A Y DADABHAI TECHNICAL INSTITUTE

DEPARTMENT OF COMPUTER ENGINEERING

A

Internship Project Report On

"OBJECT DETECTION"

SUBMITTED INFULFILLMENT OF THE DEGREE OF DIPLOMA ENGINEERING

GUIDED BY:

ANJALI V.PATEL

PREPARED BY:

MANSURI MO.UMER (226010307052)

2024-2025

A.Y.DADABHAI TECNICAL INSTITUTE,

KOSAMBA-MAHUVEJ ROAD, AT PO. KOSAMBA, TA. MANGROL, DIST.SURAT(GUJRAT)

A Y DADABHAI TECHNICAL INSTITUTE, KOSAMBA

DEPARTMENT OF COMPUTER ENGINEERING

CERTIFICATE

That is to certify that **Mr. MANSURI MO.UMER** Student of Diploma in Computer Engineering (5^{Th} Semester), and Enrollment no. **226010307052** has satisfactorily completed his project entitled **"OBJECT DETECTION"** Using PYTHON OPENCV.

Date: / / 2024

Signature of Guide

Department in-Charge

TABLE OF CONTENTS

ABSTRACT

Our project is to Detect an object with OpenCV- Python . OpenCV has a bunch of pre-trained classifiers that can be used to identify objects such as trees, number plates, faces, eyes, etc. Object detection is a well-known computer technology connected with computer vision and image processing that focuses on detecting objects or its instances of a certain class (such as humans, flowers, animals, cars) in digital images and videos. Object detection can be used for various purposes including retrieval and surveillance. The purpose of "object detection" is to properly locate an object in a photograph and label it with the relevant category. To be more specific, object detection attempts to handle the difficulty of detecting where and what an item is. However, resolving this issue is not simple. A computer, unlike the human eye, analyses pictures in two dimensions. Furthermore, the object's size, direction in space, attitude, and placement in the picture might all be somewhat different.

ACKNOWLEDGEMENT

This project has been the most practical exciting part of my learning experience, which would be an asset for me and for my future career.

No system is created entirely by individual. Many people have helped to create this system and each of their contribution has been valuable. Proper organization of concept and analysis od system is due to keen interest and helping of my teacher and colleagues.

My sincere thanks to our Director **Mr. M. M. Dalchaval** and our principal **Mr. D. V. Patel** molding my through and vision towards all subject.

I am deeply thankful to **Mr. Abdullah M. Hans** Head of Department, who is a constant source of internship during this project.

My most sincere thanks to my project guide **Anjali V.Patel** for his kind of co-operation and who has always been guiding encouraging and motivating me through the project.

Chapter 1: Introduction

1.1 OVERVIEW

INTRODUCTION TO OBJECT DETECTION

Object detection is a fundamental task in computer vision that involves identifying and localizing objects within an image or video. Unlike simple image classification, which assigns a single category to an entire image, object detection identifies multiple objects within an image and determines their positions, usually represented by bounding boxes. This makes object detection particularly useful in applications where understanding the spatial arrangement and interaction of objects is crucial.

1.2 PURPOSE OF OBJECT DETECTION

The primary purpose of object detection is to identify and locate objects within images or videos, enabling machines to understand and interpret visual information similarly to humans. Object detection is a fundamental component in various applications that require the recognition and localization of objects. It serves several key purposes:

1. Automation and Efficiency:

- **Automation of Manual Tasks:** Automating tasks such as quality inspection in manufacturing, inventory management in retail, and anomaly detection in surveillance reduces human labor and increases efficiency.
- **Real-time Decision Making:** In autonomous vehicles and robotic systems, real-time object detection is crucial for making immediate decisions to navigate and interact with the environment safely.

2. Enhancing User Experience:

- **Augmented Reality (AR) and Virtual Reality (VR):** Object detection enhances AR and VR experiences by recognizing and interacting with real-world objects.
- **Smart Assistants and Applications:** Improving functionalities in smart applications, such as photo organization, virtual try-ons, and interactive gaming.

3. Safety and Security:

- **Surveillance Systems:** Detecting and identifying unauthorized access, suspicious activities, or potential threats in real-time to ensure safety and security.
- **Driver Assistance Systems:** Assisting drivers by detecting pedestrians, other vehicles, traffic signs, and obstacles to prevent accidents.

4. Scientific and Medical Advancements:

- **Medical Imaging:** Analyzing medical images to detect and diagnose diseases, tumors, and other abnormalities, leading to better patient outcomes.
- **Environmental Monitoring:** Tracking wildlife, monitoring deforestation, and assessing natural disasters for research and conservation efforts.

1.3 SCOPE OF OBJECT DETECTION

The scope of object detection encompasses various aspects, including the types of objects detected, the contexts in which it is applied, and the technical challenges involved. It extends across multiple domains and continues to evolve with advancements in technology.

1. Types of Objects Detected:

- **Static Objects:** Detecting stationary objects such as buildings, furniture, and static obstacles in images or videos.
- **Dynamic Objects:** Identifying and tracking moving objects like vehicles, people, animals, and drones.
-

2. Application Domains:

- **Automotive Industry:** Object detection in autonomous driving, advanced driver-assistance systems (ADAS), and traffic management.
- **Healthcare:** Detecting medical conditions through imaging techniques like MRI, CT scans, and X-rays.
- **Retail:** Enhancing customer experiences with automated checkout systems, shelf monitoring, and personalized recommendations.
- **Security and Surveillance:** Monitoring public and private spaces to detect suspicious activities, intrusions, and ensuring safety.
- **Robotics:** Enabling robots to perceive and interact with their environment for tasks like picking, placing, and navigation.
- **Entertainment:** Enhancing AR, VR, and gaming experiences by detecting and interacting with real-world objects.

3. Technical Challenges:

- **Accuracy:** Ensuring high detection accuracy in varied and complex environments with diverse object appearances and occlusions.
- **Speed:** Achieving real-time detection performance to meet the demands of applications like autonomous driving and surveillance.
- **Scalability:** Handling large-scale images and videos with multiple objects efficiently.

