Scilab case study project on

Smart Traffic Light Control System

Krishnendu Roy

National Institute of Technology Rourkela

Image Processing

Date: 03/02/2025

Abstract

The Smart Traffic Light Control System using Image Processing aims to enhance pedestrian safety and improve traffic flow by dynamically adjusting traffic light timings based on realtime traffic density. Utilizing the Canny Edge Detection Technique, this system efficiently reduces pedestrian wait times at crossings. By eliminating the need for traditional hardware sensors and using image processing, this method offers a cost-effective and user-friendly solution for traffic management. The system's effectiveness is demonstrated across varying traffic conditions, showcasing its potential to revolutionize urban traffic control

1. Introduction

Traffic congestion has become a pressing issue in modern cities, significantly impacting daily life, economic productivity, and environmental sustainability. Pedestrians also face greater challenges, with limited opportunities to cross busy streets safely, often resulting in accidents. Traditional traffic management systems, such as fixed signal timers and manually controlled junctions, struggle to adapt to real-time traffic dynamics. In response, smart traffic management systems have emerged as a promising solution. These systems leverage advanced technologies like sensors, cameras, and real-time data analytics to optimize traffic flow and enhance road safety. Image processing plays a pivotal role in smart traffic systems by enabling real-time analysis of traffic conditions. By processing raw images and videos captured from cameras and sensors, image processing techniques can extract meaningful insights, such as vehicle counts, speed measurements, and pedestrian movement patterns. This paper explores the concept of smart traffic management, focusing on the integration of image processing and other advanced technologies to address congestion, improve pedestrian safety, and ensure a seamless flow of vehicles in urban areas. The proposed approach aims to revolutionize traditional traffic systems by making them more adaptive, efficient, and responsive to the needs of modern cities.

2. Problem Statement

Urban traffic congestion has become a critical issue due to the increasing number of vehicles and inefficient traffic management systems, leading to prolonged travel times, higher fuel consumption, and safety concerns for both drivers and pedestrians. Traditional traffic control methods, such as fixed signal timings, fail to adapt to real-time traffic conditions, resulting in unnecessary delays and inefficiencies. To address this, a smart traffic management system utilizing image processing is proposed, enabling real-time analysis of vehicle density and pedestrian movement. By integrating cameras and sensors with intelligent algorithms, the system will dynamically adjust traffic signals to optimize flow and enhance road safety. This project aims to implement image processing techniques such as object detection and feature extraction to develop an adaptive and efficient traffic control solution, reducing congestion and improving urban mobility.

3. Basic concepts related to the topic

Traffic management involves regulating and optimizing vehicle movement on roads to reduce congestion and improve safety. Traditional traffic control systems rely on fixed-time signal controllers, which operate on pre-set intervals regardless of actual traffic conditions. However, modern adaptive traffic control systems use sensors, cameras, and data analytics to make real-time decisions based on vehicle density and pedestrian movement.

A smart traffic system typically includes:

- **Traffic Signals and Controllers** – Regulate traffic flow at intersections using sensors and timers.
- **Vehicle Detection Systems** – Identify and count vehicles using cameras, inductive loops, or radar sensors.
- **Pedestrian Detection and Assistance** – Identify pedestrian movement to adjust signal timings for safer crossings.

1. Image Processing

Image processing involves the manipulation and analysis of images to extract useful information. In this case, we process images of traffic conditions to determine the level of congestion.

2. RGB and Grayscale Images

- **RGB Image**: A colored image composed of three channels—Red, Green, and Blue.
- **Grayscale Image**: A single-channel image where pixel values range from 0 (black) to 255 (white), reducing complexity for analysis.
- **Conversion (rgb2gray)**: Since grayscale images require less computation, converting an RGB image to grayscale is a common preprocessing step.

3. Image Resizing (imresize)

- Resizing helps standardize image dimensions for uniform processing.
- It ensures that comparisons between images (e.g., reference and test images) are accurate.

4. Power-Law (Gamma) Transformation

- This is a type of image enhancement technique used to improve image contrast.
- Formula: $$S = cR^{\gamma}$$

  Where:

  - S is the output pixel value,
  - R is the input pixel value (normalized),
  - c is a scaling constant (set to 1),
  - γ is the gamma value (adjusts brightness and contrast).

- Higher gamma values (gamma > 1) make darker regions more visible.

