RGB channels can be measured in order to determine the colorization of the image. The GAN can be

attached prior to implementing the contrast improvement algorithm, making the code an end-to-end execution. **Figure 7.2** shows the proposed integration of color correction in the algorithm.

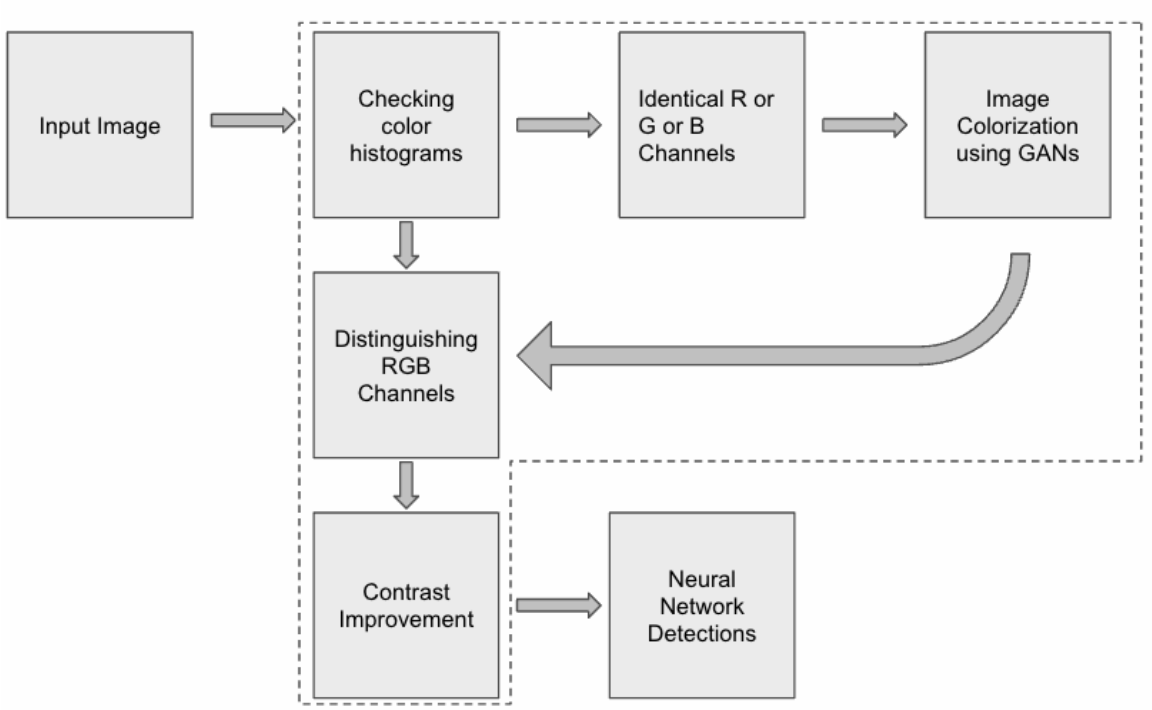


Figure 7.2: Image enhancement end-to-end implementation (blocks grouped with dashed line)

This would be an end-to-end implementation. The framework would generate an enhanced image as an output for a given input image. Incorporating image colorization in the main framework would be computationally expensive as it involves the implementation of a pre-trained GAN. This would cause the execution time to increase, resulting in an FPS drop. The algorithm would not be able to run in real-time. An alternative solution to image colorization using GAN would help maintain real-time execution.