- **Robustness:** Dealing with challenges like lighting variations, occlusions, and cluttered backgrounds.
- **Generalization:** Ensuring models generalize well across different datasets, environments, and object categories.

1.4 The Process of Object Detection

Description of Key Steps

Object detection involves several key steps, which can be broadly categorized as follows:

1. Data Collection:

Gathering a diverse and representative dataset that includes various objects to be detected. This dataset typically consists of images or videos with labeled bounding boxes for each object.

2. Data Pre-processing:

Preparing the dataset for training by performing tasks such as resizing images, normalizing pixel values, and augmenting data through techniques like thresholding, sub tractor, Erosion, Dilation and contour to increase the variability and robustness of the model.

3. Image Preprocessing:

This initial step involves preparing the image data for analysis. Common preprocessing tasks include resizing images to a standard dimension, normalizing pixel values, and converting color spaces. These steps ensure that the images are in a consistent format suitable for further processing.

4. Object Detection:

Once trained, the model can be used to detect objects in new images or video streams. The model processes the input image and outputs a set of bounding boxes, each with a class label and a confidence score indicating the likelihood of the object being present.

5. Post-Processing:

To improve the accuracy and readability of the results, post-processing techniques such as Non-Maximum Suppression (NMS) are applied. NMS eliminates redundant bounding boxes by keeping only the box with the highest confidence score when multiple boxes overlap significantly.

6. Integration and Application:

The final step involves integrating the object detection system into applications. This could range from real-time video analysis in autonomous vehicles to detecting and counting objects in static images for inventory management.

Object detection systems have revolutionized various fields by providing machines with the ability to interpret and understand visual data similarly to humans. They are widely used in autonomous driving for detecting pedestrians and other vehicles, in surveillance for monitoring activities, in retail for inventory tracking, and in medical imaging for identifying abnormalities.

The implementation of object detection using Python and the OpenCV library combines the simplicity and efficiency of OpenCV's image processing tools with the power of deep learning models.

Chapter 2: Internship Activities

2.1 Background and Context of the Object Detection Project

- **Introduction to Object Detection**:

  Provide an in-depth explanation of what object detection is, including its fundamental principles. Object detection involves identifying and locating objects within an image or video and classifying them into predefined categories. This can be achieved using various techniques and algorithms, which leverage computer vision and machine learning.

- **Significance and Applications**:

  Discuss the significance of object detection in various fields. Examples include:

  - **Automotive Industry**:Used in autonomous driving for detecting pedestrians, vehicles, traffic signs, and other obstacles.
  - **Security and Surveillance**: Enhances security systems by identifying suspicious activities and intrusions.
  - **Healthcare**: Assists in medical imaging to detect anomalies or specific features in radiology.
  - **Retail**: Utilized in inventory management and customer behavior analysis.
  - **Agriculture**: Helps in monitoring crop health and identifying pests.

2.2 Project Objectives

- **Primary Goals**:

  Clearly define the primary goals of the project. These might include:

  - Developing a robust object detection system using the OpenCV library in Python.
  - Achieving high accuracy in detecting and classifying objects in various environments.
  - Evaluating the performance of different object detection models.

- **Secondary Goals**:

  Outline any secondary goals, such as:

  - Enhancing real-time detection capabilities.
  - Reducing computational complexity and improving processing speed.
  - Creating a user-friendly interface for non-technical users.

2.3 Challenges and Solutions

- **Choosing an Appropriate Threshold Value:** One of the main challenges in thresholding is selecting an optimal threshold value. If the value is too high or too low, the segmentation may either miss significant data or include noise.
- Solutions might include:
    - Global Thresholding: Methods like Otsu's algorithm automatically select an optimal threshold by minimizing intra-class variance. It works well for bimodal histograms (where data can be split into two distinct classes).
    - Adaptive Thresholding: For images with varying lighting or intensity across the scene, adaptive thresholding methods (e.g., Gaussian or mean-based) dynamically compute the threshold based on local pixel neighbourhoods.
- **Structuring Element Selection:** Challenge such as, Choosing an inappropriate structuring element (size and shape) can either over-erode or under-erode the objects, leading to loss of essential features or inadequate noise reduction**.**
- Solutions might include:
    - Adaptive structuring element: Select a structuring element based on the scale of objects in the image. For example, if objects have different shapes, consider using different structuring elements (e.g., square, disk, cross) or a combination.
    - Multi-scale erosion: Run erosion at different scales and compare results. This can help in finding the appropriate structuring element size for different objects in the image.
- **Boundary Handling:** Challenge such as, Handling image boundaries can lead to artifacts or errors, such as artificially inflated objects at the edges or boundary loss.
- Solutions could include:
    - Padding Strategies: Use various padding techniques (e.g., zero-padding, wrap-around, or reflection) to properly manage the boundaries.
    - Border-aware Kernels: Implement kernels that are aware of the boundary, so they adapt dynamically to avoid introducing artifacts.
- **Moving Background:** Challenge such as, Things like trees blowing in the wind or water moving might get confused as "moving objects."
- Solutions could include:
    - Use techniques that can tell the difference between small movements (like leaves) and actual moving objects.
- **Lighting Changes:** Bright or dark conditions can make the counter hard to see.
- Solutions could include:
    - Use techniques to adjust and balance brightness and contrast, so the counter stands out better.

2.4 WHAT IS AN IMAGE?

A picture is spoken to as a two dimensional capacity f(x, y) where x and y are spatial co-ordinates and the adequacy of "T" at any match of directions (x, y) is known as the power of the picture by then.

2.5 DIGITAL IMAGE PROCESSING

Computerized picture preparing is a range portrayed by the requirement for broad test work to build up the practicality of proposed answers for a given issue. A critical trademark hidden the plan of picture preparing frameworks is the huge level of testing and experimentation that Typically is required before touching base at a satisfactory arrangement. This trademark informs that the capacity to plan approaches and rapidly model hopeful arrangements by and large assumes a noteworthy part in diminishing the cost and time required to land at a suitable framework execution.

