TRAFFIC MANAGEMENT

Traffic management specifications and technology development

1.Traffic signal detection sensors:

Type: Inductive Loop Detector (ILD) Sensor

Specifications:

✓ Operating Frequency	: 24GHz
✓ Max. Detection Range	: 300m(984ft)
✓ Max. Detection Range	: 260m(853ft)

(Passenger Car)

Purpose: To make the movement of goods and persons as efficient, orderly, and safe as possible.

2. IR (Infrared) Sensor:

Type: Infrared Proximity Sensors

Specifications:

✓ Operating Frequency	: 300GHz-400GHz
✓ Maximum Range	: 100cm-500cm
✓ Minimum Voltage	: 2.5V

Purpose: Signal control, Volume, Speed, and Class measurement, as well as detecting pedestrians in crosswalks.

3. Camera Systems:

Type: High-Resolution Cameras (e.g., Raspberry Pi Camera)

Specifications:

- ✓ 4K cameras with a resolution of 3840*2160P

Purpose: It provides alarm monitoring and transparent communication between the system and tracking security along the roads.

4.Environmental Sensors:

Type: Environmental Monitoring Sensors(e.g., light sensors)

Specifications:

- ✓ Parameters: CO2 levels, Noise levels

Purpose: Traffic management serves a wide range of applications such as: Variable message signs, such as warnings for high winds, poor visibility and dynamic speed controls.

The Technology Development

Hardware Setup:

- ✓ **Raspberry Pi:** Utilize a Raspberry Pi board as the core processing unit to control sensors, capture video, light sensor and run the software.
- ✓ **Hardware Setup:** You'll need a raspberry pi board a compatible camera module, and an internet connection.

Camera setup:

- ✓ Connect your camera module to the Raspberry Pi.

Capture video feed:

- ✓ Use OpenCV to capture video from the camera module. You can access the camera feed using OpenCV's video capture module.

Traffic Analysis:

- ✓ Implement traffic analysis algorithms using OpenCV. That include detecting vehicles, counting vehicles, and analyzing traffic flow.

Traffic Control Logic:

- ✓ Based on the analysis results, you can implement traffic control logic. For example, if there a traffic jam, you could change signal timings or trigger warnings.

Display and Alerts:

- ✓ You can use the Raspberry Pi to display traffic information on a screen or even provide alerts via LED's, sound, or remote notifications.

Data Logging and Reporting:

- ✓ Store and analyze traffic data for further analysis or reporting.
- ✓ You can use the Raspberry Pi's storage or cloud services for this purpose.

Integration:

- ✓ Consider integrating other technologies like machine learning for more advanced traffic analysis, IOT for remote control and monitoring.

Power and Connectivity:

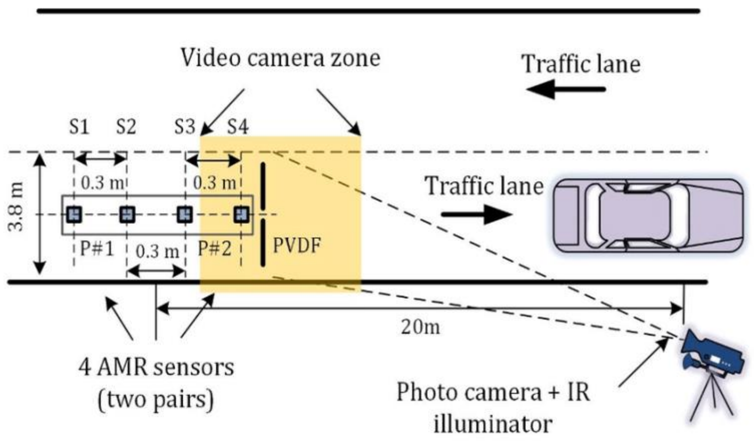
- ✓ Ensure your Raspberry Pi has a reliable power source and network connectivity for continuous operation.

Testing and Calibration:

- ✓ Thoroughly test the system in a controlled environment and calibrate it to suit your specific traffic control needs.

Deployment:

- ✓ Deploy the Raspberry Pi-based traffic control system at the desired location.



Raspberry Pi Python Script Traffic Detection Using OpenCV:

```
Import cv2

Import numpy as np

#Load a pre-trained vehicle detection model

vehicle _ cascade = cv2.CascadeClassifier( ‘ haarcascade  _ car.xml’)

#You can use a different model if needed


#Create a VideoCapture Object to access the camera

Cap = cv2.VideoCapture(0)

#You may need to specify a different camera source


While True:

    Ret, frame = cap.read()

    If not ret:

        Break

#Convert the frame to grayscale

     gray = cv2. cvtColor(frame, cv2.COLOR_BGR2GRAY)
```

```
#Detect vehicles in the frame
        vehicles = vehicle _ cascade. detectMultiScale ( gray, scaleFactor = 1.1, minNeighbours
= 5, minSize = (30, 30))
#Draw rectangles around detected vehicles
    for (x, y, w, h) in vehicles:
            cv2.rectangle(frame, (x,y), (x+ w, y + h), (0, 255, 0), 2)
#Display the frame with detected vehicles
            cv2.imshow('Traffic Management System', frame)
             if cv2.waitkey(1) & 0*FF == 27:
#Press Esc key to exit
                    break
cap.release()
cv2.destoryAllWindows()
```

This is starting point and complete traffic management system would require additional logic for analyzing traffic conditions, controlling signals, and integrating other sensors or systems.