LANE DETECTION SYSTEM USING AI EDGE

A MINI-PROJECT REPORT

18CSC305J - ARTIFICIAL INTELLIGENCE

Submitted by

SHERAPHY SNEHA J S(RA2011003010445)

JAHNAVI S(RA2011003010457)

ANUSHKA LONDHE(RA2011003010463)

Under the guidance of

Dr. M ARUNA

Assistant Professor, *Department of Computing Technologies*

in partial fulfillment for the award of the degree

of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE & ENGINEERING

of

FACULTY OF ENGINEERING AND TECHNOLOGY

S.R.M. Nagar, Kattankulathur, Chengalpattu District

MAY 2023

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

(Under Section 3 of UGC Act, 1956)

BONAFIDE CERTIFICATE

Certified that Mini project report titled **"LANE DETECTION USING AI EDGE DETECTION"** is the bona fide work of **MS. SHERAPHY SNEHA J S (RA2011003010445), MS. JAHNAVI S (RA2011003010457), AND MS. ANUSHKA LONDHE (RA2011003010463)** who carried out the minor project under my supervision. Certified further, that to the best of my knowledge, the work reported herein does not form any other project report or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

SIGNATURE

DR. M ARUNA

GUIDE

Assistant Professor

Department of Computing Technologies

SIGNATURE

Dr. M. Pushpalatha

HEAD OF THE DEPARTMENT

Professor & Head

Department of Computing Technologies

ACKNOWLEDGEMENT

We express our humble gratitude to **Dr. C. Muthamizhchelvan**, Vice-Chancellor, SRM Institute of Science and Technology, for the facilities extended for the project work and his continued support.

We extend our sincere thanks to Dean-CET, SRM Institute of Science and Technology, **Dr. T.V. Gopal**, for his invaluable support.

We wish to thank **Dr. Revathi Venkataraman**, Professor & Chairperson, School of Computing, SRM Institute of Science and Technology, for her support throughout the project work.

We are incredibly grateful to our Head of the Department, **Dr. M. Lakshmi,** Professor, Department of Data Science and Business Systems, SRM Institute of Science and Technology, for her suggestions and encouragement at all stages of the project work.

We want to convey our thanks to our Project Coordinator, **Dr. P. Rajasekar**, Assistant Professor, Panel Head, **Dr. A. Murugan**, Associate Professor, and members, **Dr. R. Rajkumar,** Assistant Professor, **Dr. T. Karthick,** Assistant Professor and **Dr. S. Jeeva,** Assistant Professor, Department of Data Science and Business Systems, SRM Institute of Science and Technology, for their inputs during the project reviews and support.

Our inexpressible respect and thanks to our guide, **Dr. Aruna Anguvel**, Associate Professor, Department of Data Science and Business Systems, SRM Institute of Science and Technology, for providing us with an opportunity to pursue our project under his mentorship. He provided us with the freedom and support to explore the research topics of our interest. His passion for solving problems and making a difference in the world has always been inspiring.
We sincerely thank the Data Science and Business Systems Department staff and students, SRM Institute of Science and Technology, for their help during our project. Finally, we would like to thank parents, family members, and friends for their unconditional love, constant support, and encouragement.

ABSTRACT

A lane detection system is an essential component of advanced driver assistance systems (ADAS) that enables vehicles to perceive and react to their surroundings.

This system uses various computer vision techniques to identify and track the lane markings on the road and alert the driver if the vehicle deviates from its lane. A lane detection system is an automated technology designed to identify and track the lanes on a road and provide feedback to drivers. The system employs image processing algorithms to extract the road's boundaries from a camera's live feed and to identify the lane markings. It can identify lane departures and provide warnings to drivers to prevent accidents. The system can be used in a variety of applications, including self-driving cars, advanced driver-assistance systems (ADAS), and traffic control systems. Lane detection systems are becoming increasingly popular as a means of improving road safety and reducing accidents caused by driver error.

The Lane Detection System using AI Edge Detection is a cutting-edge technology that utilizes artificial intelligence and computer vision algorithms to identify and track lane markings on roads. This system employs edge detection techniques to extract useful features from the images captured by a camera mounted on a vehicle. The extracted features are then processed by an AI algorithm to accurately detect lane markings and determine the position of the vehicle relative to the lane. The system is highly effective in detecting lane markings in challenging environments and can operate in real-time, providing a valuable tool for enhancing driver safety and improving vehicle automation capabilities.

