Intelligent Traffic Light Control System using GNU Octave

Group 3

Contents

Individual Task Breakdown

	Member 1	**Member 2**	**Member 3**	**Member 4**
Name	**R.M.P.U. Rajapaksha**	**S.D . Jayasinghe**	**D.R.G.A . Wijesekera**	**V.T. Dahanayake**
Code Implementation	✓	✓	✓	✓
Mathematical Modeling	✓	✓	✓	✓
Visualization	✓	✓	✓	✓
Testing & Debugging	✓	✓	✓	✓
Final Report & Documentation	✓	✓	✓	✓

Acknowledgment

As a team we would like to express our sincere gratitude to our lecturers and lab instructors who teach and continue to guidance, support, and also encourage throughout the duration of this semester. Their instructions and guidance provided us with a solid foundation in Discreate Mathematics and Octave programming which is essential for the completion of this project. Furthermore, everyone who contributed to the successful completion of this report is acknowledged for their valuable contributions, which really inspired and shaped the work that we have done.

Introduction

In the present society, one of the most significant problems which is facing in cities throughout the world is traffic congestion. The traditional traffic signal systems mostly use fixed timers which frequently created inefficient traffic flows, increased wait times and less effective emergency response. As undergraduates we wish to address this problem by utilizing GNU Octave to develop and implement Smart Traffic Light Control System in this project. This innovative software prioritizes emergency vehicles and ensures optimal traffic flow by dynamically changing traffic lights based on real time traffic simulation data.

Additionally, the project provides a hands-on application of several Discrete Mathematics programming principles that were covered in the course, including loops, conditionals, matrices, vectorized operations, and graphing functions.

Methodology

In order to study and control smart traffic light system flow over a 24 hour period, we have used a simulation-based technique to the development of the smart traffic signal system. In order to simulate vehicle movement over six lanes and four directions throughout the day, firstly we have created synthetic traffic data. In order to simulate real-world variability, the simulation looped through the red, green, and yellow traffic signal phases, each of which was shown with a randomly chosen duration. Furthermore b y calculating the total number of vehicles per hour, peak traffic hours could be identified for analysis. To see how traffic moved through each lane throughout the day, a line plot was made in this scenario. A vehicle count was chosen at random, and the green light length was modified in accordance with its value in order to dynamically adjust to traffic conditions. Then, emergency vehicles, such as fire engines and ambulances, and more were given priority by using immediate green signals instead of regular timing. The present traffic sample was compared to past peak numbers in order to identify congestion. If a match was discovered, then alarms were triggered. In order to demonstrate mathematical modeling of traffic flow, a simple quadratic model was also solved. Performance metrics were then reflected by a vectorized computation of average vehicle speed. This thorough approach simulates an intelligent and flexible traffic control system by combining data production, conditional logic, real-time prioritization, and analytical computation.

Assumptions in the Code

- Traffic volume for each lane and direction is randomly generated between 5 and 100 vehicles per hour.
- Traffic behavior is assumed to be consistent throughout the 24-hour period, with no consideration for rush hours.
- Traffic signals operate using three fixed phases (Red, Green, Yellow), with each phase displayed for only 1 second.
- Green light duration is randomly chosen between 5 to 15 seconds, without being adaptive to actual traffic density.
- Peak hour is detected using the total number of vehicles across all lanes, without analyzing per-lane congestion.
- Dynamic green time is determined using a single random vehicle count with basic threshold logic, unrelated to real-time traffic data.
- Emergency vehicle detection is based on a hardcoded type ('ambulance') and only results in a console message, without affecting traffic flow.
- Congestion is assumed if current traffic randomly matches values in a predefined historical peak list, ignoring real-world congestion indicators.
- The entire simulation runs only once and lacks real-time progression or iterative updates over time.