Fig. 1.1 digital image

2.5.1 Processing on image:

Processing on image can be of three types They are low-level, mid-level, high level.

1. Low-level Processing:

- Preprocessing to remove noise.
- Contrast enhancement.
- Image sharpening.

2. Medium Level Processing:

- Segmentation.
- Edge detection

- Object extraction.

3. High Level Processing:

- Image analysis
- Scene interpretation

2.5.2 Why Image Processing?

Since the digital image is invisible, it must be prepared for viewing on one or more output device(laser printer, monitor at).The digital image can be optimized for the application by enhancing the appearance of the structures within it.

There are three of image processing used. They are

- Image to Image transformation
- Image to Information transformations
- Information to Image transformations

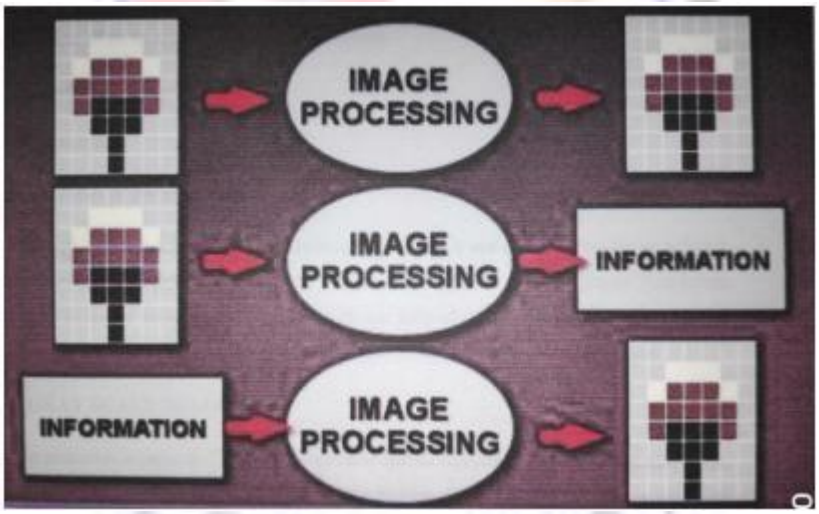


Fig. 1.2 Types of Image Processing

Pixel:

Pixel is the smallest element of an image. Each pixel correspond to any one value. In an 8-bit gray scale image, the value of the pixel between 0 and 255.Each pixel store a value proportional to the light intensity at that particular location. It is indicated in either Pixels per inch or Dots per inch.

Resolution:

The resolution can be defined in many ways. Such as pixel resolution, spatial resolution, temporal resolution, spectral resolution.In pixel resolution, the term resolution refers to the total number of count of pixels in an digital image. For example, If an image has M rows and N columns, then its resolution can be defined as MX N. Higher is the pixel resolution, the higher is the quality of the image.

Resolution of an image is of generally two types.

• Low Resolution image

• High Resolution

Since high resolution is not a cost effective process It is not always possible to achieve high resolution images with low cost. Hence it is desirable Imaging. In Super Resolution imaging, with the help of certain methods and algorithms we can be able to produce high resolution images from the low resolution image from the low resolution images.

2.5.3 GRAY SCALE IMAGE

A gray scale picture is a capacity I (xylem) of the two spatial directions of the picture plane. I(x,y) is the force of the picture force of picture at the point (x, y) on the picture plane. I (xylem) take nonnegative expect the picture is limited by a rectangle.

COLOR IMAGE

It can be spoken to by three capacities, R (xylem) for red, G (xylem) for green and B (xylem) for blue. A picture might be persistent as for the x and y facilitates and furthermore in adequacy. Changing over such a picture to advanced shape requires that the directions and the adequacy to be digitized. Digitizing the facilitates esteems is called inspecting. Digitizing the adequacy esteems is called quantization.

2.6 RELATED FUNCTION:

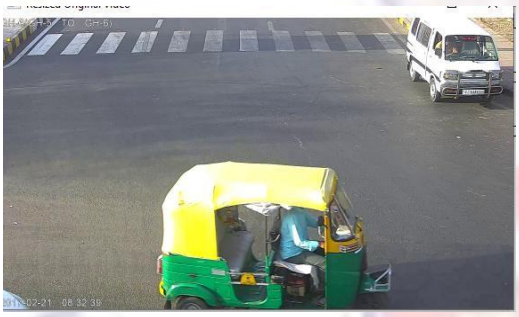
Certainly! Here is an overview of the key image processing techniques in OpenCV you mentioned: thresholding, background subtraction, erosion, dilation, and contour detection, with an explanation of how each works and their applications in a deep learning project report.

1. Thresholding

Thresholding is a technique used to segment an image by setting all pixel values above a certain threshold to one value (e.g., white) and all pixel values below that threshold to another value (e.g., black). It's commonly used for binary segmentation.

EXAMPLE SIMPLE THRESHOLDING:

INPUT:



OUTPUT:

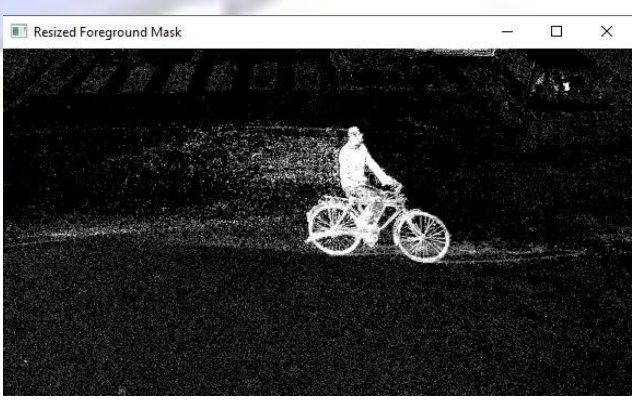
2. Background Subtraction

Background subtraction is used to separate foreground objects from the background. It's commonly used in video processing for motion detection.