TABLE OF CONTENTS

LIST OF FIGURES

ABBREVIATIONS

AI	Artificial Intelligence
ADAS	Advanced Driver Assistance Systems
ML	Machine Learning
CNN	Convolution Neural Network
3D	3 Dimensional
ROI	Region of Interest
GPS	Global Positioning System

CHAPTER 1
INTRODUCTION

1.1 GENERAL

Lane detection is a key technology in the field of artificial intelligence(AI)and computer vision, which involves detecting and tracking lanes on a road or highway. The goal of lane detection is to provide accurate information to a self-driving car or assistive driving system to help it navigate safely on the road.

In a typical lane detection system, a camera or sensor mounted on the vehicle captures images of the road ahead. The ai algorithm analyzes these images to identify the lane markings, including lane boundaries, road edges, and other traffic markings such as arrows or symbols. The algorithm may use various techniques such as edge detection, color segmentation, and machine learning to accurately detect and track the lanes.

Once the lanes are detected, the AI system can make decisions based on the position and trajectory of the vehicle relative to the lanes. For example, if the vehicle starts to drift out of the lane, the system can issue a warning or take corrective action to steer the car back into the lane.

Lane detection is an essential component of many self-driving car and advanced driver assistance systems (ADAS), which rely on AI and computer vision to provide safe and reliable driving experiences. It is a rapidly evolving field that promises to revolutionize the way we travel and commute.

1.2 PURPOSE

The purpose of lane detection in AI is to enable autonomous vehicles to accurately identify and track the lanes on a road or highway, which is a critical step in the process of driving safely and efficiently. Lane detection allows the AI system to make decisions based on the position and trajectory of the vehicle relative to the lanes, including maintaining the appropriate speed, avoiding collisions, and staying within the designated driving lane.

Furthermore, lane detection is a key technology in the development of advanced driver assistance systems (ADAS), which provide drivers with real-time feedback and warnings to help prevent accidents and improve overall driving safety. By using AI algorithms to detect and track lanes, ADAS can provide a range of features, including lane departure warnings, adaptive cruise control, and automatic lane centering.

Overall, the purpose of lane detection in AI is to enhance the safety and efficiency of driving by providing accurate and reliable information to autonomous vehicles and ADAS systems.

1.3 SCOPE

The scope of lane detection in AI is vast and continually evolving. Lane detection is a fundamental building block for the development of autonomous vehicles and advanced driver assistance systems (ADAS), which aim to provide safer and more efficient driving experiences. Here are some specific areas where lane detection is applicable:

- **Autonomous Driving:** Lane detection is a critical component in the development of autonomous vehicles. Self-driving cars rely on AI algorithms to detect and track lanes accurately, enabling them to navigate safely on the road without human intervention.

- **ADAS:** Lane detection is an essential technology for ADAS, which provides real-time feedback and warnings to drivers to prevent accidents and improve overall driving safety. By using AI to detect and track lanes, ADAS can provide a range of features, including lane departure warnings, adaptive cruise control, and automatic lane centering.

- **Traffic Management:** Lane detection can be used in traffic management systems to monitor and regulate traffic flow, reduce congestion, and improve safety. By analyzing the flow of vehicles within different lanes, traffic management systems can make real-time adjustments to optimize traffic flow and prevent accidents.

- **Road Maintenance:** Lane detection can also be used to detect and monitor road markings, including lane boundaries and other traffic symbols. This information can be used to identify and prioritize road maintenance needs, ensuring that roads remain safe and efficient for drivers.

Overall, the scope of lane detection in AI is vast and has the potential to revolutionize the way we travel and commute. As the technology continues to evolve, we can expect to see more advanced applications of lane detection in a variety of fields, from transportation to infrastructure and beyond.

1.4. Artificial Intelligence, Machine Learning and Deep Learning

Artificial Intelligence is a subset of computational theory and logic which focuses on the development of computer systems and algorithms designed to perform tasks that usually require human intelligence, logical thinking and expertise. These programs and applications imitate the behaviour of the human mind, hence require superabundant knowledge related to all the variables of the problem. These include the various objects, their properties, differentiating categories and the relationships between them. Artificial Intelligence can be broadly classified into two types depending on the kind of problems they are developed to solve;

- **Vertical AI:** These AI algorithms focus on learning to solve a single problem. They are usually restrictively programmed to follow the instructions for a single, automated, and repetitive task. For example, scheduling meetings and calls on a daily basis, and automatically updating the database with information obtained from a single external source.

- **Horizontal AI:** These AI algorithms focus on broader problem statements. Applicationsclassified under this type of AI have the ability to handle multiple tasks and cater to allmultiple needs of their user with a single logic and setting. For example, virtualassistants like Siri, Cortana and Alexa that tackle multiple tasks for their users.