7.2 Implementing Non-Linear Contrast Improvement

The image enhancement proposed in this research is based on a solid foundation of linear contrast enhancement. The further development of SVL-DCI is obtained after adding logical complexities to the baseline implementation of the linear contrast improvement algorithm. Consider an RGB image of size MxN pixels. Given the pixel at the location (x,y), the intensity of that pixel, I, is a vector of intensity values for R, G, and B channels given by **equation 7.1:**

$$\boldsymbol{I(x,y) = [I_R\ I_G\ I_B]^T} \qquad \textbf{(7.1)}$$

Linear Contrast Improvement function, f(I), changes the intensity of the pixel at the location (x,y), as shown below:

$$I'(x,y) = \alpha * I(x,y) + \beta \quad \textbf{(7.2)}$$

This contrast improvement function could be further modified, and a non-linear term could be introduced. Non-linear contrast improvement is useful to remove the shadows from the image and helps deal with high-contrast images [63-66]. The function f(I) is given by **equation 7.3**:

$$I'(x,y) = A * I(x,y)^{\gamma} \quad \textbf{(7.3)}$$

A is a positive constant to the intensity value, and $\boldsymbol{\gamma}$ is the non-linear factor known as the gamma value. Gamma-value less than one is called encoding gamma, which is used to make the darker regions lighter. Gamma-value more than one is called decoding gamma and is used to make the shadows darker.

There are various logarithmic transformation functions for modifying the pixel intensities. The image enhancement function f(I) for a log transformation is given by **equation 7.4**:

$$I'(x,y) = A * \log(1 + I(x,y)) \quad \textbf{(7.4)}$$

A is a positive constant to scale the pixel intensity. Log transformation is used when an image has more low-value pixels than high-value pixels. The image will be enhanced further when this technique is used on images with higher pixel values. The information on the image's features will be lost as the pixel values reach the maximum intensity limit.

The overall framework can be utilized as described in this research. SVL-DCI can be implemented on an image with any non-linear contrast improvement function. The algorithm is modular and scalable, and the image enhancement function can be modified anytime.

References

[1] R. Coppola and M. Morisio, “Connected Car,” *ACM Comput Surv*, vol. 49, no. 3, pp. 1–36, Dec. 2016.

[2] X. Zhu, Z. Gu, and Z. Wang, “Ethical challenges and countermeasures of autonomous vehicles,” in *E3S Web of Conferences*, Jan. 2021, vol. 233.

[3] D. Parekh *et al.*, “A Review on Autonomous Vehicles: Progress, Methods and Challenges,” *Electronics (Switzerland)*, vol. 11, no. 14, Jul. 2022.

[4] Q. Zou, H. Jiang, Q. Dai, Y. Yue, L. Chen, and Q. Wang, “Robust Lane Detection From Continuous Driving Scenes Using Deep Neural Networks,” *IEEE Trans Veh Technol*, vol. 69, no. 1, pp. 41–54, Jan. 2020.

[5] S. He, R. W. H. Lau, W. Liu, Z. Huang, and Q. Yang, “SuperCNN: A Superpixelwise Convolutional Neural Network for Salient Object Detection,” *Int J Comput Vis*, vol. 115, no. 3, pp. 330–344, Dec. 2015.

[6] W. Yang, J. Liu, K. Zhou, Z. Zhang, and X. Qu, “An Automatic Emergency Braking Model considering Driver’s Intention Recognition of the Front Vehicle,” *J Adv Transp*, vol. 2020, pp. 1–15, Dec. 2020.

[7] M. Mitchell *et al.*, “Model cards for model reporting,” in *FAT* 2019 - Proceedings of the 2019 Conference on Fairness, Accountability, and Transparency*, Jan. 2019, pp. 220–229

[8] D. Ulyanov, A. Vedaldi, and V. Lempitsky, “Deep Image Prior.”

[9] K. He, G. Gkioxari, P. Dollár, and R. Girshick, “Mask R-CNN,” Mar. 2017.

[10] D. Neven, B. de Brabandere, S. Georgoulis, M. Proesmans, and L. van Gool, “Towards End-to-End Lane Detection: an Instance Segmentation Approach,” Feb. 2018.

[11] G. Balasekaran, S. Jayakumar, and R. Pérez de Prado, “An intelligent task scheduling mechanism for autonomous vehicles via deep learning,” *Energies (Basel)*, vol. 14, no. 6, Mar. 2021.

[12] K. Khan, W. Ahmad, M. N. Amin, and A. Ahmad, “A Systematic Review of the Research Development on the Application of Machine Learning for Concrete,” *Materials*, vol. 15, no. 13. MDPI, Jul. 01, 2022.