Script

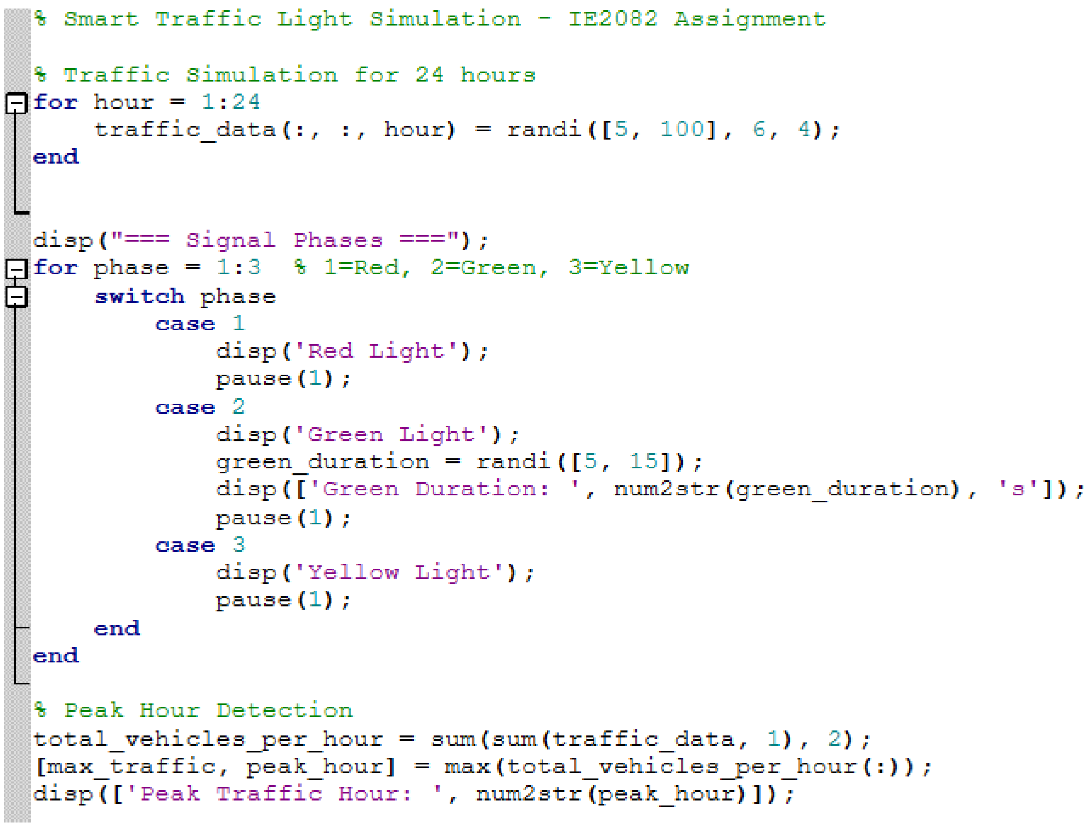
```
% Traffic Simulation for 24 hours

for hour = 1:24

    traffic_data(:, :, hour) = randi([5, 100], 6, 4);

end


disp("=== Signal Phases ===");

for phase = 1:3  % 1=Red, 2=Green, 3=Yellow

    switch phase

        case 1

            disp('Red Light');

            pause(1);

        case 2

            disp('Green Light');

            green_duration = randi([5, 15]);

            disp(['Green Duration: ', num2str(green_duration), 's']);

            pause(1);

        case 3

            disp('Yellow Light');

            pause(1);

    end

end
```

```
% Peak Hour Detection

total_vehicles_per_hour = sum(sum(traffic_data, 1), 2);

[max_traffic, peak_hour] = max(total_vehicles_per_hour(:));

disp(['Peak Traffic Hour: ', num2str(peak_hour)]);


% Plotting

figure;

for lane = 1:6

    plot(1:24, squeeze(traffic_data(lane, 1, :)));

    hold on;

end

xlabel('Hour');

ylabel('Vehicle Count');

title('Traffic Flow per Lane');

legend('Lane 1', 'Lane 2', 'Lane 3', 'Lane 4', 'Lane 5', 'Lane 6');

grid on;


% Conditional Green Light Time

vehicle_count = randi([1, 100]);

if vehicle_count > 50

    green_time = 20;

elseif vehicle_count < 10

    green_time = 5;

else

    green_time = 10;

end

disp(['Dynamic Green Time: ', num2str(green_time)]);
```

