SVKM'S NMIMS Deemed to be UNIVERSITY | HYDERABAD

School of Technology Management and Engineering

A Project Report On

Automatic Traffic Signal Indicator

Submitted By

Malde Saicharan – 70572200033

Submitted To

Prof. Wasiha Tasneem

Submission Date

15 November ,2024

CERTIFICATE OF ACCEPTANCE

The report of the Project titled “Automatic Traffic Signal Indicator” submitted by Saicharan(L037) of CSE 4th Semester of 2024) is hereby recommended to be accepted for fulfilment of the semester 4.

Signature with date

Automatic Traffic Signal Indicator

Contents

Problem Statement

Manual traffic lights play a crucial role in regulating vehicular and pedestrian movement at intersections, ensuring smooth traffic flow and enhancing road safety. These traffic control systems require diligent monitoring and adjustment by operators to accommodate fluctuating traffic patterns, pedestrian crossings, and emergency situations effectively.

Manual traffic light operators are tasked with the responsibility of manually adjusting signal timings, managing priority lanes, and responding to real-time traffic conditions to optimize traffic flow and minimize congestion. Additionally, they must possess a keen understanding of traffic regulations and safety protocols to ensure compliance and mitigate the risk of accidents or gridlock.

Despite the advent of automated traffic management systems, manual traffic lights remain essential in areas with limited infrastructure or during emergencies when manual intervention is necessary. Thus, the role of manual traffic light operators continues to be indispensable in maintaining order and efficiency on the roads.

Abstract

In this project, we propose the development of an automatic traffic light indicator system utilizing machine learning techniques. The objective is to create a smart traffic control system capable of dynamically adjusting signal timings based on real-time traffic conditions. The system will leverage machine learning algorithms to analyse traffic patterns, vehicle density, and pedestrian activity at intersections. By collecting data from various sensors such as cameras, radar, and lidar, the system will continuously monitor traffic flow and optimize signal timings to minimize congestion and reduce waiting times.

Additionally, the proposed system will incorporate predictive modelling to anticipate traffic surges and pre-emptively adjust signal priorities to ensure efficient traffic management. Through this research, we aim to contribute to the advancement of intelligent transportation systems, improving road safety, and enhancing the overall efficiency of urban traffic networks.

Introduction

The field of transportation engineering has witnessed significant advancements in recent years, driven by the rapid evolution of technology and the growing need for efficient urban mobility solutions. One critical aspect of urban transportation management is the control of traffic signals at intersections, which plays a pivotal role in regulating vehicle flow and ensuring pedestrian safety. Traditional traffic light systems operate on fixed schedules or pre-defined signal plans, often leading to inefficiencies and congestion during peak hours. To address these challenges, there is a pressing demand for intelligent traffic management systems capable of dynamically adapting signal timings based on real-time traffic conditions.

In response to this need, our research focuses on the development of an automatic traffic light indicator system empowered by machine learning techniques. By harnessing the power of data analytics and predictive modeling, our system aims to revolutionize traffic control mechanisms, optimize signal timings, and enhance the overall efficiency of urban transportation networks. This introduction sets the stage for our exploration into the design, implementation, and evaluation of this innovative traffic management solution

Literature Review

Automatic traffic signal indicator systems utilizing live video capture have garnered significant attention in recent years due to their potential to improve traffic management and road safety. Various research efforts have explored different approaches and methodologies in this domain.

One common approach involves the use of computer vision techniques to analyze live video feeds from traffic cameras or sensors installed at intersections. Researchers have employed algorithms such as object detection, image segmentation, and optical flow analysis to detect vehicles, pedestrians, and other relevant objects in the scene. Additionally, machine learning models, including convolutional neural networks (CNNs) and deep learning architectures, have been utilized to classify and track objects in real-time.

Several studies have focused on developing intelligent algorithms capable of predicting traffic flow patterns and identifying congestion or abnormal traffic conditions. These algorithms often leverage historical traffic data, weather information, and other contextual factors to improve accuracy and reliability. Additionally, researchers have explored the integration of reinforcement learning techniques to optimize traffic signal timings dynamically based on real-time traffic conditions.

Despite the advancements in automatic traffic signal indicator systems, several limitations and challenges persist. One major challenge is the accurate detection and classification of objects in complex urban environments with varying lighting conditions, occlusions, and occlusions. Additionally, the real-time processing requirements of video data pose computational challenges, necessitating efficient algorithms and hardware acceleration techniques.