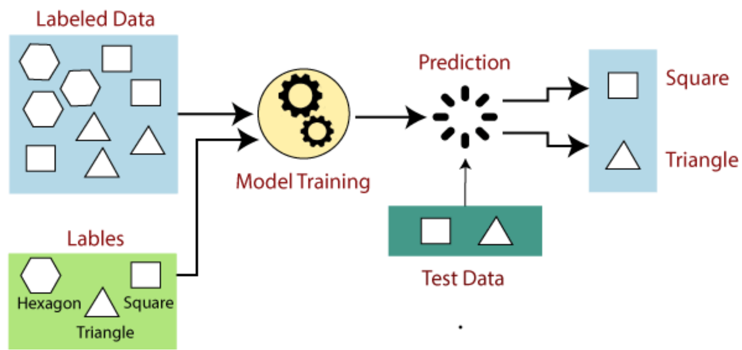
Machine Learning (ML), though often confused and interchangeably used with Artificial Intelligence, is a subset of artificial intelligence and is the science of developing machine intelligence algorithms that help the machine learn from existing and past data. The algorithms achieve this by identifying and learning the various relationships between individual features of the data, looking for common patterns, responding to different situations outside of their programming restrictions, and making predictions accordingly.

Machine Learning can be broadly classified into three categories (figure 1.4.1) based on the kind of data they use to learn and the type of outputs produced by them. They are:

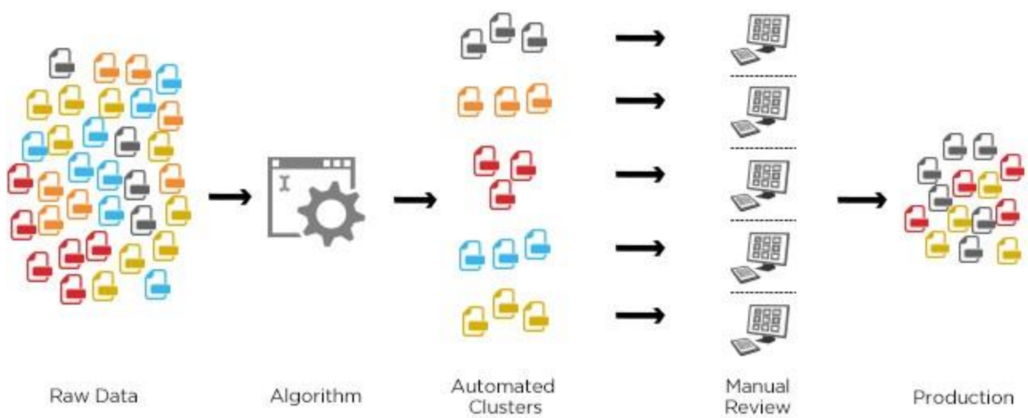
- **Supervised Learning:** In this type of learning, the datasets that the machine utilises to learn are entirely structured and labelled. The datasets have a set of input features and their corresponding outputs and are given to the ML algorithm as inputs for training. While training, the model learns by mapping its predictions to the true predictions present in the dataset, evaluates its performance, and automatically updates its learning curve to achieve better scores of the performance metrics. For example, classification and regression tasks such as image classification and commodity price prediction respectively.

- **Unsupervised Learning:** In this type of learning, the datasets are not mapped with their outputs. The models are trained on a dataset consisting of several features and are expected to produce outputs by learning and identifying common patterns without conforming to a certain "correct answer". For example, clustering algorithms are used for customer segmentation, DNA pattern recognition, and grouping in evolutionary biology.

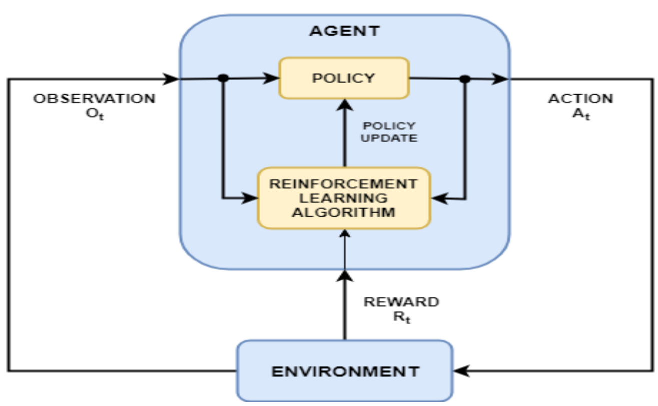
- **Reinforcement Learning:** In this type of learning, the algorithm updates its learning curve by studying the features of the learning problem, environment, and the behavior of the agent in order to maximize its performance. It works on an action-reward-based system by learning any method that is well-suited to solve the problem, while simultaneously correcting any method that does not. The ultimate goal is reached efficiently and rapidly and the machine makes sure to learn only those methods that facilitate this. For example, autonomous driving and gaming technologies.