5. Edge Detection (edge)

- **Purpose**: Identifies boundaries and structures in an image.
- **Canny Method** : The **Canny Edge Detection Algorithm** is a multi-step process used to detect edges in an image efficiently. It begins by converting the image to grayscale if it is in RGB format, reducing computational complexity. To minimize noise, a **Gaussian filter** is applied, smoothing the image while preserving important structural details. Next, the **Sobel operator** is used to compute the image gradient, determining the intensity changes in both horizontal and vertical directions. The gradient magnitude and direction are then calculated to identify potential edges. A process called **non-maximum suppression** follows, which thins the edges by removing weaker pixels that are not part of the most significant gradients. Finally, **hysteresis thresholding** is applied to differentiate between strong edges, weak edges, and noise. This step ensures that only edges with significant intensity differences are retained while weak edges connected to strong ones are preserved. The Canny method is highly effective in detecting edges while reducing false positives and maintaining precision.
- **Thresholding**: Controls edge sensitivity (higher threshold detects only strong edges).

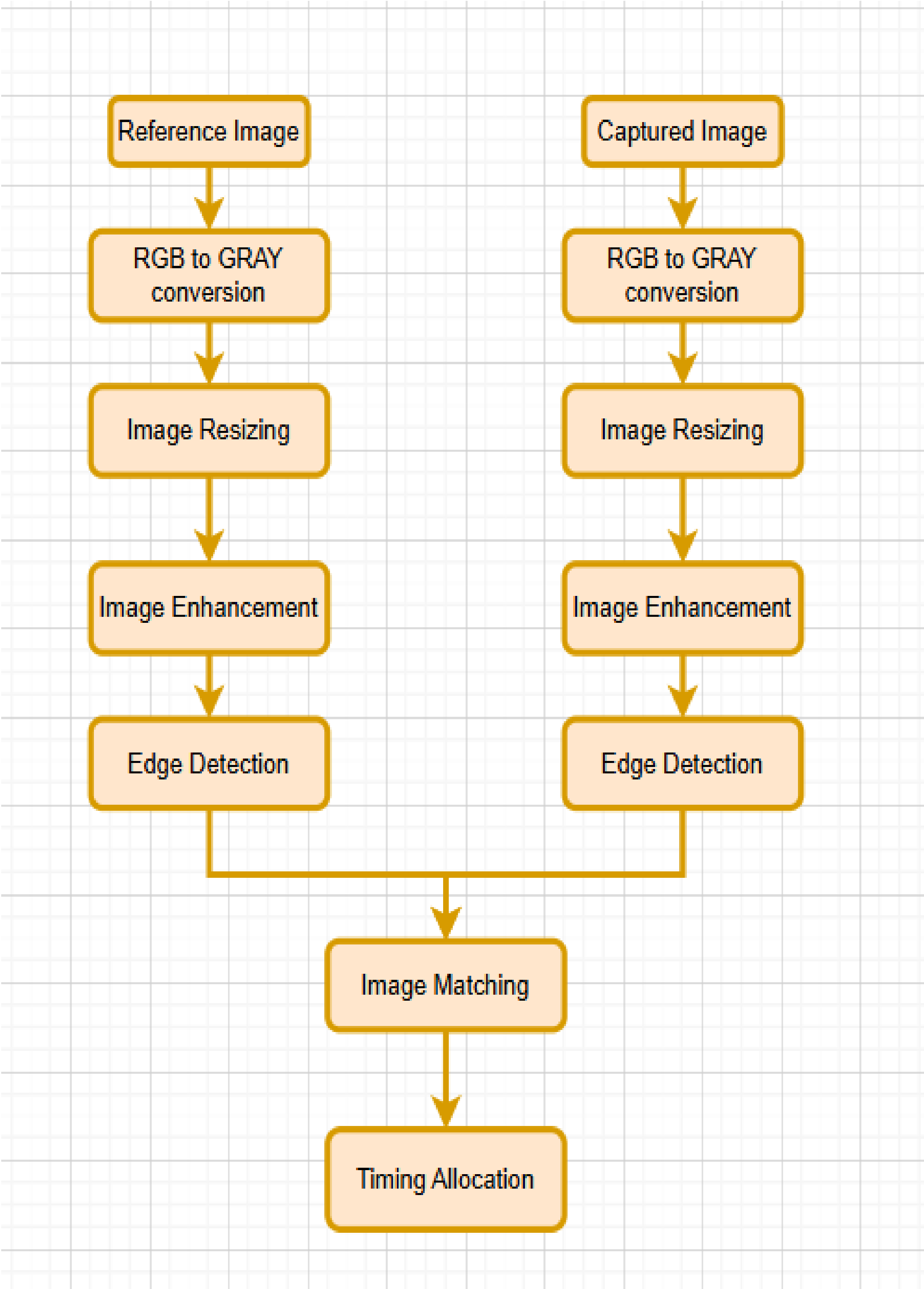
6. Image Matching for Traffic Analysis

- **Absolute Difference (abs)**: Compares edge-detected images of current traffic and a reference "less traffic" image.
- **Pixel Mismatch Calculation**: Determines how different the two images are.
- **Matching Percentage**:

$$\text{Matching Percentage} = \left(1 - \frac{\text{Mismatched Pixels}}{\text{Total Pixels}}\right) \times 100$$