EXAMPLE USING MOG2:

INPUT:

OUTPUT:

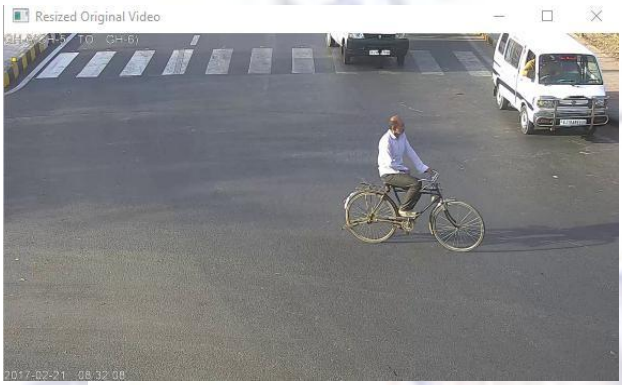


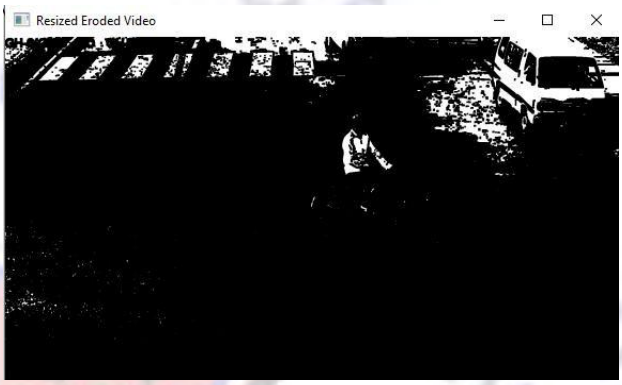
3.Erosion

Erosion is a morphological operation that erodes away the boundaries of the foreground object. It is used to remove small white noises, detach two connected objects, etc.

EXAMPLE USING EROSION:

INPUT: **OUTPUT:**



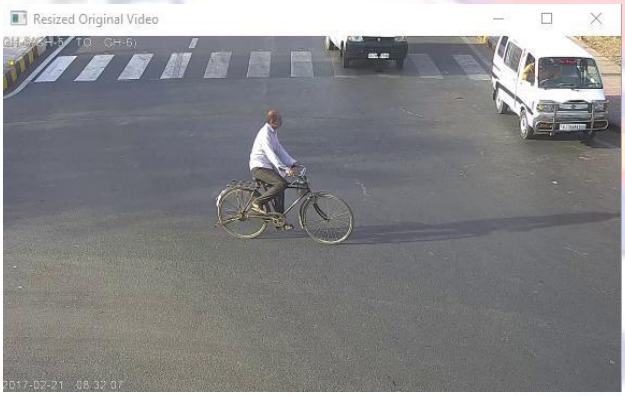


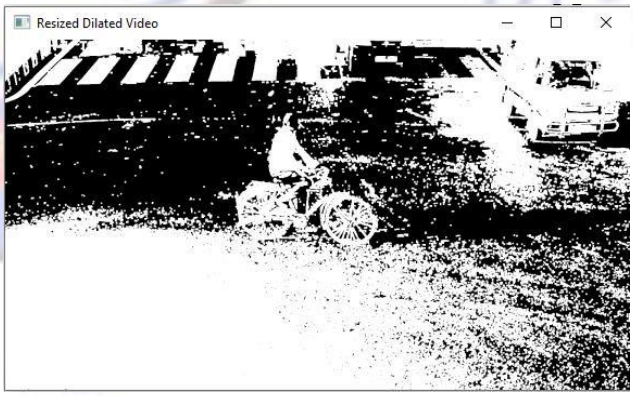
4.Dilation

Dilation is the opposite of erosion. It increases the object area and is used to connect broken parts of an object.

EXAMPLE USING DILATION:

INPUT: **OUTPUT:**



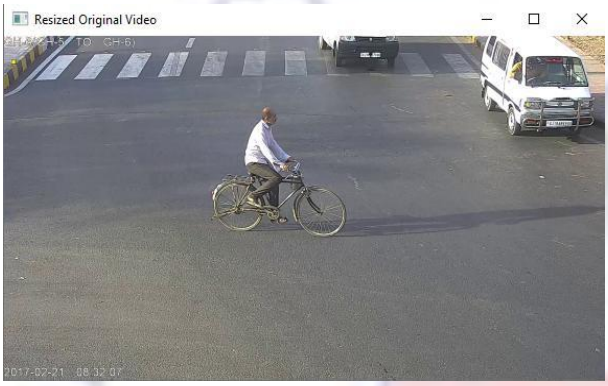


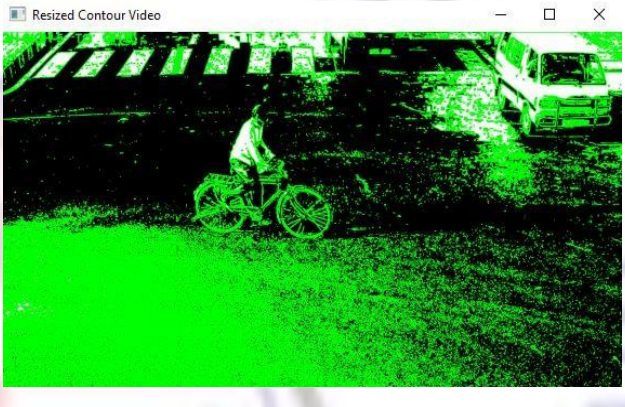
5.Contour Detection

Contour detection is used to detect the shapes of objects in an image. It is useful in shape analysis and object detection and recognition.

