Fuzzy Traffic Control

A Smart Solution for Urban Congestion Alleviation

A MINI PROJECT REPORT

18CSC305J - ARTIFICIAL INTELLIGENCE

Submitted by

SNEHAL SUKUNDARI [RA2111027010049]
N. SAI TUSHAR [RA2111027010050]
KSHITIJ RASTOGI [RA2111027010051]
SEYJUTI BANERJEE [RA2111027010052]

Under the guidance of

Dr. Arthy M

Assistant Professor, Department of Computer Science and Engineering

in partial fulfillment for the award of the degree

of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE & ENGINEERING

of

FACULTY OF ENGINEERING AND TECHNOLOGY

S.R.M. Nagar, Kattankulathur, Chengalpattu District

MAY 2024

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

(Under Section 3 of UGC Act, 1956)

BONAFIDE CERTIFICATE

Certified that Mini project report titled Fuzzy Traffic Control *A Smart Solution for Urban Congestion Alleviation* is the bona fide work of **Snehal Sukundari (RA2111027010049), N. Sai Tushar (RA2111027010050), Kshitij Rastogi (RA2111027010051), Seyjuti Banerjee (RA2111027010052)** who carried out the minor project under my supervision. Certified further, that to the best of my knowledge, the work reported herein does not form any other project report or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

SIGNATURE

FACULTY NAME

Dr. Arthy M

Designation

Assistant Professor

Department

Department of Computer Science and Engineering

ABSTRACT

- Traffic fuzzy control is a technique used to optimize traffic flow and reduce congestion in urban areas
- By applying fuzzy logic principles to traffic control systems, it is possible to create adaptive and responsive traffic management systems.
- Fuzzy logic-based traffic control systems offer a promising solution by incorporating human-like reasoning and decision-making processes into traffic management.
- Fuzzy logic algorithms are employed to interpret the imprecise and uncertain nature of traffic data, enabling the system to adjust traffic signal timings dynamically.

ABBREVIATIONS

1. **FTC -** Fuzzy Traffic Control
2. **STCUCA -** Smart Solution for Urban Congestion Alleviation
3. **FuzzTraC -** Fuzzy Traffic Control
4. **UCA -** Urban Congestion Alleviation
5. **FuzzyTC -** Fuzzy Traffic Control
6. **SmartTraffic -** Smart Solution for Urban Congestion Alleviation
7. **UCASol -** Urban Congestion Alleviation Solution
8. **FuzzConAllev -** Fuzzy Control for Congestion Alleviation
9. **FTCSol -** Fuzzy Traffic Control Solution
10. **STCA -** Smart Traffic Congestion Alleviation

CHAPTER 1

INTRODUCTION

- Traffic fuzzy control is a method of controlling traffic flow using fuzzy logic. It is a decision-making system that takes into account various factors such as traffic volume, congestion, and road conditions to optimize traffic signal timings.
- Traffic fuzzy control is a method of controlling traffic flow using fuzzy logic. It is a decision-making system that takes into account various factors such as traffic volume, congestion, and road conditions to optimize traffic signal timings.
- Reduced Travel Time: Optimized signal timings can lead to shorter travel times for drivers, resulting in increased efficiency and productivity.