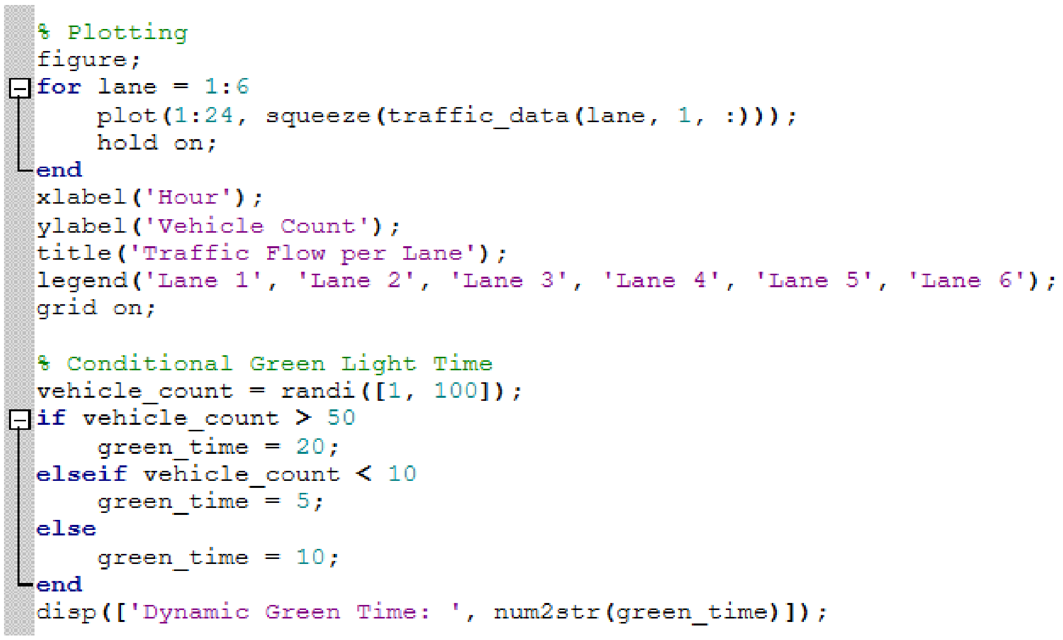
```
% Emergency Vehicle

vehicle_type = 'ambulance';

switch vehicle_type

    case 'ambulance'

        disp('Priority Green for Ambulance!');

    case 'fire_truck'

        disp('Priority Green for Fire Truck!');

    otherwise

        disp('Normal Traffic Flow');

end


% Congestion Detection

historical_peaks = [90, 95, 100];

current_traffic = randi([80, 100], 1, 5);

disp('Current Traffic Snapshot:');

disp(current_traffic);

congested = ismember(current_traffic, historical_peaks);

if any(congested)

    disp('Congestion Detected!');

else

    disp('Traffic Normal.');

end
```

```
% Quadratic Equation for Flow Modeling

a = 1; b = -3; c = 2;

D = b^2 - 4*a*c;

if D >= 0

    root1 = (-b + sqrt(D)) / (2*a);

    root2 = (-b - sqrt(D)) / (2*a);

    disp(['Quadratic Roots: ', num2str(root1), ', ', num2str(root2)]);

else

    disp('No Real Roots');

end


% Vectorized Operation

speeds = [40 35 30 45 50 38];

average_speed = mean(speeds);

disp(['Average Speed: ', num2str(average_speed)]);
```