EXAMPLE USING CONTOUR DETECTION:

INPUT: **OUTPUT:**





2.7 APPLICATION OF OBJECT DETECTION

The major applications of Object Detection are:

1. FACIAL RECOGNITION

"Deep Face" is a deep learning facial recognition system developed to identify human faces in a digital image. Designed and developed by a group of researchers in Facebook. Google also has its own facial recognition system in Google Photos, which automatically separates all the photos according to the person in the image.

There are various components involved in Facial Recognition or authors could say it focuses on various aspects like the eyes, nose, mouth and the eyebrows for recognizing a faces.

2. PEOPLE COUNTING

People counting is also a part of object detection which can be used for various purposes like finding person or a criminal; it is used for analyzing store performance or statistics of crowd during festivals. This process is considered a difficult one as people move out of the frame quickly.

3. INDUSTRIAL QUALITY CHECK

Object detection also plays an important role in industrial processes to identify or recognize products. Finding a particular object through visual examination could be a basic task that's involved in

multiple industrial processes like sorting, inventory management, machining, quality management, packaging and so on. Inventory management can be terribly tough as things are hard to trace in real time. Automatic object counting and localization permits improving inventory accuracy.

4. SELF DRIVING CARS

Self-driving is the future most promising technology to be used, but the working behind can be very complex as it combines a variety of techniques to perceive their surroundings, including radar, laser light, GPS, odometer, and computer vision. Advanced control systems interpret sensory info to allow navigation methods to work, as well as obstacles and it. This is a big step towards Driverless cars as it happens at very fast speed.

5. SECURITY

Object Detection plays a vital role in the field of Security; it takes part in major fields such as face ID of Apple or the retina scan used in all the sci-fi movies. Government also widely use this application to access the security feed and match it with their existing database to find any criminals or to detecting objects like car number involved in criminal activities. The applications are limitless.

Chapter 3: Technical Skills and Knowledge Acquired

3.1 Overview

Python is a versatile programming language known for its simplicity and readability, making it popular for both beginners and professionals. Its extensive libraries and frameworks support various domains such as web development, data analysis, artificial intelligence, and computer vision.

3.2 Introduction Of Opencv (Open Source Computer Vision Library):

is an open-source computer vision and machine learning software library. It contains more than 2500 optimized algorithms used for various tasks such as image processing, video capture, and analysis. OpenCV is widely used for real-time computer vision applications.

3.2.1 OpenCV's application areas include:

1. 2D and 3D feature toolkits
2. Egomotion estimation
3. Facial recognition system
4. Gesture recognition
5. Human–computer interaction (HCI)
6. Mobile robotics
7. Motion understanding
8. Object identification
9. Segmentation and recognition

3.2.2 Key Concepts in OpenCV:

1. **Image Processing**: Fundamental operations like reading, writing, and displaying images, along with basic manipulations such as resizing, rotating, and cropping.
2. **Video Processing**: Capture and process video from files or cameras.
3. **Drawing Functions**: Drawing shapes like lines, circles, rectangles, and text on images.
4. **Geometric Transformations**: Transformations such as scaling, translation, rotation, and perspective transformation.
5. **Color Space Conversions**: Converting images between different color spaces, e.g., BGR to grayscale or HSV.
6. **Image Thresholding**: Segmenting images by setting a threshold to separate pixels based on intensity.
7. **Contour Detection**: Identifying and processing contours in an image.
8. **Feature Detection**: Detecting features like edges, corners, and key points in an image.
9. **Object Detection**: Identifying and locating objects within an image using various techniques, including deep learning models.

3.2.3 Advantages of Opencv:

1. Open Source and Free:
- OpenCV is free to use and open source, meaning developers can use it without any cost and can contribute to its development.

2. Cross-Platform Compatibility:
- OpenCV supports multiple operating systems including Windows, macOS, Linux, Android, and iOS, making it versatile for different development environments.

3. Extensive Library:
- It offers a wide range of functions for computer vision tasks such as image processing, object detection, face recognition, and machine learning.

4. Fast Processing:
- OpenCV is optimized for real-time applications and can process images and videos quickly, which is essential for tasks like real-time video analysis.

5. Interoperability with Other Libraries:
- It can easily be integrated with other libraries and frameworks such as NumPy which are commonly used in machine learning and deep learning projects.

3.2.4 Disadvantages of OpenCV:

1. Steep Learning Curve:
- While it is powerful, OpenCV can have a steep learning curve, especially for beginners who are not familiar with computer vision concepts.

2. Limited High-Level Abstractions:
- OpenCV provides low-level functions, meaning developers often need to write more code for complex tasks compared to higher-level frameworks or libraries.

3. Not Tailored for Deep Learning:
- While OpenCV supports some deep learning functions, it is not as specialized or optimized for deep learning as other libraries like TensorFlow or PyTorch.

4. Complexity in Setting Up:
- Installing and setting up OpenCV, especially with dependencies like CUDA or integrating with other languages (e.g., Python bindings), can be challenging.

5. Lack of Support for Some Modern Algorithms:
- OpenCV may not have the latest state-of-the-art algorithms implemented as quickly as other libraries focused solely on deep learning or specific domains.

3.3 Introduction To Numpy (Numerical Python Library):

NumPy is a fundamental package for scientific computing in Python. It provides support for large, multi-dimensional arrays and matrices, along with a collection of mathematical functions to operate on these arrays. In computer vision, NumPy is essential for handling image data, which is often represented as multi-dimensional arrays.

3.3.1 Numpy's application areas include:

1. Array Creation
2. Mathematical Operations
3. Linear Algebra
4. Random Number Generation
5. Statistical Analysis
6. Matrix Operations
7. Sorting and Searching
8. Broadcasting
9. Data I/O (Input/Output)
10. Image Processing (with libraries like OpenCV)

3.3.2 Key Concepts in NumPy:

1. **Ndarray (N-dimensional array)**: The core data structure in NumPy, ndarray is a multi-dimensional array that holds elements of the same data type. It supports fast and efficient operations on large datasets.
2. **Array Creation**: NumPy provides functions like np.array(), np.zeros(), np.ones(), np.arange(),and np.linspace() for creating arrays with specific values or patterns.
3. **Broadcasting**: A powerful feature that allows operations on arrays of different shapes, making it possible to perform element-wise operations without explicitly resizing arrays.
4. **Vectorization**: NumPy allows for the application of operations to entire arrays without the need for explicit loops, which leads to more efficient and readable code.
5. **Indexing and Slicing**: NumPy supports slicing and indexing of arrays to access subsets of data. Advanced indexing techniques like 16oolean indexing and fancy indexing are also supported.