CHAPTER 2

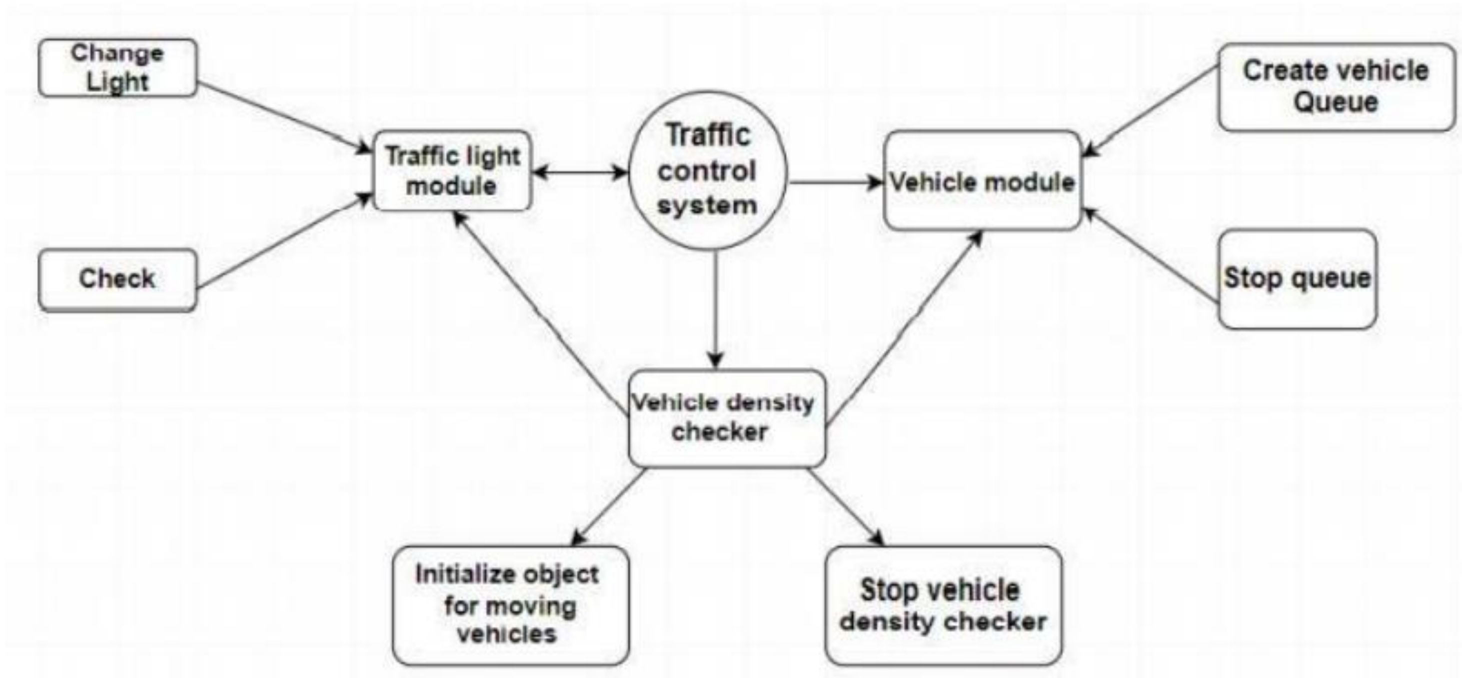
LITERATURE SURVEY

Authors	Title	Dataset	Methods	Remarks
Mojtaba Salehi, Iman Sepahvand, Mohammad Yarahmadi	A Traffic Lights Control System Based on Fuzzy Logic (2014)	Traffic density, queue length	Fuzzy logic technology, Visual Basic simulation, real-time strategies, fixed-time strategies, distance velocity number of vehicles, input/output member functions	Flexible system, easily adaptable fuzzy sets, requires further simulation verification
C. Ugwu, Bale, Dennis	An Application of Fuzzy Logic Model in Solving Road Traffic Congestion (2014)	Real-time traffic information	Fuzzy Logic, Fuzzification, Defuzzification, sensor-based model, fuzzy rules	Potential to minimize road traffic congestion, decision-making based on real-time information
Mohit Jha, Shailja Shukla	A Design Of Fuzzy Logic Traffic Controller for Isolated Intersections with Emergency Vehicle Priority System Using MATLAB Simulation (2014)	Vehicle queue length, waiting time	Queuing theory model, SIMULINK model, SimEvent toolbox, fuzzy inference system, embedded MATLAB function block, Poisson distribution function	Intelligent control, effective system, improvement in intersection vehicle crossing capacity, issues with 'C' coding in MATLAB function block
Mohit Jha, Shailja Shukla	Design and Simulation of Adaptive Traffic Light Controller Using Fuzzy Logic Control Sugeno Method (2015)	Waiting time, Number of Queue, Number of Departure	FUZZY INFERENCE SYSTEM SUGENO METHOD, Manual Kapasitas Jalan Indonesia (MKJI), Additive Method, Probabilistic OR Method, Max Method	Improved system performance, consideration of waiting time, queue length, and number of departures, lacks consideration for traffic density