  - **High Matching (> 85%)** → Less traffic.
  - **Moderate Matching (70-85%)** → Moderate traffic.
  - **Low Matching (< 70%)** → Heavy traffic.

4. Flowchart



Reference Image
Captured Image
RGB to GRAY conversion
RGB to GRAY conversion
Image Resizing
Image Resizing
Image Enhancement
Image Enhancement
Edge Detection
Edge Detection
Image Matching
Timing Allocation

5. Software/Hardware used

Operating System: Windows 11

Toolbox: Image Processing and Computer Vision Toolbox, Version:4.5.0

Hardware: Personal Computer with 12th Gen Intel Core Processor, 16GB RAM

Software: Scilab Version: 2025.0.0 and Microsoft Office 2021

6. Procedure of execution

I. Launch the Scilab Desktop on the computer.

II. Download the toolbox Image Processing and Computer Vision Toolbox, Version:4.5.0

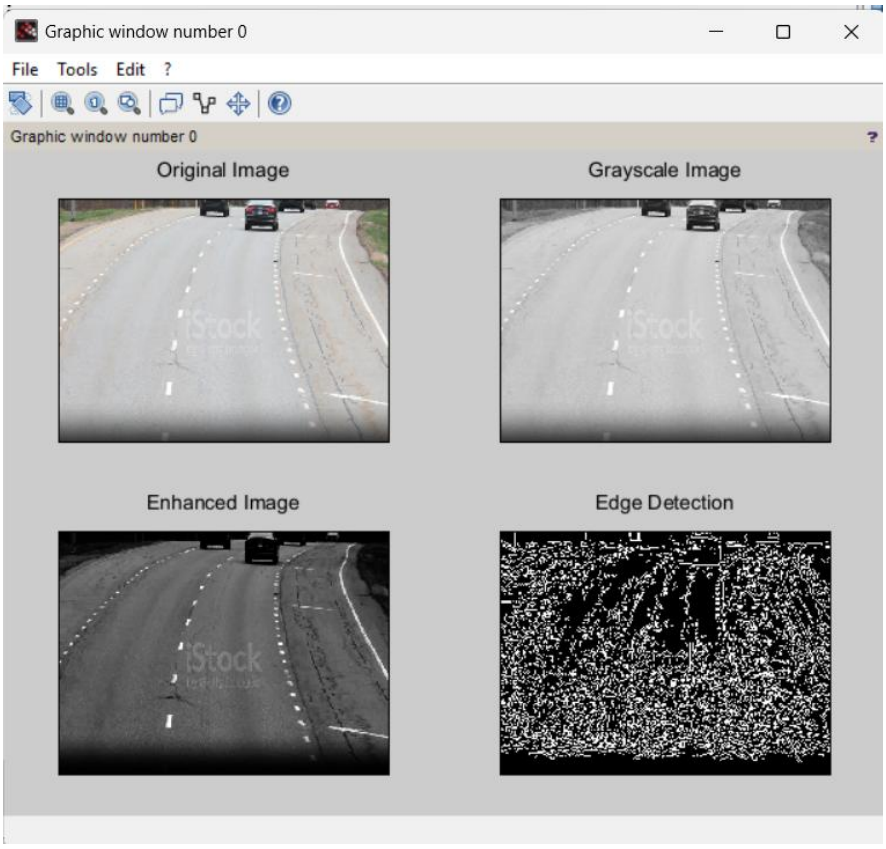
III. Open any file of your choice “high_traffic.sci”, ”moderate_traffic.sci” or “less_traffic.sci”

IV. Run the code

V. Observe the result

7. Result

Less Traffic Situation Output:

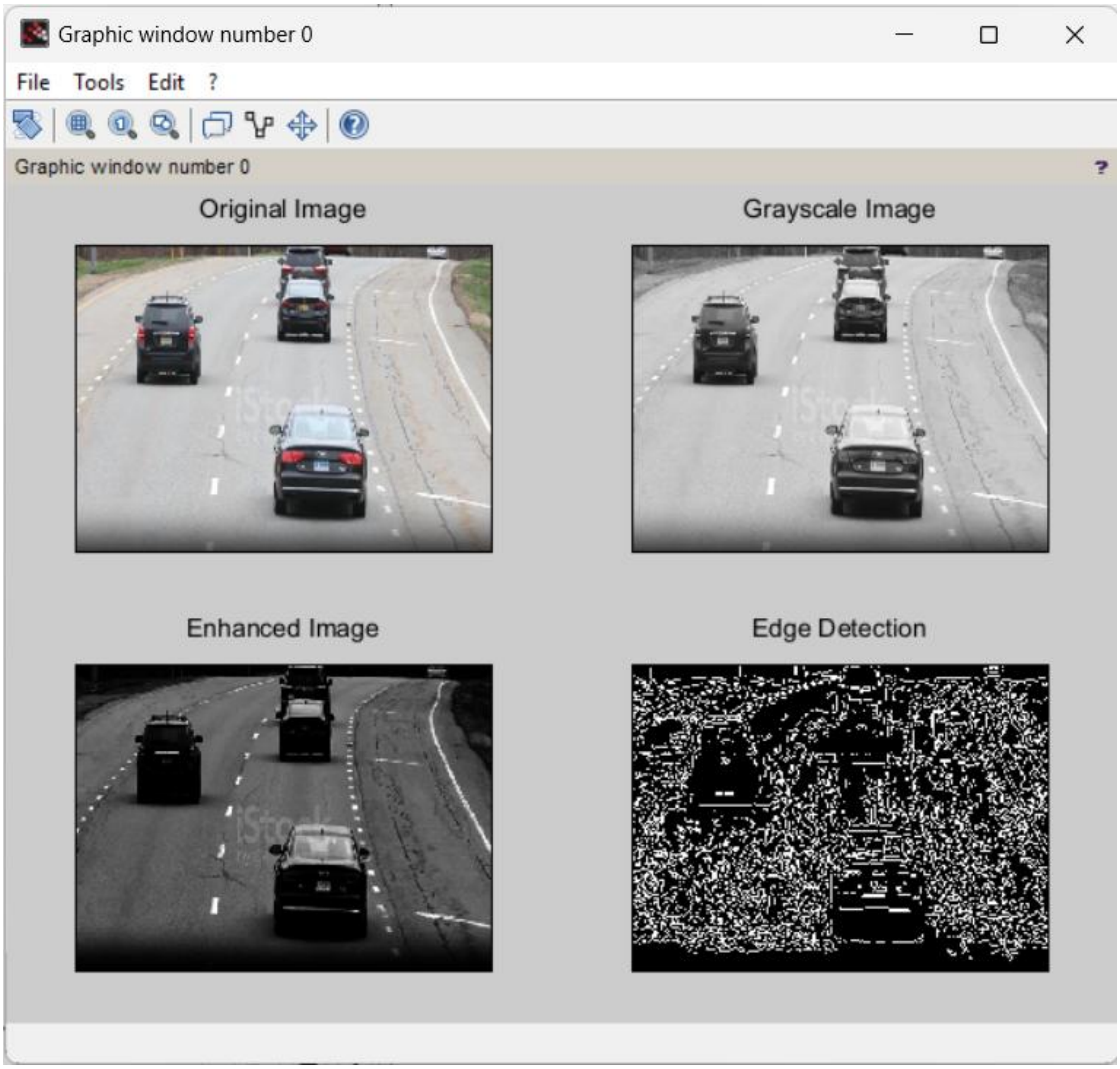


```
--> exec('C:\Users\Lenovo\Documents\test.sci', -1)
Enter image name: less.png

  "Less Traffic Situation"
  "Traffic Stops in: 2 seconds"
  "Pedestrian Walk Time: 20 seconds"
```

In case less or equal no of vehicles is detected from the reference image the traffic stops in 2 seconds and the pedestrian walk time is 20 seconds

Moderate Traffic Situation Output:



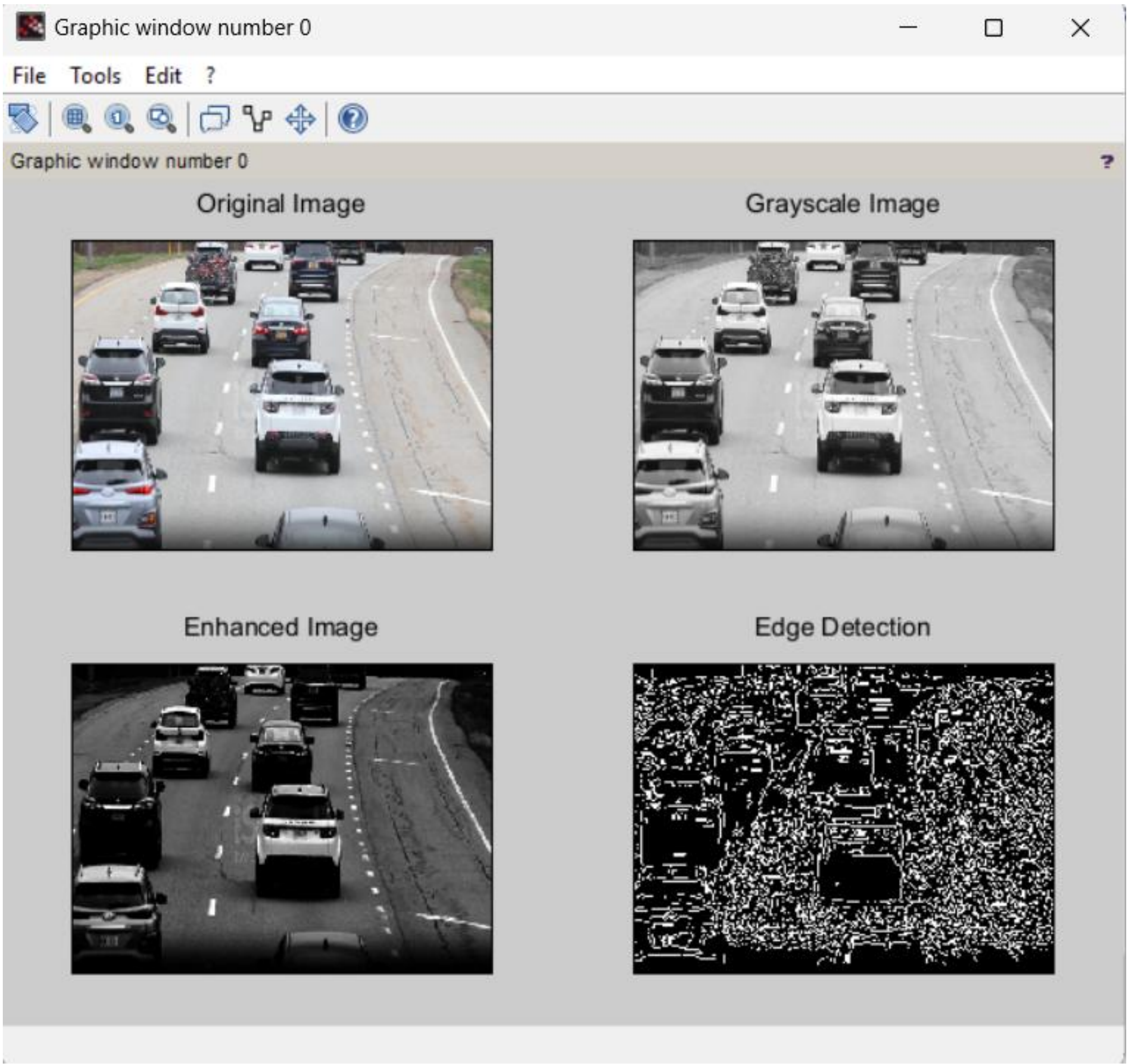
```
--> exec('C:\Users\Lenovo\Documents\test.sci', -1)
Enter image name: moderate.png

  "Moderate Traffic Situation"
  "Traffic Stops in: 10 seconds"
  "Pedestrian Walk Time: 20 seconds"

-->
```

In case, a moderate number of vehicles in acquired image when compared to that of reference image is detected, the traffic stops in 10 seconds and allows pedestrian walk for 20 seconds.

Moderate Traffic Situation Output:



```
--> exec('C:\Users\Lenovo\Documents\test.sci', -1)
Enter image name: high.png

  "Heavy Traffic Situation"
  "Traffic Stops in: 20 seconds"
  "Pedestrian Walk Time: 20 seconds"
```

In case, a greater number of vehicles in acquired image when compared to that of the reference image is detected, the traffic stops in 20 seconds and allows pedestrian walk for 20 seconds

References

1) **Title of the paper**: Smart Traffic Light Control System Using Image Processing **Name of journal/publication** : The Institute of Electrical and Electronics Engineers **(IEEE) Link1**: https://ieeexplore.ieee.org/document/10010764

2) **Images Used:** Palisades Interstate Parkway Going North 4K Stock Video - Download Video Clip Now - Traffic, Driving, Car - iStock

3) **Flowchart is made using:** Untitled Diagram.drawio - draw.io