[13] R. Bhardwaj, A. R. Nambiar, and D. Dutta, “A Study of Machine Learning in Healthcare,” in *Proceedings - International Computer Software and Applications Conference*, Sep. 2017, vol. 2, pp. 236–241.

[14] N. A. Jalil, H. J. Hwang, and N. M. Dawi, “Machines learning trends, perspectives and prospects in education sector,” in *ACM International Conference Proceeding Series*, Jul. 2019, pp. 201–205.

[15] A. Arnab and P. H. S. Torr, “Pixelwise Instance Segmentation with a Dynamically Instantiated Network.”

[16] F. Liu, C. Shen, and G. Lin, “Deep Convolutional Neural Fields for Depth Estimation from a Single Image.”

[17] D. Ciregan, U. Meier, and J. Schmidhuber, “Multi-column deep neural networks for image classification,” in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2012, pp. 3642–3649.

[18] A. Tavanaei, M. Ghodrati, S. R. Kheradpisheh, T. Masquelier, and A. Maida, “Deep learning in spiking neural networks,” *Neural Networks*, vol. 111, pp. 47–63, Mar. 2019.

[19] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, “Image-to-Image Translation with Conditional Adversarial Networks,” Nov. 2016.

[20] M. Buechel *et al.*, “An Automated Electric Vehicle Prototype Showing New Trends in Automotive Architectures,” in *2015 IEEE 18th International Conference on Intelligent Transportation Systems*, Sep. 2015, pp. 1274–1279.

[21] B. Dogan and D. Erol, “The Future of Fossil and Alternative Fuels Used in Automotive Industry,” in *2019 3rd International Symposium on Multidisciplinary Studies and Innovative Technologies (ISMSIT)*, Oct. 2019, pp. 1–8.

[22] K. Othman, “Exploring the implications of autonomous vehicles: a comprehensive review,” *Innovative Infrastructure Solutions*, vol. 7, no. 2, p. 165, Apr. 2022.

[23] K. Othman, “Public acceptance and perception of autonomous vehicles: a comprehensive review,” *AI and Ethics*, vol. 1, no. 3, pp. 355–387, Aug. 2021.

[24] J. B. Greenblatt and S. Shaheen, “Automated Vehicles, On-Demand Mobility, and Environmental Impacts,” *Current Sustainable/Renewable Energy Reports*, vol. 2, no. 3. Springer Nature, pp. 74–81, Sep. 01, 2015.

[25] B. C. Zanchin, R. Adamshuk, M. M. Santos, and K. S. Collazos, “On the instrumentation and classification of autonomous cars,” in *2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, Oct. 2017, pp. 2631–2636.

[26] J. Kocic, N. Jovicic, and V. Drndarevic, “Sensors and Sensor Fusion in Autonomous Vehicles,” in *2018 26th Telecommunications Forum (TELFOR)*, Nov. 2018, pp. 420–425

[27] E. Uhlemann, “Time for Autonomous Vehicles to Connect [Connected Vehicles],” *IEEE Vehicular Technology Magazine*, vol. 13, no. 3, pp. 10–13, Sep. 2018.

[28] E. Martí, M. Á. de Miguel, F. García, and J. Pérez, “A Review of Sensor Technologies for Perception in Automated Driving,” *IEEE Intelligent Transportation Systems Magazine*, vol. 11, no. 4, pp. 94–108, Dec. 2019.

[29] V. K. Kukkala, J. Tunnell, S. Pasricha, and T. Bradley, “Advanced Driver-Assistance Systems: A Path Toward Autonomous Vehicles,” *IEEE Consumer Electronics Magazine*, vol. 7, no. 5, pp. 18–25, Sep. 2018.

[30] Jianbo Shi and Tomasi, “Good features to track,” in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition CVPR-94*, 1994, pp. 593–600.