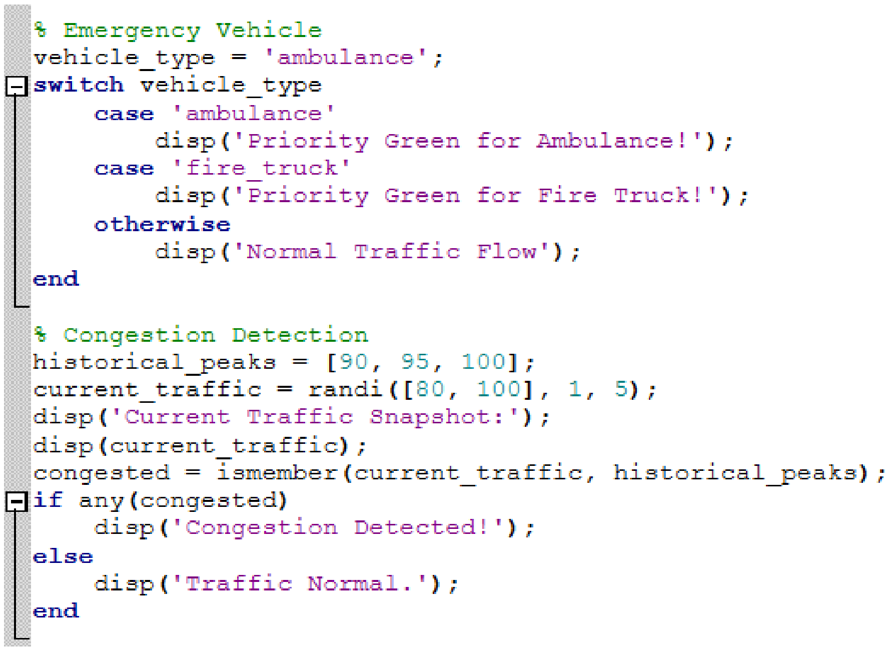
Screenshots when compiling



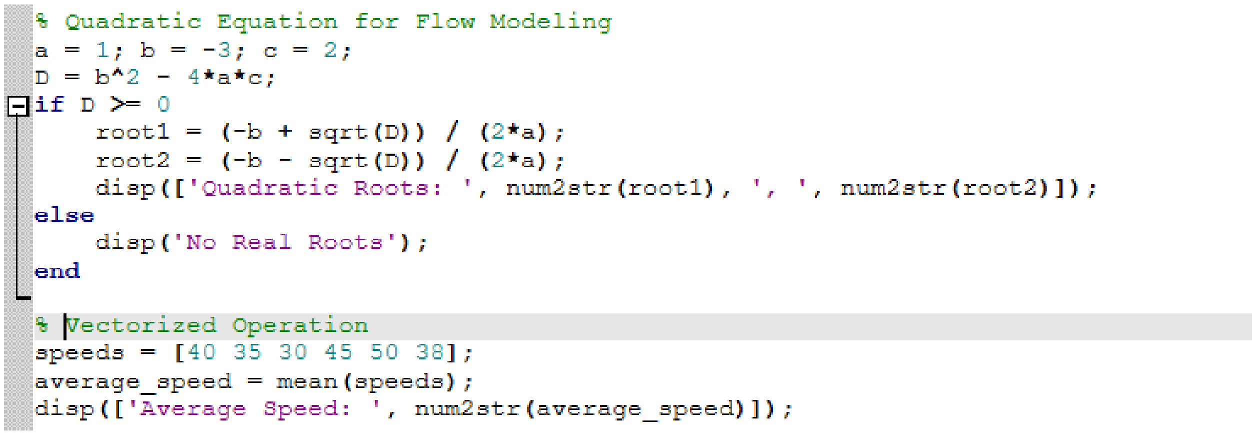
```
% Smart Traffic Light Simulation - IE2082 Assignment

% Traffic Simulation for 24 hours
for hour = 1:24
    traffic_data(:, :, hour) = randi([5, 100], 6, 4);
end


disp("=== Signal Phases ===");
for phase = 1:3  % 1=Red, 2=Green, 3=Yellow
    switch phase
        case 1
            disp('Red Light');
            pause(1);
        case 2
            disp('Green Light');
            green_duration = randi([5, 15]);
            disp(['Green Duration: ', num2str(green_duration), 's']);
            pause(1);
        case 3
            disp('Yellow Light');
            pause(1);
    end
end

% Peak Hour Detection
total_vehicles_per_hour = sum(sum(traffic_data, 1), 2);
[max_traffic, peak_hour] = max(total_vehicles_per_hour(:));
disp(['Peak Traffic Hour: ', num2str(peak_hour)]);
```



```
% Plotting
figure;
for lane = 1:6
    plot(1:24, squeeze(traffic_data(lane, 1, :)));
    hold on;
end
xlabel('Hour');
ylabel('Vehicle Count');
title('Traffic Flow per Lane');
legend('Lane 1', 'Lane 2', 'Lane 3', 'Lane 4', 'Lane 5', 'Lane 6');
grid on;

% Conditional Green Light Time
vehicle_count = randi([1, 100]);
if vehicle_count > 50
    green_time = 20;
elseif vehicle_count < 10
    green_time = 5;
else
    green_time = 10;
end
disp(['Dynamic Green Time: ', num2str(green_time)]);
```