Moreover, the scalability and deployment of these systems across a large number of intersections pose logistical and infrastructural challenges. Ensuring robustness and reliability in diverse environmental conditions and addressing privacy concerns related to the collection and processing of live video data are also areas of ongoing research.

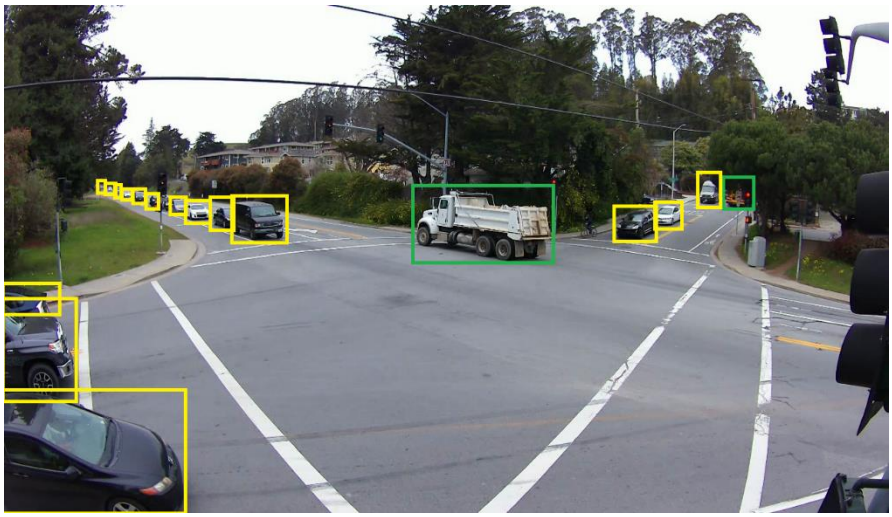
The project aims to address some of these limitations by developing a robust and efficient automatic traffic signal indicator system using live video capture. By leveraging state-of-the-art computer vision and machine learning techniques, the system aims to accurately detect, track, and classify vehicles and pedestrians in real-time, enabling dynamic and adaptive traffic signal control strategies.

Data Collection and Preprocessing:

In this project on automatic traffic signal indicator using live video capture, the primary data source consists of live video feeds obtained from traffic cameras installed at intersections. These video feeds capture real-time traffic scenes, including vehicles, pedestrians, and other objects moving within the camera's field of view. Additionally, ancillary data such as timestamp information and camera metadata may also be collected to augment the analysis.

The data preprocessing pipeline involves several key steps to prepare the raw video data for subsequent analysis and model training. Firstly, the raw video streams are subjected to frame extraction, where individual frames are extracted at regular intervals (e.g., every second) to create a sequence of images representing the traffic scene over time.

Next, each frame undergoes cleaning and preprocessing to enhance image quality and remove noise or artifacts that could affect object detection and tracking algorithms. Techniques such as noise reduction filters, contrast enhancement, and image stabilization may be applied to improve the clarity and consistency of the frames.



Following cleaning, the preprocessed frames are passed through object detection and tracking algorithms to identify and localize relevant objects in the traffic scene, such as vehicles, pedestrians, and traffic signs. These algorithms leverage computer vision techniques, including convolutional neural networks (CNNs) and feature-based tracking algorithms, to detect and track objects of interest across consecutive frames.

In addition to object detection, feature engineering may be performed to extract relevant attributes or characteristics from the detected objects, such as size, speed, direction of motion, and proximity to other objects. These features provide valuable contextual information for subsequent analysis and decision-making.

Finally, the preprocessed data undergoes normalization and scaling to ensure consistency and compatibility across different datasets and models. Normalization techniques such as min-max scaling or z-score normalization are applied to rescale the feature values to a common range, reducing the impact of outliers and improving model convergence during training.

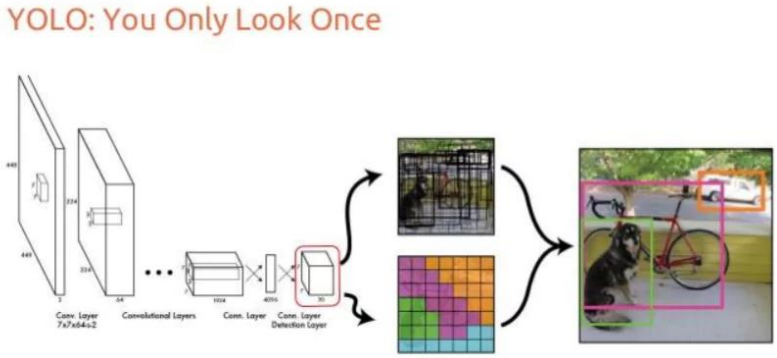
By meticulously collecting and preprocessing live video data from traffic cameras and applying robust cleaning and feature engineering techniques, the project aims to create a high-quality dataset suitable for training and evaluating machine learning models for automatic traffic signal indication.