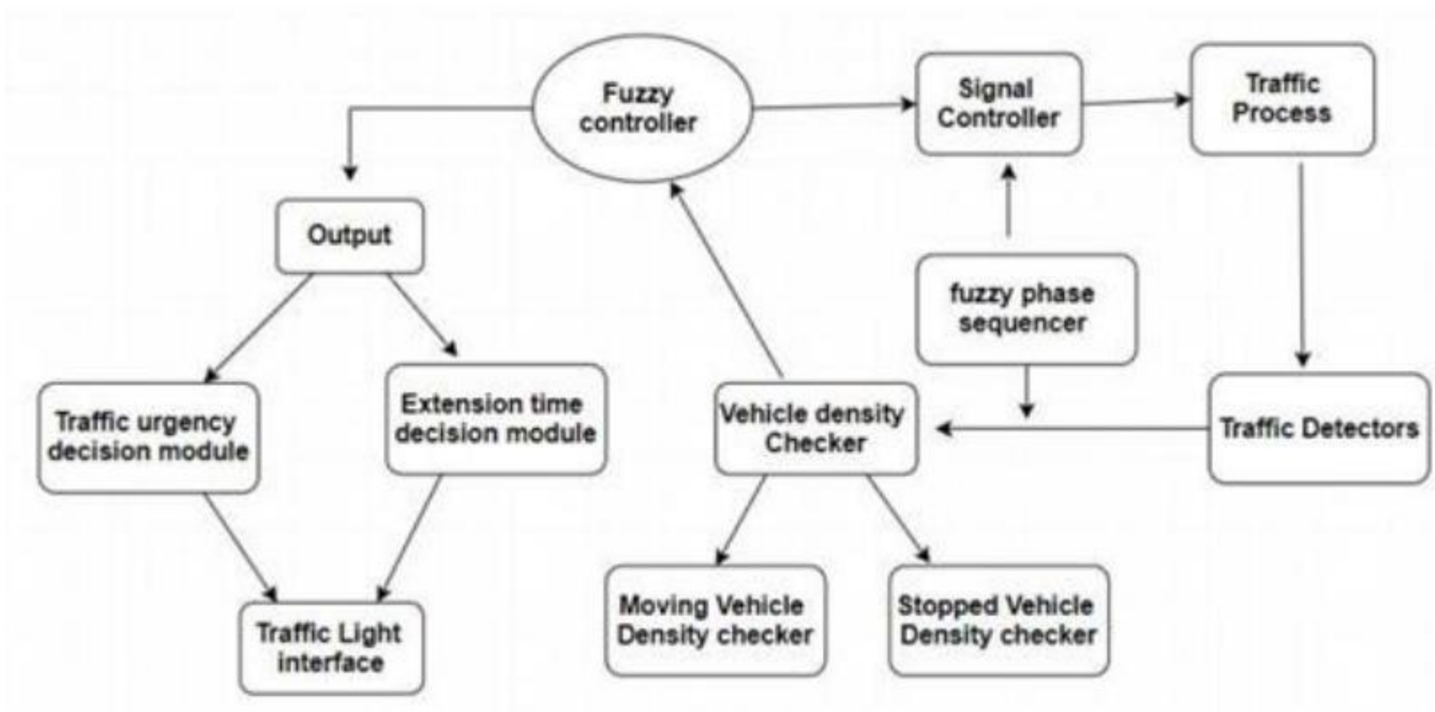
CHAPTER

SYSTEM ARCHITECTURE AND DESIGN

WORK FLOW DIAGRAM



ARCHITECTURE DIAGRAM



CHAPTER

CHAPTER 4

METHODOLOGY

Fuzzy Logic Technology

- Fuzzy logic is utilized as the core technology for designing intelligent traffic light control systems.
- Fuzzy logic allows for the representation of imprecise and uncertain data, making it suitable for modeling complex systems like traffic flow.
- Fuzzy sets and fuzzy rules are defined to capture the relationships between input variables (such as traffic density, queue length, etc.) and output variables (such as traffic light timings).
- Fuzzy logic plays a central role in the methodologies described in the provided studies. Here's how fuzzy logic is integrated into the traffic control systems:

1. Representation of Uncertainty:

- Fuzzy logic is used to handle the inherent uncertainty and imprecision in traffic data, such as traffic density, queue length, and vehicle counts.
- Traditional binary logic is inadequate for modeling real-world traffic systems due to the continuous and overlapping nature of traffic conditions. Fuzzy logic provides a more flexible approach by allowing degrees of truth instead of strict true/false values.

2. Fuzzy Sets and Membership Functions:
- Input variables, such as traffic density and queue length, are represented as fuzzy sets with associated membership functions.
- Membership functions define the degree of membership of an element in the fuzzy set. For example, a vehicle count of 20 might have a high membership in the "high traffic density" fuzzy set and a low membership in the "low traffic density" fuzzy set.

3. Fuzzy Rules:
- Fuzzy rules are defined based on expert knowledge and domain expertise to map input variables to output actions.
- These rules capture the relationships between the fuzzy input variables and the

control actions of the traffic lights.
- For example, a fuzzy rule might state: "If the traffic density is high and the queue length is increasing, then increase the green light duration."

4. Fuzzy Inference System:
- A fuzzy inference system is employed to process the fuzzy rules and determine appropriate control actions for the traffic lights.
- Fuzzification is performed to convert crisp input data into fuzzy sets using the defined membership functions.
- Inference is then carried out using the fuzzy rules to generate fuzzy output sets.
- Defuzzification is the final step where fuzzy output sets are converted back into crisp control actions for the traffic lights.

5. Adaptability and Learning:
- One advantage of fuzzy logic-based systems is their adaptability and ability to learn from real-time data.
- The fuzzy logic controllers can adjust their parameters and rules based on feedback from sensors and real-time traffic information, allowing them to dynamically respond to changing traffic conditions.