(a)



(b)



(c)

Figure 1.4.1.: Types of Machine Learning (a) supervised learning (b) unsupervised learningand (c) reinforcement learning

Deep Learning is a subset of machine learning, where the machine-intelligent models are essentially imitations of the human brain. The model, called an Artificial Neural Network, is similar to the human brain where connections exist between the individual units (neurons). However, unlike the human brain, the individual perceptrons are restricted to forming a set number of connections with certain layers and are allowed to propagate data only in restricted directions defined according to the type of neural network.

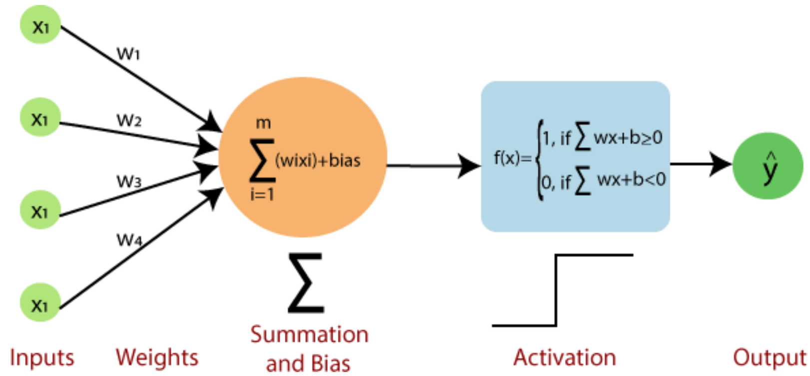
Neural networks can be designed and trained to perform any kind of task that machine learning algorithms can and are more robust than ML models. These networks have the ability to learn from raw data, and grasp the various features as the data is passed on through the different layers while simultaneously compressing the data and producing the output from these sets of features extracted during propagation.

A perceptron, which is an individual unit of a layer of a neural network, is a complex mathematical function that takes a set of inputs, adds weights and biases separately to each input, and passes it through a non-linear function to produce the output. This output is carried tothe perceptron of the next layer through a connection link along with certain other characteristics such as the activation signal, previous internal state, etc. depending on the type of neuralnetwork.

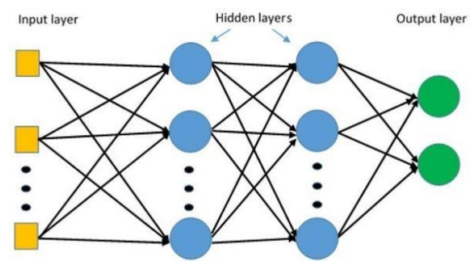
There are two types of perceptrons:

- **Single-layer perceptron:** They can only learn linearly-separable tasks. For example, the linear binary classification problem

- **Multi-layer perceptron:** Also called a fully connected neural network. It contains two ormore layers (input layer, hidden layer(s) and output layer), which helps bring out the non-linearity in the input data, thereby increasing processing power and helping the model learn better. For example, image recognition, and stock analysis.



(a)



(b)

Figure 1.4.2: (a) A single-layer perceptron and (b) a multi-layer perceptron

1.5. Computer Vision

Computer vision is a subset of artificial intelligence which includes the methods designed to enable machines to process visual data such as images, videos, and data from other visual inputsin order to derive meaningful information from them. These methods help machines gather theknowledge and learn the context to tell objects apart, understand the various components and features present in an image, and process sequences of slices of 3D images and videos.

Computer vision uses machine learning and deep learning models such as convolution neural networks (CNNs) to learn, analyze and interpret visual data as humans do. At a basic level, computer vision is mostly pattern recognition. Hence, the most common way to increase the efficiency of a computer vision algorithm is to feed it large amounts of labeled visual data and further use different methods and techniques to help the machine learn these patterns. Computer vision can further be classified into various domains based on its applications (figure 1.5.1.). For example, scene reconstruction, object detection, object recognition, event detection, motion estimation, 3D pose estimation, 3D scene modeling and image restoration to name a few.