3.3.3 Advantages of NumPy:

1. Performance:

- NumPy operations are implemented in C, which makes them much faster than equivalent Python operations, especially for large datasets.

2. Memory Efficiency:

- NumPy arrays consume less memory compared to Python lists for storing the same amount of data, as they store elements of the same data type contiguously in memory.

3. Broad Functionality:

- NumPy provides a wide range of mathematical, logical, and statistical operations, along with linear algebra and random number generation, making it a comprehensive tool for numerical computing.

4. Interoperability:

- NumPy arrays can be easily integrated with other libraries such as Pandas, SciPy, Matplotlib,openCV and machine learning frameworks like TensorFlow and PyTorch.

5. Ease of Use:

- NumPy's syntax is intuitive and similar to standard Python operations, making it easy to learn and use for those familiar with Python.

3.3.4 Disadvantages of NumPy:

1. Steep Learning Curve for Beginners:

- While NumPy is powerful, beginners might find it challenging to grasp concepts like broadcasting, vectorization, and multidimensional arrays.

2. Less Flexibility with Data Types:

- NumPy arrays require uniform data types, which can be limiting compared to Python lists that can hold different data types in a single structure.

3. Memory Consumption:

- For very large datasets, even though NumPy is more memory-efficient than lists, it can still consume a significant amount of memory, especially when working with high-dimensional arrays.

4. Complexity with Large-Scale Data:

- NumPy is not optimized for distributed computing or handling extremely large datasets that don't fit into memory, unlike libraries such as Dask or PySpark.

Chapter 4: Analysis and Findings

4.1 Experimental Result of object detecting:

INPUT:

```
import cv2
import numpy as np

# Function to rescale frames
def rescaleframe(frame, scale=0.5):
    width = int(frame.shape[1] * scale)
    height = int(frame.shape[0] * scale)
    dimensions = (width, height)
    return cv2.resize(frame, dimensions, interpolation=cv2.INTER_AREA)

# Reading the video
capture = cv2.VideoCapture('ambulance.mkv')

# Create a background subtractor object with adjusted parameters
backgroundObject = cv2.createBackgroundSubtractorMOG2(history=500, varThreshold=30,
detectShadows=False)
kernel = np.ones((3, 3), np.uint8)
kernel2 = np.ones((5, 5), np.uint8)  # Slightly larger kernel for dilation

while True:
    ret, frame = capture.read()
    if not ret:
        break

    # Apply the background subtractor
    fgmask = backgroundObject.apply(frame)

    # Threshold and clean up the mask
    __, fgmask = cv2.threshold(fgmask, 25, 255, cv2.THRESH_BINARY) # Slightly higher
threshold
    fgmask = cv2.erode(fgmask, kernel, iterations=1)  # Less aggressive erosion
    fgmask = cv2.dilate(fgmask, kernel2, iterations=4)  # Adjust dilation to match eroded size

    # Detect contours
    contours,___= cv2.findContours(fgmask, cv2.RETR_EXTERNAL,
cv2.CHAIN_APPROX_SIMPLE)

    frameCopy = frame.copy()

    # Loop through contours and detect moving objects (cars)
    for cnt in contours:
```

```
        if cv2.contourArea(cnt) > 5000:  # Lower contour area threshold
            x, y, width, height = cv2.boundingRect(cnt)
            # Draw rectangle around the detected object
            cv2.rectangle(frameCopy, (x, y), (x + width, y + height), (0, 255, 0), thickness=2)
            # Add label to the detected object
            cv2.putText(frameCopy, "OBJECT DETECTED", (x, y - 10),
cv2.FONT_HERSHEY_SIMPLEX, 0.6, (0, 255, 0), 2, cv2.LINE_AA)

    # Foreground mask application
    foreground = cv2.bitwise_and(frame, frame, mask=fgmask)

    # Resize frames for display (optional, for better visibility)
    resized_frame = rescaleframe(frame, scale=0.6)
    resized_foreground = rescaleframe(foreground, scale=0.6)
    resized_frameCopy = rescaleframe(frameCopy, scale=0.6)
    resized_fgmask = rescaleframe(fgmask, scale=0.6)

    # Display frames
    cv2.imshow('Foreground', resized_foreground)
    cv2.imshow('Detected Cars', resized_frameCopy)
    cv2.imshow('FG Mask', resized_fgmask)
    cv2.imshow('Original Frame', resized_frame)

    # Exit on 'q' key
    if cv2.waitKey(1) == ord('q'):
        break

# Release video capture and close all windows
capture.release()
cv2.destroyAllWindows()
```

OUTPUT:

Figure of Original frame:

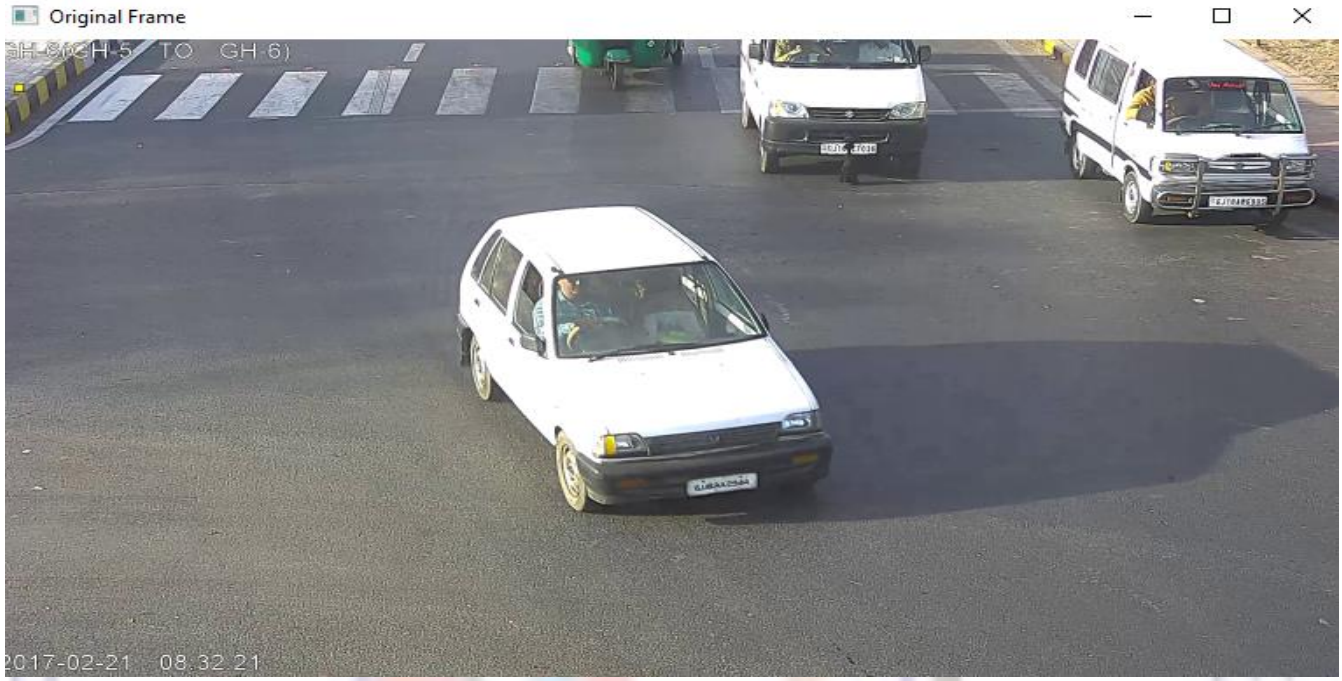
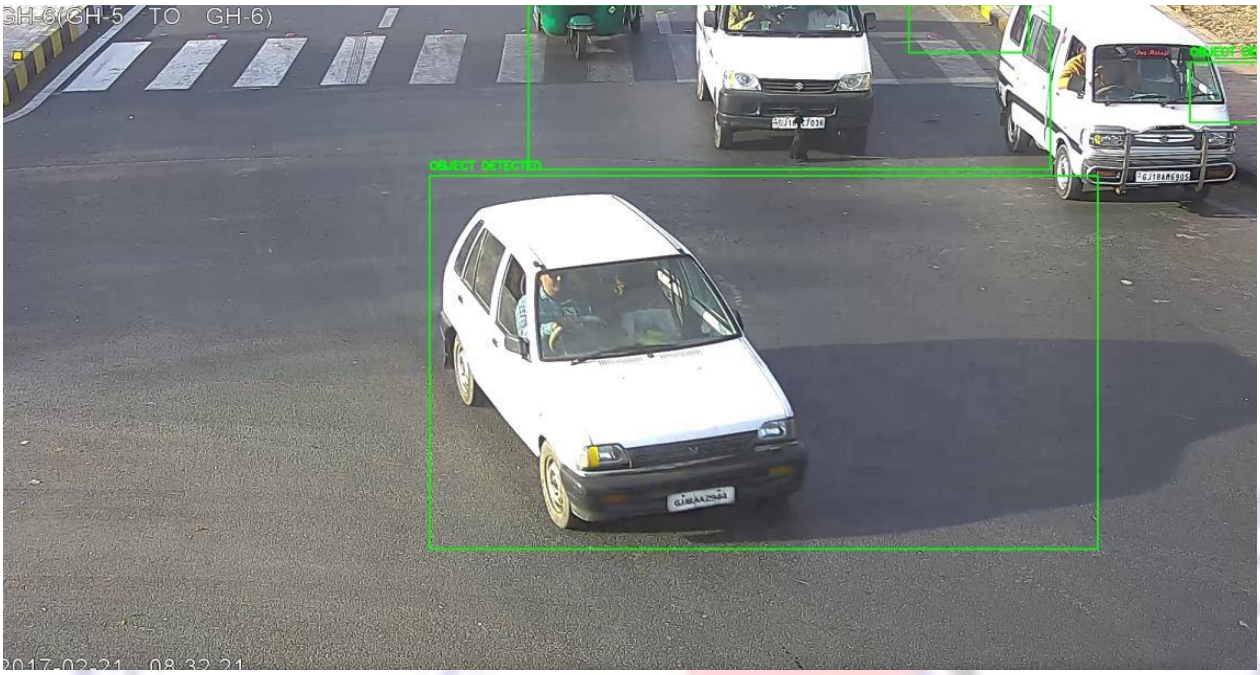
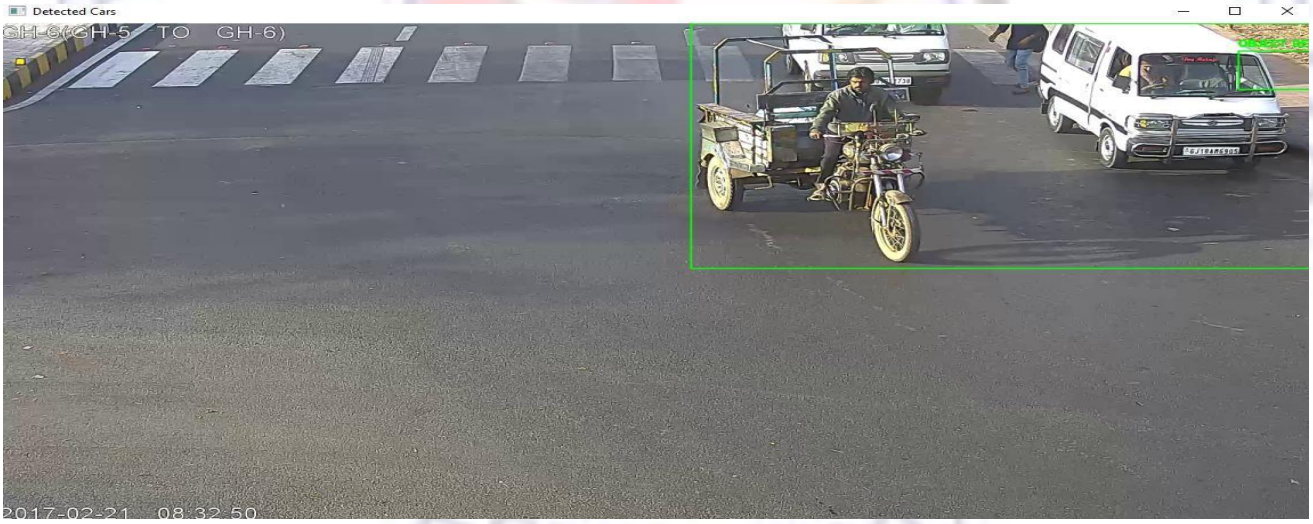