CHAPTER

CODING AND TESTING

Simulator Code:

```
import pygame
import time

from src.Common import Lane, DoubleLane
from src.Config import Config
from src.Controller.VehicleController import VehicleController
from src.Controller.TrafficController import TrafficController
from src.Controller.BackgroundController import BackgroundController


class Simulator:
    def _init_(self, caption):
        self.caption = caption
        self.surface = pygame.display.set_mode((Config['simulator']['screen_width'],
                                                Config['simulator']['screen_height']))
        self.vehicle_ctrl = VehicleController(self.surface)
        self.traffic_ctrl = TrafficController(self.surface)
        self.background_ctrl = BackgroundController(self.surface,
                                                    self.traffic_ctrl.get_traffic_lights(DoubleLane.Horizontal) +
                                                    self.traffic_ctrl.get_traffic_lights(DoubleLane.Vertical))
        self.clock = pygame.time.Clock()
        self.gap_between_switch = Config['simulator']['gap_between_traffic_switch']

        self.HORIZONTAL_SPAWN_EVENT = pygame.USEREVENT + 1
        self.VERTICAL_SPAWN_EVENT = pygame.USEREVENT + 2

        self.switching_traffic = False
        self.switching_traffic_start_time = None
        self.start_time = time.time()
        self.moving_averages =
self.vehicle_ctrl.get_moving_averages_num_vehicles_behind_traffic()

        self.is_extended = False
        self.green_light_remaining_time = Config['traffic_light']['green_light_duration']
        self.extension_notification_start_time = time.time() - 10

    def spawn(self, double_lane: DoubleLane):
        if double_lane == DoubleLane.Horizontal:
            self.spawn_single_vehicle(Lane.left_to_right)
            self.spawn_single_vehicle(Lane.right_to_left)
        elif double_lane == DoubleLane.Vertical:
```

```
            self.spawn_single_vehicle(Lane.bottom_to_top)
            self.spawn_single_vehicle(Lane.top_to_bottom)

    def spawn_single_vehicle(self, lane: Lane):
        self.vehicle_ctrl.create_vehicle(lane, self.traffic_ctrl.traffic_lights[lane])

    def main_loop(self):
        game_over = False

        pygame.time.set_timer(self.HORIZONTAL_SPAWN_EVENT,
Config['simulator']['spawn_rate']['slow'])
        pygame.time.set_timer(self.VERTICAL_SPAWN_EVENT,
Config['simulator']['spawn_rate']['slow'])

        while not game_over:

            for event in pygame.event.get():
                if event.type == self.HORIZONTAL_SPAWN_EVENT:
                    rate = self.background_ctrl.get_spawn_rate(DoubleLane.Horizontal)
                    pygame.time.set_timer(self.HORIZONTAL_SPAWN_EVENT,
Config['simulator']['spawn_rate'][rate])
                    self.spawn(DoubleLane.Horizontal)

                if event.type == self.VERTICAL_SPAWN_EVENT:
                    rate = self.background_ctrl.get_spawn_rate(DoubleLane.Vertical)
                    pygame.time.set_timer(self.VERTICAL_SPAWN_EVENT,
Config['simulator']['spawn_rate'][rate])
                    self.spawn(DoubleLane.Vertical)

                if event.type == pygame.QUIT:
                    game_over = True

                if event.type == pygame.MOUSEBUTTONDOWN:
                    for double_lane in [DoubleLane.Horizontal, DoubleLane.Vertical]:
                        for rate in ['slow', 'medium', 'fast']:
                            if
self.background_ctrl.spawn_rate_buttons[double_lane][rate].collidepoint(event.pos):
                                self.background_ctrl.set_spawn_rate(double_lane, rate)
                    # if self.background_ctrl.fuzzy_button.collidepoint(event.pos):
                    #     moving_averages =
self.vehicle_ctrl.get_moving_averages_num_vehicles_behind_traffic()
                    #     print(self.calculate_fuzzy_score(moving_averages))

            self.background_ctrl.refresh_screen()
            self.background_ctrl.draw_road_markings()
            self.background_ctrl.draw_vehicle_count(self.vehicle_ctrl.counter)
            self.background_ctrl.draw_spawn_rate_buttons()
```

```
            self.background_ctrl.draw_light_durations(self.traffic_ctrl.get_green_light_extension())

            # print(self.traffic_ctrl.get_green_light_remaining())

            self.traffic_ctrl.update_and_draw_traffic_lights()
            self.vehicle_ctrl.destroy_vehicles_outside_canvas()
            self.vehicle_ctrl.update_and_draw_vehicles()
            self.vehicle_ctrl.update_num_vehicles_behind_traffic()

            if round((time.time() - self.start_time), 1) % Config['simulator']['static_duration'] == 0:
                self.moving_averages =
self.vehicle_ctrl.get_moving_averages_num_vehicles_behind_traffic()
            self.background_ctrl.draw_moving_averages(self.moving_averages)

            current_green_light_remaining_time = self.traffic_ctrl.get_green_light_remaining()
            direction_changed = current_green_light_remaining_time >
self.green_light_remaining_time
            self.green_light_remaining_time = current_green_light_remaining_time

            if not self.is_extended:
                if current_green_light_remaining_time <=
Config['simulator']['seconds_before_extension']:
                    fuzzy_score = self.calculate_fuzzy_score(self.moving_averages)
                    self.horizontal = self.moving_averages[Lane.left_to_right]
                    self.vertical = self.moving_averages[Lane.top_to_bottom]
                    self.background_ctrl.draw_fuzzy_score(fuzzy_score,
self.traffic_ctrl.get_current_active_lane())
                    self.traffic_ctrl.set_green_light_extension(fuzzy_score)
                    self.is_extended = True
                    self.extension_notification_start_time = time.time()
                    self.green_light_remaining_time = self.traffic_ctrl.get_green_light_remaining()
            else:
                if direction_changed:
                    self.traffic_ctrl.clear_all_green_light_extension()
                    self.is_extended = False

            if time.time() - self.extension_notification_start_time <
Config['simulator']['fuzzy_notification_duration']:

self.background_ctrl.draw_extension_notification(self.traffic_ctrl.get_green_light_extension(),
self.horizontal, self.vertical)

            pygame.display.update()
            self.clock.tick(Config['simulator']['frame_rate'])

    def calculate_fuzzy_score(self, moving_averages):
        traffic_state = self.traffic_ctrl.get_current_active_lane()
```