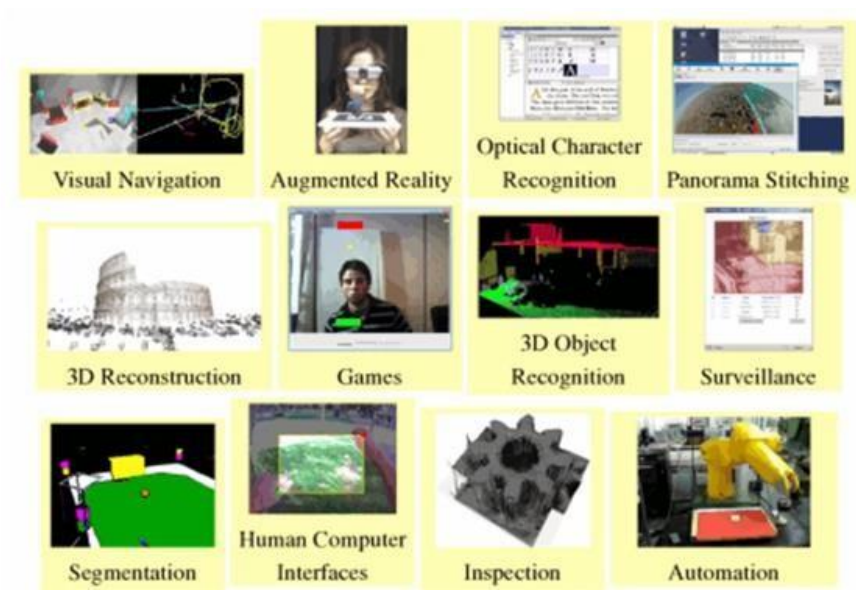


Figure 1.5.1.: Applications of Computer

CHAPTER 2
LITERATURE REVIEW

Lane detection is a well-established area of research in the field of artificial intelligence (AI) and computer vision. Numerous studies and research papers have been published on the topic, focusing on different approaches and techniques for detecting and tracking lanes on the road. Here are some examples of recent literature on lane detection in AI:

- "Lane Detection and Tracking Using Deep Learning Techniques: A Review" by S. R. Sharma and A. K. Chaturvedi (2021): This paper provides a comprehensive review of the different deep learning techniques used for lane detection and tracking in autonomous vehicles. The authors discuss the advantages and limitations of different approaches, including convolutional neural networks (CNNs) and recurrent neural networks (RNNs).

- "A Comprehensive Study of Lane Detection Algorithms in Autonomous Vehicles" by Y. Sun, Z. Chen, and M. Tomizuka (2019): This paper presents a comprehensive study of different lane detection algorithms used in autonomous vehicles, including the Hough Transform, Canny Edge Detection, and Sobel Edge Detection. The authors evaluate the performance of these algorithms in different scenarios and provide recommendations for selecting the most appropriate algorithm for a given application.

- "Real-Time Lane Detection and Tracking for Autonomous Driving Using a Convolutional Neural Network" by A. Li and J. Lu (2018): This paper proposes a real-time lane detection and tracking system for autonomous driving using a CNN. The authors evaluate the performance of the system in different lighting and weather conditions and demonstrate its effectiveness in accurately detecting and tracking lanes on the road.

- "Robust Lane Detection and Tracking in Challenging Scenarios" by T. Chen, S. Shen, and G. Yang (2015): This paper proposes a robust lane detection and tracking algorithm that can handle challenging scenarios such as occlusions, curved lanes, and sharp turns. The authors use a combination of edge detection, colour segmentation, and Hough Transform to detect and track lanes accurately.

Overall, these studies and others demonstrate the importance of lane detection in the development of autonomous vehicles and ADAS systems, as well as the ongoing research efforts to improve the accuracy and reliability of lane detection algorithm