[31] G. Kumar and P. K. Bhatia, “A Detailed Review of Feature Extraction in Image Processing Systems,” in *2014 Fourth International Conference on Advanced Computing & Communication Technologies*, Feb. 2014, pp. 5–12.

[32] H.-J. Yoo, “Deep Convolution Neural Networks in Computer Vision: a Review,” *IEIE Transactions on Smart Processing and Computing*, vol. 4, no. 1, pp. 35–43, Feb. 2015.

[33] K. Patel, K. Rambach, T. Visentin, D. Rusev, M. Pfeiffer, and B. Yang, “Deep learning-based object classification on automotive radar spectra,” in *2019 IEEE Radar Conference, RadarConf 2019*, Apr. 2019.

[34] L. Xu, A. Krzyżak, and C. Y. Suen, “Methods of Combining Multiple Classifiers and Their Applications to Handwriting Recognition,” *IEEE Trans Syst Man Cybern*, vol. 22, no. 3, pp. 418–435, 1992.

[35] K. Wongsritong, K. Kittayaruasiriwat, F. Cheevasuvit, K. Dejhan, and A. Somboonkaew, “Contrast enhancement using multipeak histogram equalization with brightness preserving,” in *IEEE. APCCAS 1998. 1998 IEEE Asia-Pacific Conference on Circuits and Systems. Microelectronics and Integrating Systems. Proceedings (Cat. No.98EX242)*, pp. 455–458.

[36] J. A. Stark, “Adaptive image contrast enhancement using generalizations of histogram equalization,” *IEEE Transactions on Image Processing*, vol. 9, no. 5, pp. 889–896, May 2000.

[37] Soong-Der Chen and A. R. Ramli, “Minimum mean brightness error bi-histogram equalization in contrast enhancement,” *IEEE Transactions on Consumer Electronics*, vol. 49, no. 4, pp. 1310–1319, Nov. 2003.

[38] Chi-Chia Sun, Shanq-Jang Ruan, Mon-Chau Shie, and Tun-Wen Pai, “Dynamic contrast enhancement based on histogram specification,” *IEEE Transactions on Consumer Electronics*, vol. 51, no. 4, pp. 1300–1305, Nov. 2005.

[39] K. R. Gegenfurtner and D. C. Kiper, “Color vision,” *Annual Review of Neuroscience*, vol. 26. pp. 181–206, 2003.

[40] I. Goodfellow *et al.*, “Generative adversarial networks,” *Commun ACM*, vol. 63, no. 11, pp. 139–144, Oct. 2020.

[41] A. Creswell, T. White, V. Dumoulin, K. Arulkumaran, B. Sengupta, and A. A. Bharath, “Generative Adversarial Networks: An Overview,” *IEEE Signal Process Mag*, vol. 35, no. 1, pp. 53–65, Jan. 2018.

[42] R. Zhang, P. Isola, and A. A. Efros, “Colorful Image Colorization,” Mar. 2016.

[43] L. Guo, L. Liang, S. Zeng, J. He, and G. Dai, “Gray Scale Image Coloring Method Based on GAN,” in *Proceedings of the 3rd International Conference on Data Science and Information Technology*, Jul. 2020, pp. 71–75.

[44] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, “Image-to-Image Translation with Conditional Adversarial Networks,” Nov. 2016.

[45] Moein Shariatnia, “Image Colorization,” *https://github.com/moein-shariatnia/Deep-Learning/tree/main/Image%20Colorization%20Tutorial*.

[46] Y. Li, Y. Fu, H. Li, and S. W. Zhang, “The improved training algorithm of back propagation neural network with selfadaptive learning rate,” in *Proceedings of the 2009 International Conference on Computational Intelligence and Natural Computing, CINC 2009*, 2009, no. 1, pp. 73–76.

[47] A. Pavelka and A. Procházka, “ALGORITHMS FOR INITIALIZATION OF NEURAL NETWORK WEIGHTS.”

[48] Minnesota Department of Natural Resources, “Average Annual Snowfall in Minnesota,” *https://www.dnr.state.mn.us/climate/faqs.html*.