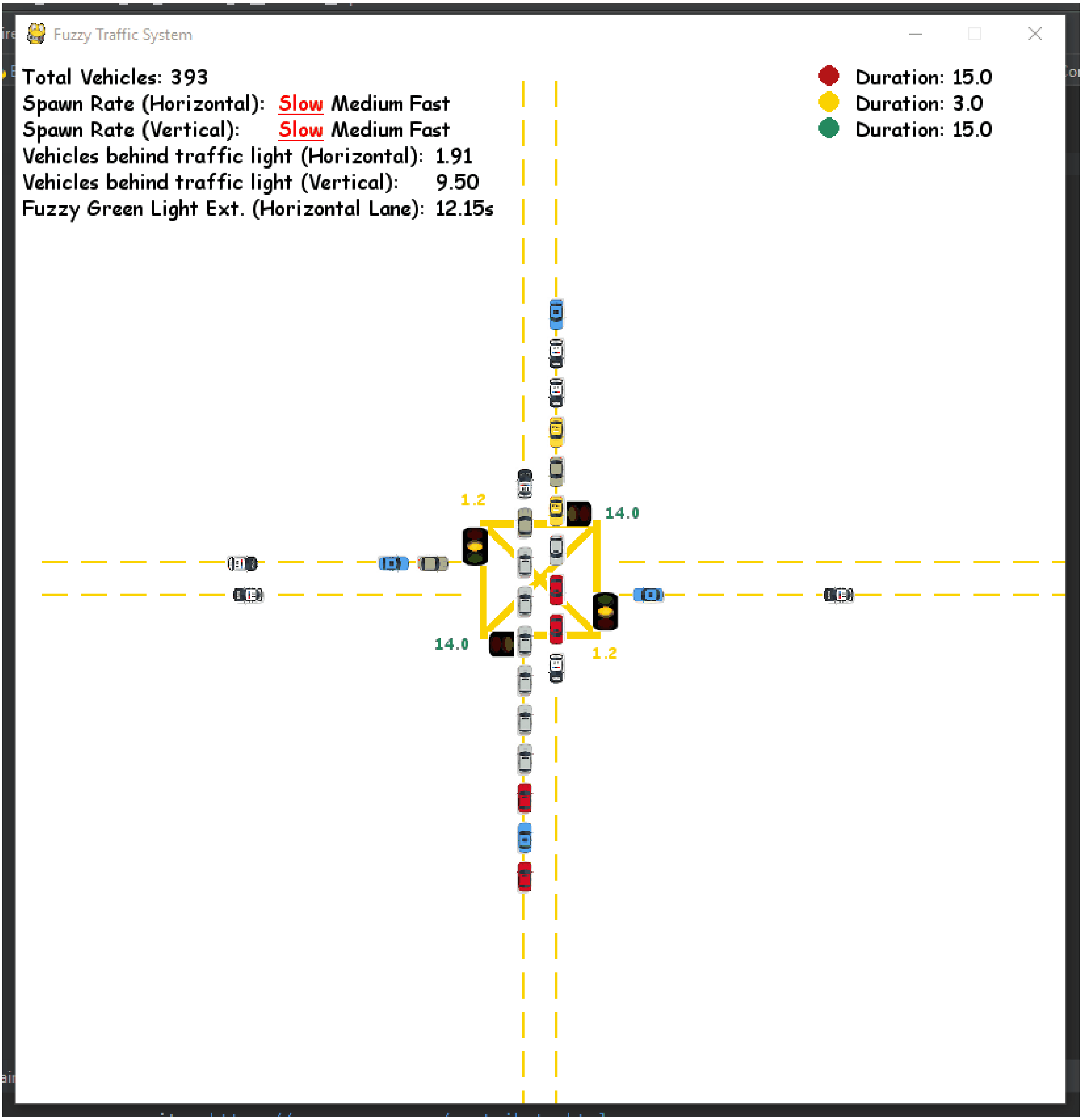
```
        if self.is_extended :
            ext_count = 1
        else:
            ext_count =0

        if traffic_state == DoubleLane.Vertical:
            return self.traffic_ctrl.calculate_fuzzy_score(moving_averages[Lane.top_to_bottom],
moving_averages[Lane.left_to_right], ext_count)
        elif traffic_state == DoubleLane.Horizontal:
            return self.traffic_ctrl.calculate_fuzzy_score(moving_averages[Lane.left_to_right],
moving_averages[Lane.top_to_bottom], ext_count)

    def initialize(self):
        self.spawn(DoubleLane.Horizontal)
        self.spawn(DoubleLane.Vertical)
        # self.toggle_traffic()

    def start(self):
        pygame.init()
        pygame.display.set_caption(self.caption)

        self.initialize()
        self.main_loop()

        pygame.quit()
        quit()
```