CHAPTER 3
SYSTEM ARCHITECTURE AND DESIGN

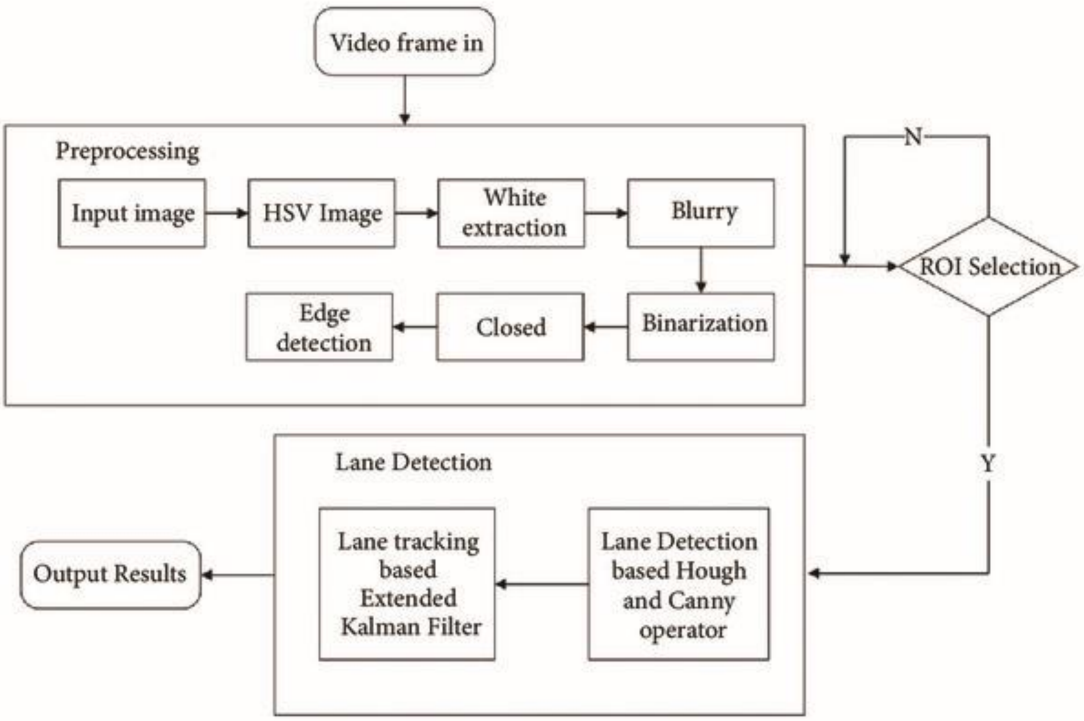


Figure 3.1: System Architecture Diagram

The system architecture for lane detection using OpenCV typically involves several components, including:

Image/Video input: The input component of the system is responsible for capturing the input image or video stream. This can be done using OpenCV's Video Capture class, which can read from a camera or a video file. Once the input is captured, it is passed to the preprocessing component.

Preprocessing: The preprocessing component is responsible for preparing the input image or video stream for processing. This can include resizing the image, converting it to grayscale, and applying filters such as Gaussian blur or Canny edge detection to enhance the edges in the image. OpenCV provides various image processing functions that can be used for preprocessing, such as cv::resize(), cv::cvtColor(), cv::GaussianBlur(), and cv::Canny().

Lane detection: The lane detection component is responsible for detecting the lane markings in the input image or video stream. This can be done using various techniques such as Hough transform, sliding window method, or deep learning-based models. These techniques detect the edges or lines corresponding to the lane markings in the image and return their parameters such as slope and intercept.

Lane tracking: The lane tracking component is responsible for tracking the detected lane markings over time to estimate the position and orientation of the lane. This can be done using techniques such as Kalman filtering, polynomial fitting, or curve fitting. These techniques predict the position and orientation of the lane markings based on their previous positions and velocities.

Output: The output component of the system is responsible for displaying the results of the lane detection and tracking process. This can be done by displaying the input image or video stream with the detected lane markings overlaid on top. The output can also be saved to a file or a database for further analysis.

Overall, the system architecture for lane detection using OpenCV is a combination of several components that work together to detect and track the lane markings in an input image or video stream.

CHAPTER 4
METHODOLOGY

4.1. Dataset

In the lane detection using the Canny operator, we typically use a dataset of images or videos of roads captured by cameras mounted on a vehicle or other devices. The dataset should contain different road scenarios, including straight roads, curved roads, and roads with varying lighting conditions and weather conditions. The images in the dataset should also be labeled to indicate the ground truth information, such as the position and width of the lanes on the road.
One commonly used dataset for lane detection is the TuSimple Lane Detection Challenge dataset, which contains over 6,000 images and videos captured from different angles and under varying conditions. The dataset includes annotations for the position and shape of the lanes, as well as information about the road curvature and driving direction.
Another popular dataset for lane detection is the KITTI Vision Benchmark Suite, which contains a variety of datasets for different computer vision tasks, including lane detection. The lane detection dataset consists of over 289 training images and 290 testing images, along with annotations for the position and shape of the lanes.
Other datasets used for lane detection include the Caltech Lanes dataset, the Penn-Fudan database for Pedestrian Detection and Segmentation, and the Berkeley DeepDrive dataset.

Overall, selecting an appropriate dataset for lane detection using the Canny operator is critical to train and test the accuracy and robustness of the algorithm. It is essential to ensure that the dataset covers a wide range of road scenarios and provides accurate ground truth information to evaluate the performance of the algorithm.