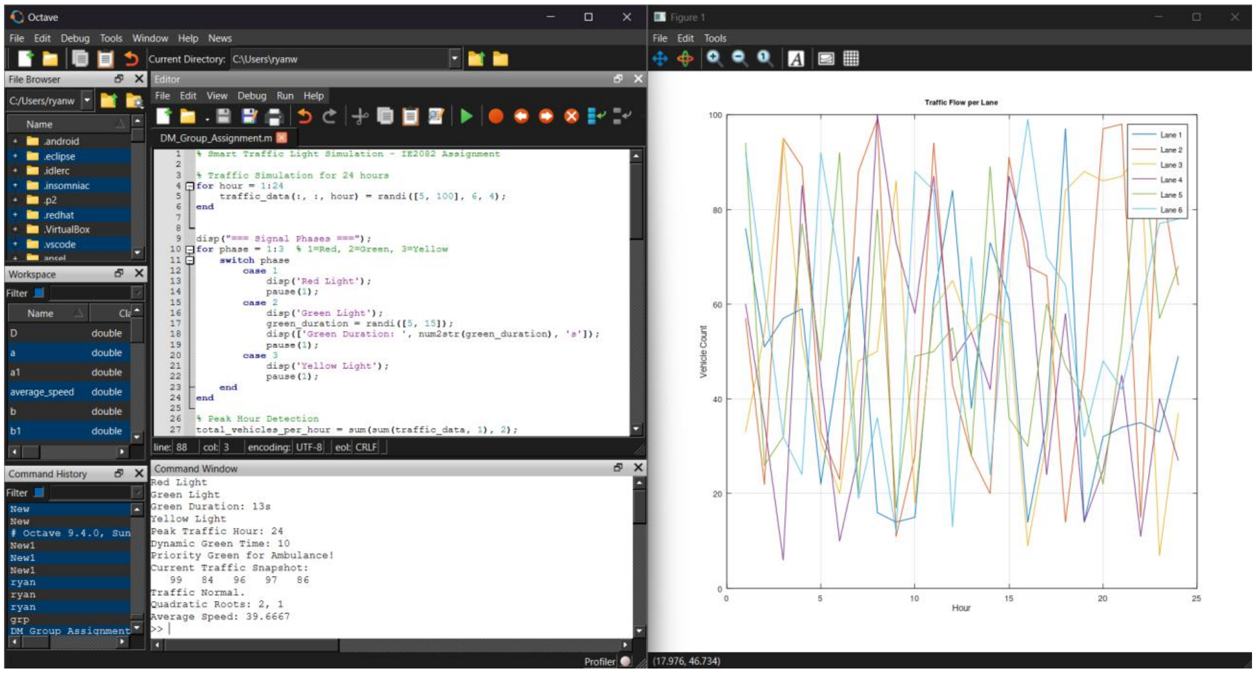


```matlab
% Emergency Vehicle
vehicle_type = 'ambulance';
switch vehicle_type
    case 'ambulance'
        disp('Priority Green for Ambulance!');
    case 'fire_truck'
        disp('Priority Green for Fire Truck!');
    otherwise
        disp('Normal Traffic Flow');
end

% Congestion Detection
historical_peaks = [90, 95, 100];
current_traffic = randi([80, 100], 1, 5);
disp('Current Traffic Snapshot:');
disp(current_traffic);
congested = ismember(current_traffic, historical_peaks);
if any(congested)
    disp('Congestion Detected!');
else
    disp('Traffic Normal.');
end
```



```matlab
% Quadratic Equation for Flow Modeling
a = 1; b = -3; c = 2;
D = b^2 - 4*a*c;
if D >= 0
    root1 = (-b + sqrt(D)) / (2*a);
    root2 = (-b - sqrt(D)) / (2*a);
    disp(['Quadratic Roots: ', num2str(root1), ', ', num2str(root2)]);
else
    disp('No Real Roots');
end

% Vectorized Operation
speeds = [40 35 30 45 50 38];
average_speed = mean(speeds);
disp(['Average Speed: ', num2str(average_speed)]);
```