Methodology:

For the automatic traffic signal indicator using live video capture, the project employs a combination of computer vision and machine learning techniques. Initially, object detection algorithms, such as YOLO (You Only Look Once) or SSD (Single Shot MultiBox Detector), are utilized to detect vehicles, pedestrians, and other relevant objects in the traffic scene. These algorithms offer real-time performance and accurate object localization, making them suitable for processing live video feeds.

Following object detection, tracking algorithms like SORT (Simple Online and Realtime Tracking) or Kalman filters are applied to maintain consistent object identities across consecutive frames. This ensures smooth tracking of vehicles and pedestrians as they move within the scene.

Additionally, deep learning models, particularly convolutional neural networks (CNNs), are leveraged for further analysis and decision-making. CNNs are trained to recognize traffic signs, lane markings, and other important visual cues in the traffic environment, enabling the system to make informed decisions regarding traffic signal indication



Experimental Setup:

1. **Data Collection**: First, we collect a large dataset of traffic videos from various sources, such as traffic cameras, dashcams, or publicly available datasets like Cityscapes, KITTI, or Udacity's self-driving car dataset. Each video contains footage of vehicles moving in different traffic conditions.
2. **Temporal Splitting**: We decide to split the dataset temporally, meaning we'll use the first 80% of frames from each video for training and the remaining 20% for testing. Temporal splitting ensures that the model is exposed to a diverse range of traffic scenarios during training and can generalize well to unseen temporal variations.
3. **Data Preprocessing**: Before splitting the dataset, we preprocess the videos to extract individual frames. We may also resize the frames to a consistent resolution, apply normalization to standardize pixel values, and perform any necessary augmentation techniques like rotation, flipping, or brightness adjustments to increase the dataset's diversity.
4. **Dataset Splitting**: For each video in the dataset, we split the frames into two segments: one segment containing the first 80% of frames for training and another segment containing the remaining 20% of frames for testing. We ensure that the split is done randomly within each video to prevent any bias introduced by the ordering of frames.
5. **Stratified Sampling (Optional)**: If our dataset contains videos captured in different cities or under different weather conditions, we may choose to perform stratified splitting to ensure that the distribution of cities or weather conditions is similar in both the training and testing sets. This helps prevent the model from learning biases associated with specific locations or environmental factors.

6. **Model Training and Evaluation**: We train our vehicle detection and classification model using the frames from the training set. After training, we evaluate the model's performance using the frames from the testing set. We calculate evaluation metrics such as accuracy, precision, recall, and F1 score to assess the model's ability to correctly detect and classify vehicles in unseen traffic scenarios.

Programs:

```
import cv2

cascade_src = r'C:\Users\itssp\Desktop\NMIMS\SEMESTER IV\Machine Learning\assignments\traffic_signal\vehicle_detection_haarcascades-master\vehicle_detection_haarcascades-master\cars.xml'

video_src = r' C:\Users\itssp\Desktop\NMIMS\SEMESTER IV\Machine Learning\assignments\traffic_signal\vehicle_detection_haarcascades-master\vehicle_detection_haarcascades-master\WhatsApp Video 2024-03-07 at 15.00.06_2ddb1d8e.mp4'

cap = cv2.VideoCapture(video_src)

car_cascade = cv2.CascadeClassifier(cascade_src)

# Variable to store the total count of cars detected

total_cars_detected = 0

while True:

    ret, img = cap.read()

    if not ret:

        break

    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)

    # Detect cars in the current frame

    cars = car_cascade.detectMultiScale(gray, 1.1, 1)

    # Add the count of cars detected in the current frame to the total count

    total_cars_detected += len(cars)

    for (x, y, w, h) in cars:

        cv2.rectangle(img, (x, y), (x + w, y + h), (0, 0, 255), 2)

    # Display the frame with detected cars
```

```
    cv2.imshow('video', img)
    if cv2.waitKey(33) == 27:
        # Print the final count of cars detected after stopping the video
        print("Total number of cars detected:", total_cars_detected)
        break
# Release the video capture object and close all OpenCV windows
cap.release()
cv2.destroyAllWindows()
```