Main Code:

```
from src.Simulator import Simulator
if _name_ == "_main_":
    simulator = Simulator('Fuzzy Traffic System')
    simulator.start()
```

CHAPTER

SCREENSHOTS AND RESULTS



CHAPTER

CONCLUSION AND FUTURE ENHANCEMENTS

Future Enhancements:

Optimization Algorithms:

- Incorporate advanced optimization algorithms to dynamically adjust traffic light timings based on real-time traffic data, considering factors such as traffic density, flow rates, and historical patterns.

Machine Learning Integration:

- Explore the integration of machine learning techniques to improve the adaptability and predictive capabilities of traffic control systems.
- Use machine learning models to learn from historical traffic data and predict future traffic patterns, enhancing the proactive nature of the control strategies.

Smart Sensor Networks:

- Implement smart sensor networks and Internet of Things (IoT) devices to gather more comprehensive and accurate real-time traffic information.
- Utilize data from cameras, vehicle detectors, and other sensors to enhance the granularity and precision of input data for the control systems.

Dynamic Adjustment Mechanisms:

- Develop mechanisms for dynamically adjusting fuzzy logic parameters and rules based on feedback from real-time traffic conditions.
- Implement self-learning algorithms that continuously improve the performance of the control systems over time.

Integration with Smart City Initiatives:

- Integrate fuzzy logic-based traffic control systems with broader smart city initiatives, such as intelligent transportation systems and urban planning strategies.
- Collaborate with city authorities and transportation agencies to deploy and evaluate the effectiveness of these systems in real-world urban environments.

In conclusion, while fuzzy logic-based traffic control systems offer promising solutions for managing congestion and optimizing traffic flow, continuous research and development efforts are needed to address challenges, enhance performance, and realize the full potential of these systems in creating smarter and more efficient transportation network.