Figure 4.1.1.: Sample Frames from the Dataset

4.2. Feature Extractor

The Canny edge detection operator is a feature extractor that is commonly used in lane detection algorithms. The Canny operator works by detecting the edges in an image, which are the boundaries between areas of different intensity or color. In lane detection, the edges correspond to the lane markings on the road, which can be used to determine the position, width, and direction of the lanes.

The Canny operator uses a multi-stage algorithm that first applies a Gaussian filter to the image to remove noise and then calculates the gradient magnitude and direction of each pixel in the image. Next, non-maximum suppression is applied to thin out the edges, and hysteresis thresholding is used to determine the final edges.

The gradient magnitude and direction of each pixel in the image are important features extracted by the Canny operator, as they provide information about the intensity and direction of the edges in the image. The non-maximum suppression stage is also an important feature extractor, as it helps to thin out the edges and reduce false positives.

In addition to the Canny operator, other feature extractors can be used in lane detection algorithms, depending on the specific requirements and constraints of the application. For example, other edge detection operators, such as Sobel and Laplacian operators, can be used to extract edges in an image. Other feature extraction techniques, such as Hough transform or polynomial fitting, can be used to extract the lane markings from the edges detected by the Canny operator. Overall, the choice of feature extractors in lane detection depends on the specific requirements of the application and the characteristics of the input data.
While the Canny operator is a popular technique for detecting edges in lane detection, there are some potential problems that can arise when using this operator. Here are a few examples:

Sensitivity to Noise: The Canny operator is sensitive to noise in the input image, which can lead to false positive or false negative detections of the lane markings. To mitigate this problem, it is important to apply appropriate preprocessing techniques, such as Gaussian blurring, to reduce the noise in the image.

Thresholding: The Canny operator relies on thresholding to determine the edges in the image. However, choosing appropriate thresholds can be challenging, especially in cases where the lighting conditions or road surfaces change rapidly. To overcome this problem, adaptive thresholding techniques can be used to adjust the thresholds based on the local image characteristics.

Non-Uniform Illumination: The Canny operator can struggle to detect edges in regions with non-uniform illumination, such as shadows or areas of bright sunlight. This problem can be mitigated by applying image normalization or contrast enhancement techniques to improve the overall illumination of the image.

Overlapping Lanes: The Canny operator can struggle to distinguish between overlapping lanes, especially in cases where the lane markings are faded or missing. This problem can be addressed by using additional feature extraction techniques, such as Hough transform or polynomial fitting, to extract the lanes from the edges detected by the Canny operator.

Computational Complexity: The Canny operator can be computationally expensive, especially when processing high-resolution images or videos in real time. This problem can be mitigated by using optimized implementations of the Canny operator, or by using alternative edge detection techniques that are more computationally efficient.

Overall, while the Canny operator is a powerful technique for detecting edges in lane detection, it is important to be aware of these potential problems and to take appropriate measures to address them in order to achieve accurate and robust lane detection results.

CHAPTER 5

CODING AND TESTING

```
import cv2
import numpy as np
# white lines need to find thats the region of interest
def region_of_interest(img, vertices):
    mask = np.zeros_like(img)
    match_mask_color = 255
    cv2.fillPoly(mask, vertices, match_mask_color)
    masked_image = cv2.bitwise_and(img, mask)
    return masked_image
def detect_lane(image):
    gray = cv2.cvtColor(image, cv2.COLOR_RGB2GRAY)
    kernel_size = 5
    blur_gray = cv2.GaussianBlur(gray,(kernel_size, kernel_size), 0)
    low_threshold = 50
    high_threshold = 150
    edges = cv2.Canny(blur_gray, low_threshold, high_threshold)
    imshape = image.shape
    vertices = np.array([[(0,imshape[0]),(450, 320), (490, 320),
(imshape[1],imshape[0])]], dtype=np.int32)
    masked_edges = region_of_interest(edges, vertices)
    rho = 2
    theta = np.pi/180
    threshold = 100
    min_line_length = 40
    max_line_gap = 50
    line_image = np.copy(image)*0
    lines = cv2.HoughLinesP(masked_edges, rho, theta, threshold,
np.array([]), min_line_length, max_line_gap)
    for line in lines:
        for x1,y1,x2,y2 in line:
            cv2.line(line_image,(x1,y1),(x2,y2),(0,0,255),5)
    color_edges = np.dstack((edges, edges, edges))
    lines_edges = cv2.addWeighted(color_edges, 0.8, line_image, 1, 0)
    return lines_edges

# Test the model
cap = cv2.VideoCapture('lanevid.mp4')
while(cap.isOpened()):
    ret, frame = cap.read()
    if ret == True:
        result = detect_lane(frame)
        cv2.imshow('result',result)
        if cv2.waitKey(25) & 0xFF == ord('q'):
            break
    else:
        break

cap.release()
cv2.destroyAllWindows()
```

CHAPTER 6

SCREENSHOTS AND RESULTS

The output of the Canny operator is a binary edge map where the edges are represented as white pixels on a black background. The edges are usually thin and continuous lines with a width of one pixel, which makes them easy to process and analyze.