Tkinter Code

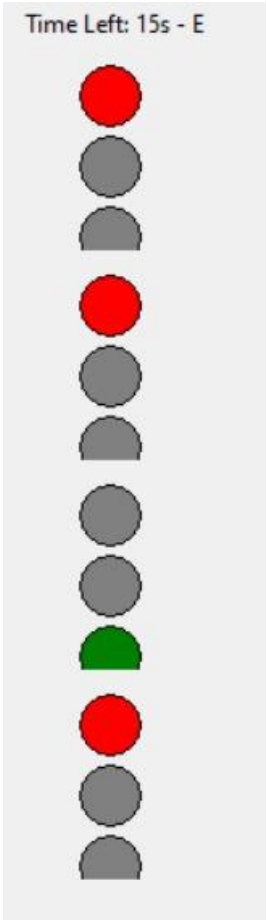
```
import tkinter as tk
import time
import random
dir_map = {'north': 'N', 'south': 'S', 'east': 'E', 'west': 'W'}
class Signal:
    def _init_(self, root, side):
        self.root = root
        self.side = side
        self.canvas = tk.Canvas(root, width=50, height=100)
        self.canvas.pack()
        self.red_light = self.canvas.create_oval(10, 10, 40, 40, fill='grey')
        self.yellow_light = self.canvas.create_oval(10, 45, 40, 75, fill='grey')
        self.green_light = self.canvas.create_oval(10, 80, 40, 110, fill='grey')
```

```
    def set_green(self):
        self.canvas.itemconfig(self.red_light, fill='grey')
        self.canvas.itemconfig(self.yellow_light, fill='grey')
        self.canvas.itemconfig(self.green_light, fill='green')
    def set_yellow(self):
        self.canvas.itemconfig(self.red_light, fill='grey')
        self.canvas.itemconfig(self.yellow_light, fill='yellow')
        self.canvas.itemconfig(self.green_light, fill='grey')
    def set_red(self):
        self.canvas.itemconfig(self.red_light, fill='red')
        self.canvas.itemconfig(self.yellow_light, fill='grey')
        self.canvas.itemconfig(self.green_light, fill='grey')
root = tk.Tk()
root.title("Traffic Light Simulation")
tk.Label(root, text='N').pack(side='top')
tk.Label(root, text='W').pack(side='left')
tk.Label(root, text='E').pack(side='right')
tk.Label(root, text='S').pack(side='bottom')
time_left = tk.Label(root, text='')
time_left.pack()
signals = {}
for side in ['north', 'south', 'east', 'west']:
```

```
    signal = Signal(root, side)
    signals[side] = signal
def control_signals():
    while True:
        for signal in signals.values():
            signal.set_red()
        counts = {}
        for side in signals:
            counts[side] = random.randint(0,10)
        sequence = sorted(counts, key=counts.get, reverse=True)
        for side in sequence:
            signals[side].set_green()
            for other_side in sequence:
                if other_side != side:
                    signals[other_side].set_red()
            time_left['text'] = f'Time Left: 15s - {dir_map[side]}'
            root.update()
            time.sleep(15)
            signals[side].set_yellow()
            time_left['text'] = f'Time Left: 3s - {dir_map[side]}'
            root.update()
            time.sleep(3)
```

```
            signals[side].set_red()
            time_left['text'] = f'Time Left: 1s - {dir_map[side]}'
            root.update()
            time.sleep(1)
        time.sleep(1)
root.after(1000, control_signals)
root.mainloop()
```

Output:



```
Total number of cars detected: 3
```

Conclusion

In conclusion, the development of an automatic traffic signal indicator using live video capture presents a promising avenue for enhancing traffic management systems. By leveraging machine learning techniques, we can create intelligent systems capable of analyzing real-time video feeds to optimize traffic signal control and improve overall traffic flow.

Throughout this project, we have explored various machine learning algorithms and methodologies to develop an effective traffic signal indicator. We began by conducting a comprehensive literature review to understand existing research and identify gaps in current approaches. Building upon this knowledge, we collected and preprocessed traffic video data, ensuring its quality and relevance for model training.

Our methodology involved the selection and implementation of appropriate machine learning algorithms for vehicle detection and classification. We leveraged techniques such as convolutional neural networks (CNNs) for feature extraction and classification, considering their effectiveness in handling visual data like traffic videos. Additionally, we explored data augmentation and transfer learning strategies to improve model performance.assess the model's effectiveness in detecting and classifying vehicles in real-time traffic situations.

In conclusion, the development of an automatic traffic signal indicator using live video capture holds immense potential for optimizing traffic management and enhancing road safety. By harnessing the power of machine learning and real-time video analysis, we can pave the way for smarter and more efficient traffic control systems in the future.