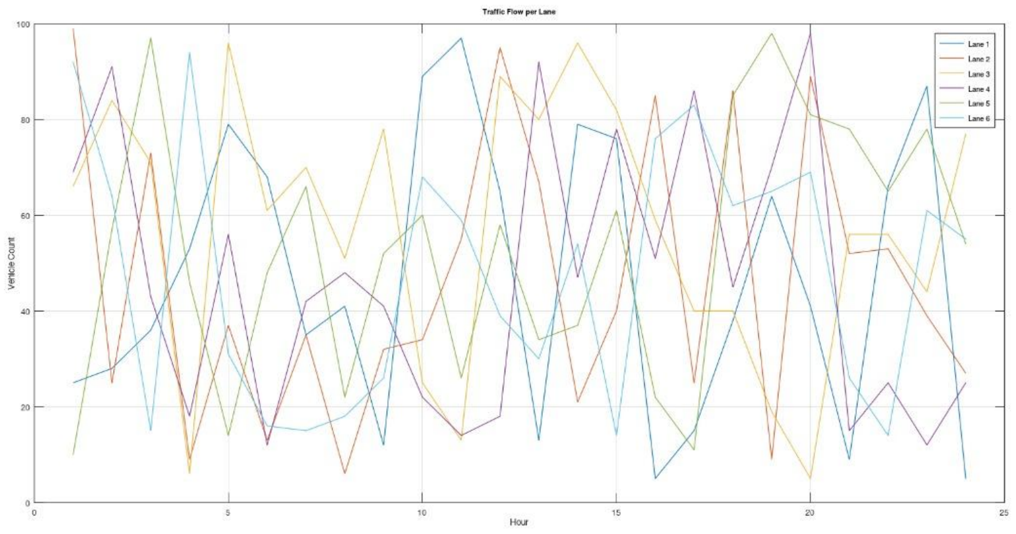
Result



```
% Smart Traffic Light Simulation - IE2082 Assignment

% Traffic Simulation for 24 hours
for hour = 1:24
    traffic_data(:, :, hour) = randi([5, 100], 6, 4);
end


disp("=== Signal Phases ===");
for phase = 1:3  % 1=Red, 2=Green, 3=Yellow
    switch phase
        case 1
            disp('Red Light');
            pause(1);
        case 2
            disp('Green Light');
            green_duration = randi([5, 15]);
            disp(['Green Duration: ', num2str(green_duration), 's']);
            pause(1);
        case 3
            disp('Yellow Light');
            pause(1);
    end
end

% Peak Hour Detection
total_vehicles_per_hour = sum(sum(traffic_data, 1), 2);
```

```
Red Light
Green Light
Green Duration: 13s
Yellow Light
Peak Traffic Hour: 24
Dynamic Green Time: 10
Priority Green for Ambulance!
Current Traffic Snapshot:
   99   84   96   97   86
Traffic Normal.
Quadratic Roots: 2, 1
Average Speed: 39.6667
>>
```

Graphical Output



Conclusion

This assignment provided us with a great chance to use programming concepts and discrete mathematics in the real-world, practical scenario which is Intelligent Traffic Light Control System. We have created a system that reacts to peak hours and the presence of emergency vehicles. It is done by modeling dynamic traffic patterns, implementing adaptive signal phase controls. We have used GNU Octave as our simulation platform for our project. We developed an intelligent, data-driven traffic control system by utilizing loops, conditional statements, matrix and vector operations, and mathematical operations like solving quadratic equations.

Beyond technical execution, this project demonstrates the value of data visualization, logical thinking, and real-time decision-making in resolving complex technical issues. Additionally, it enhanced the link between mathematical theory and practical applications, which improved our comprehension of how computational tools may support safer and more effective infrastructure. Future systems modeling and automation efforts will benefit greatly from this real-world experience.