The Canny operator is widely used for various computer vision applications, such as lane detection in autonomous vehicles, object recognition in images, and boundary detection in medical imaging. The output of the Canny operator can be further processed and analyzed using various techniques, such as edge linking, contour detection, and shape recognition.

Here is an example of the output image of the Canny operator applied to an input image:

Canny Operator Output;

Figure 6.1 : Canny Operator Output

In this example, the Canny operator is used to detect the edges of a hand in an input image. The output image shows the edges of the hand as white lines on a black background, with a few small artifacts and noise removed by the edge thinning and hysteresis thresholding steps.

CHAPTER 7

CONCLUSION AND FUTURE ENHANCEMENTS

Lane detection is a critical component of autonomous vehicles and driver assistance systems. In recent years, there has been significant progress in the development of lane detection algorithms, which use various computer vision techniques to identify lane markings on the road.

In conclusion, lane detection algorithms have shown promising results in detecting and tracking lane markings under different driving conditions. Current lane detection systems rely heavily on deep learning models like convolutional Neural Networks (CNN) and are able to achieve high accuracy under good weather and road conditions. However, there are still some limitations and challenges that need to be addressed in future enhancements.

One of the challenges is dealing with adverse weather conditions such as rain, snow, and fog, which can obscure the lane markings and make it difficult for the algorithm to detect them accurately. Another challenge is dealing with complex road configurations, such as intersections and roundabouts, where multiple lanes merge or diverge.

Lane detection is an important task in autonomous driving and advanced driver assistance system (ADAS). There are several future enhancements that can be made to improve the accuracy and reliability of lane detection. Here are some possibilities:

1. **Multi-modal sensor fusion:** Instead of relying solely on camera images, combining the data from multiple sensors like lidar, radar and GPS can provide more accurate and reliable lane information.

2. **Deep Learning:** Currently, deep learning models like convolutional neural networks (CNN) are used widely for lane detection. Future enhancements can involve exploring more advanced models like Recurrent Neural Networks (RNN) and transformers which can improve the accuracy of lane detection and helps with tasks like predicting lane changes.

3. **Real-time processing:** With the increasing demand for real-time processing in autonomous driving, future enhancements in lane detection can focus on developing algorithms that can process the data in real-time with minimal latency.

4. **Robustness to weather conditions:** Lane detection systems often struggle in adverse weather conditions like rain, snow and fog. Future enhancements can

involve developing algorithms that are more robust to these conditions and can provide accurate lane information even in low visibility conditions.

5. **Integration with other ADAS systems:** Lane detection can be integrated with other ADAS systems like adaptive cruise control and lane departure warning systems to provide a more comprehensive driving experience.

Overall, lane detection algorithms have shown great potential in enhancing the safety and reliability of autonomous vehicles and driver assistance systems. With further research and development, we can expect to see significant advancements in this field in the coming years.

REFERENCES

1. C. Shen, Y. Wang, Y. Wang, and Z. Zhang, "Deep Learning in End-to-End Lane Detection: A Review," IEEE Transactions on Intelligent Transportation Systems, vol. 19, no. 4, pp. 1149-1169, Apr. 2018.

2. H. Lee, S. Kim, and Y. Choi, "Real-Time Lane Detection Using Deep Learning Techniques," IEEE Intelligent Transportation Systems Magazine, vol. 9, no. 2, pp. 64-77, Summer 2017.

3. R. Cucchiara, C. Grana, M. Piccardi, and A. Prati, "Detecting Moving Objects, Ghosts, and Shadows in Video Streams," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 25, no. 10, pp. 1337-1342, Oct. 2003.

4. Y. Luo, J. Ren, W. Zhang, and H. Wang, "Real-Time Lane Detection Using Convolutional Neural Networks," in Proceedings of the IEEE Intelligent Vehicles Symposium (IV), pp. 834-839, June 2018.

5. D. M. Gavrila, "The Visual Analysis of Human Movement: A Survey," Computer Vision and Image Understanding, vol. 73, no. 1, pp. 82-98, Jan. 1999.