Figure of detected frame:





Chapter 5: Conclusion

The proposed system effectively uses image processing techniques for accurate vehicle detection on the road. Techniques such as thresholding, background subtraction, erosion, and dilation are applied to refine the detection process and isolate vehicle contours. By identifying connected components, the system can reliably detect and count vehicles as they enter and exit the monitored area. This approach demonstrates the effectiveness of image processing for vehicle detection in traffic monitoring applications.

References

- List all the references used in the report, including websites, books, and articles.

[1] Ganesan, V., Devi, V. A., & Yellamma, P. (2022). Object Detection using PYTHON Programming. Journal of Next Generation Technology (ISSN: 2583-021X), 2(1).

[2] Sharma, A., Pathak, J., Prakash, M., & Singh, J. N. (2021, December). Object detection using OpenCV and python. In 2021 3rd international conference on advances in computing, communicationcontrol and networking (ICAC3N) (pp. 501-505). IEEE.

[3] "Introduction to object detection" ,Available at:

https://www.geeksforgeeks.org/opencv-python-

tutorial/

[4] "tutorial of object detection" ,Available at:

https://youtu.be/oXlwWbU8l2o?si=PDhdeqCzorR88E

g0

[5] Jalled, F., & Voronkov, I. (2016). Object detection using image processing. arXiv preprint arXiv:1611.07791.

[6] Joshi, P. (2015). OpenCV with Python by example. Packt Publishing Ltd.

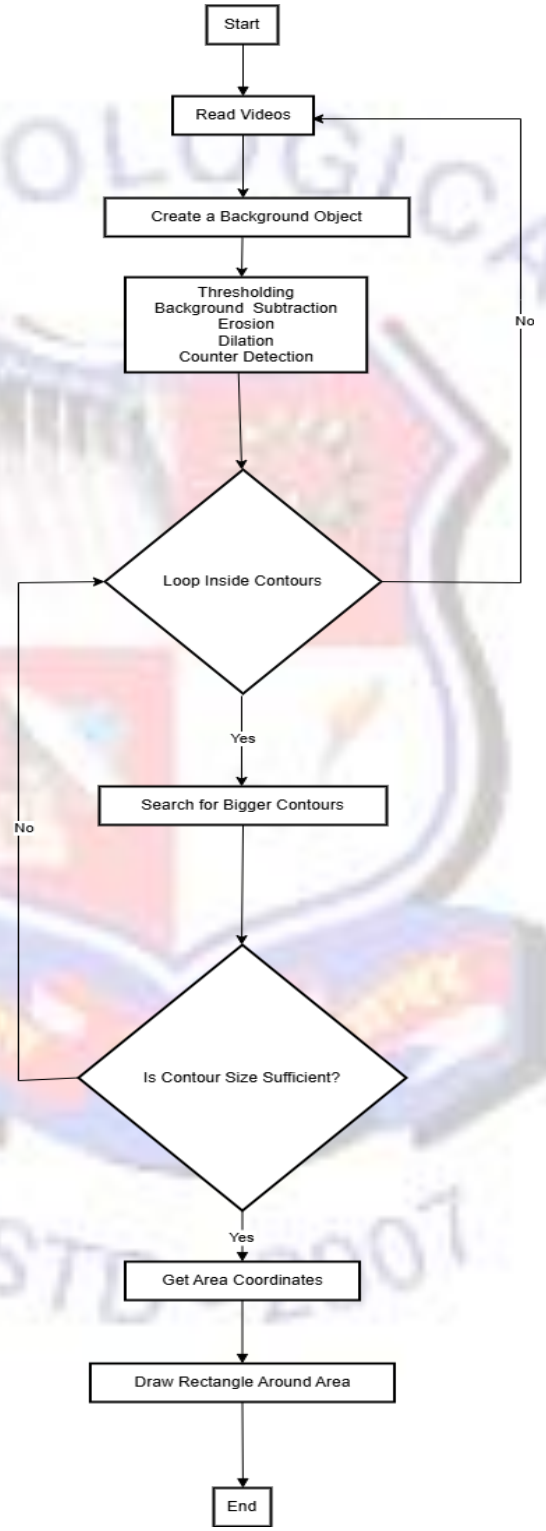
[7] Mordvintsev, A., & Abid, K. (2014). Opencv-python tutorials documentation. Obtenido de https://media. readthedocs. org/pdf/opencv-python-tutroals/latest/opencv-python-tutroals. pdf.

[8] "simplification of object detection", Avaliable

at:http://www.kaggle.com/

Appendices

- **Flow chart for object detection:**




GUJARAT TECHNOLOGICAL UNIVERSITY
ESTD - 2007