[49] U.S. DEPARTMENT OF COMMERCE, “Weekly Weather and Crop Bulletin,” 2022.

[50] T. Arici and Y. Altunbasak, “Image local contrast enhancement using adaptive non-linear filters,” in *Proceedings - International Conference on Image Processing, ICIP*, 2006, pp. 2881–2884.

[51] S. Ghosh, S. Hazra, S. P. Maity, and H. Rahaman, “A new algorithm for grayscale image histogram computation,” in *2015 Annual IEEE India Conference (INDICON)*, Dec. 2015, pp. 1–6.

[52] N. Araslanov and S. Roth, “Single-Stage Semantic Segmentation from Image Labels.”

[53] H. Gao, B. Cheng, J. Wang, K. Li, J. Zhao, and D. Li, “Object Classification Using CNN-Based Fusion of Vision and LIDAR in Autonomous Vehicle Environment,” *IEEE Trans Industr Inform*, vol. 14, no. 9, pp. 4224–4231, Sep. 2018.

[54] M. Yahiaoui *et al.*, “FisheyeMODNet: Moving Object detection on Surround-view Cameras for Autonomous Driving,” Aug. 2019.

[55] F. Cabello, J. Leon, Y. Iano, and R. Arthur, “Implementation of a fixed-point 2D Gaussian Filter for Image Processing based on FPGA,” in *2015 Signal Processing: Algorithms, Architectures, Arrangements, and Applications (SPA)*, Sep. 2015, pp. 28–33.

[56] D. Min, S. Choi, J. Lu, B. Ham, K. Sohn, and M. N. Do, “Fast Global Image Smoothing Based on Weighted Least Squares,” *IEEE Transactions on Image Processing*, vol. 23, no. 12, pp. 5638–5653, Dec. 2014.

[57] N. Hassan and N. Akamatsu, “A new approach for contrast enhancement using sigmoid function,” *The International Arab Journal of Information Technology*, vol. 1, pp. 221–226, 2004.

[58] R. Girshick, J. Donahue, T. Darrell, and J. Malik, “Rich feature hierarchies for accurate object detection and semantic segmentation,” Nov. 2013.

[59] P. Lu, C. Cui, S. Xu, H. Peng, and F. Wang, “SUPER: A Novel Lane Detection System,” *IEEE Transactions on Intelligent Vehicles*, vol. 6, no. 3, pp. 583–593, Sep. 2021.

[60] A. A. Taha and A. Hanbury, “Metrics for evaluating 3D medical image segmentation: analysis, selection, and tool,” *BMC Med Imaging*, vol. 15, no. 1, p. 29, Dec. 2015.

[61] N. Ye, K. A. Ming Chai, and H. Leong Chieu, “Optimizing F-Measures: A Tale of Two Approaches,” 2012. [Online]. Available: http://www.comp.nus.edu.sg/

[62] M. P. Popovi´c, “CHRF: character n-gram F-score for automatic MT evaluation,” Association for Computational Linguistics, 2015.

[63] X. Guan, S. Jian, P. Hongda, Z. Zhiguo, and G. Haibin, “An Image Enhancement Method Based on Gamma Correction,” in *2009 Second International Symposium on Computational Intelligence and Design*, 2009, pp. 60–63.

[64] H. Farid, “Blind inverse gamma correction,” *IEEE Transactions on Image Processing*, vol. 10, no. 10, pp. 1428–1433, 2001.

[65] S. Rahman, M. M. Rahman, M. Abdullah-Al-Wadud, G. D. Al-Quaderi, and M. Shoyaib, “An adaptive gamma correction for image enhancement,” *EURASIP J Image Video Process*, vol. 2016, no. 1, p. 35, Dec. 2016.

[66] S. Yelmanov and Y. Romanyshyn, “Image contrast enhancement in automatic mode by nonlinear stretching,” in *2018 XIV-th International Conference on Perspective Technologies and Methods in MEMS Design (MEMSTECH)*, Apr. 2018, pp. 104–108.