Project Report
On

Adaptive Traffic Light Control System

Engineering Clinics – Multi Disciplinary Project

BACHELOR OF TECHNOLOGY

(Artificial Intelligence (AI) and Data Sciences)

SUBMITTED BY:
Bhoomi Chauhan (2420694)
Bhoomika Rana (2420695)
Bhumika (2420696)
Bitu Sahu (2420697)
Nov 2025

Under the Guidance of
Er. Shubham Sharma J4224
(Assistant Professor)

Department of Computer Science & Engineering
Chandigarh Engineering College Jhanjeri, Mohali - 140307

Table of Contents

Chapter 1: Introduction

1.1 Main Idea of the Project

Controlling and managing traffic signals at to ensure vehicular traffic safety and a constant traffic flow is difficult for any dense urban area. The excessive number of vehicles transitioning from one place to another and the absence of proper flow management causes an interruption in the smooth traffic flow, resulting in lousy traffic congestion and inappropriate traffic control [1]. According to [2], fixed traffic signal (FTS) latencies account for about 10% of worldwide traffic delays. During the initial days of urbanization, many government agencies realized the need for traffic control at RIs and deployed old-age manual traffic signalized systems. In the manual Traffic Signal Control (TSC) system, the traffic officers modify the duration of the yellow, red, and green signals according to the traffic volume. This mode aims to achieve a smooth flow of traffic and safe passage for road commuters. Today, traffic light synchronization controls traffic at major RIs by providing the same amount of green signal time to all directions during the one cycle known as FTS. Various papers [3,4] suggested that developed countries suffer over 295 million traffic hours of delay with Fixed Traffic System. Thus, detailed estimations of time-dependent delays are needed for better traffic control and management.

Urban traffic congestion is one of the most pressing challenges faced by modern cities, leading to increased travel times, fuel consumption, and environmental pollution. To address these issues, traffic management systems need to be more adaptive and intelligent. Our project introduces an Adaptive Traffic Light Control System (ATLCS) that synchronizes consecutive traffic lights by applying dynamic delays between green signals in a given direction. These delays are updated in real time based on the number of vehicles waiting at each junction. This approach enables vehicles, particularly those leaving the city center, to cover longer distances with fewer stops, thereby reducing the "stop-and-go" effect and improving travel efficiency.

Performance evaluations of the ATLCS have shown promising results. In the synchronized direction, the average travel time of vehicles was reduced by up to 39% compared to conventional non-synchronized fixed-time traffic light systems. Across the entire simulated road network, the overall improvement achieved was 17%.

The broader context of this project lies in the vision of the Smart City, which seeks to create sustainable ecosystems while promoting citizen welfare and economic development. Smart Cities rely on advanced information and communication technologies (ICTs) to monitor and manage critical urban assets such as energy, water, and transportation. Realizing this vision requires collaboration among governments, industries, academia, and communities. Furthermore, enabling technologies such as the Internet of Things (IoT), 5G, cloud computing, artificial intelligence, and connected vehicles play a vital role in achieving sustainable advancements, including smart buildings, renewable energy systems, and intelligent green transportation.

This project contributes to that vision by presenting an innovative traffic management solution that supports sustainable urban mobility, improves travel efficiency, and enhances the overall quality of life in cities.

1.2 Objectives

The primary objective of this project is to develop an Adaptive Traffic Control System (ATCS) capable of optimizing traffic signal operations in real time. By continuously analysing the current traffic conditions and dynamically modifying signal timings, the system aims to reduce congestion, minimize vehicular delays, and lower overall environmental impact at urban intersections. Unlike conventional fixed-time or semi-actuated control systems, which operate on pre-defined cycles and fail to respond effectively to fluctuating traffic volumes, the ATCS intelligently adjusts its operation based on live data [5]. This results in smoother traffic movement, reduced waiting periods, and a more efficient utilisation of existing road infrastructure.

1. Real-time Traffic Monitoring

This objective focuses on collecting accurate and continuous traffic data from various sources such as sensors, cameras, and historical datasets. The monitoring system measures key parameters including vehicle density, queue lengths, lane occupancy, and flow rates. By detecting peak and off-peak hours, traffic surges, and unusual patterns, the ATCS can understand the real-time conditions of each intersection. This enables the system to react promptly and take data-driven decisions.

2. Dynamic Signal Timing

A core function of the ATCS is to dynamically adjust the duration of green, yellow, and red signals based on real-time traffic demands. Instead of relying on static timing plans, the system evaluates current conditions and allocates signal time proportionally to the traffic load on each approach. Additionally, it synchronizes neighbouring intersections to maintain a “green corridor,” reducing the chances of bottlenecks and unnecessary halts. This coordinated approach ensures that traffic moves efficiently across larger road networks.

3. Reduction of Travel Time and Congestion

Through adaptive algorithms—such as machine learning models, rule-based logics, or traffic optimisation techniques—the system aims to minimize delays for motorists. By reducing queue lengths, balancing traffic loads, and prioritizing heavily congested lanes when required, the ATCS improves overall travel time reliability. The reduction in stop-and-go traffic not only enhances driver convenience but also increases the effective road capacity without requiring physical infrastructure expansion.

4. Environmental and Economic Benefits

By decreasing idle time at intersections and minimizing unnecessary stopping, the ATCS contributes to lower vehicular emissions such as CO_2, NOx, and particulate matter. Reduced fuel consumption leads to economic benefits for both individual drivers and the broader community. Over time, improved traffic flow can also reduce maintenance costs for road infrastructure and support sustainable urban development.

Chapter 2: System Requirements

The implementation of the Adaptive Traffic Control System (ATCS) prototype relies on a set of essential hardware components that work together to simulate real-time traffic signal control. Each component plays a specific role in enabling data processing, signal switching, and safe circuit operation. The following hardware elements are used in the prototype:

• Arduino Uno

The **Arduino Uno** serves as the central processing unit of the system. Acting as the main controller, it receives input signals, runs the adaptive traffic control algorithm, and manages the timing and sequencing of the traffic lights. Its ATmega328P microcontroller provides sufficient computational power and memory for real-time decision-making in small-scale simulations. The Uno's simplicity, reliability, USB programmability, and extensive support libraries make it ideal for rapid prototyping of embedded control systems like ATCS.

• LEDs (Red, Yellow, Green)

Three **Light Emitting Diodes (LEDs)** represent the standard traffic signal states.

- The **Red LED** indicates a complete stop for vehicles.
- The **Yellow LED** warns drivers to slow down or prepare to stop.
- The **Green LED** will allow traffic to move. These LEDs provide a clear and visual demonstration of how the adaptive algorithm updates signal durations based on simulated traffic conditions. They serve as the primary output indicators of the system.

• 100-Ohm Resistors (×3)

Each LED is connected in series with a **100-ohm resistor** to limit the current flowing through it. Without current-limiting resistors, LEDs can draw excessive current from the Arduino pins, leading to overheating or damage. The resistors ensure safe operation of the LEDs by maintaining stable brightness levels and preventing short circuits. Using three resistors—one for each LED—ensures consistent protection across all traffic light outputs.

• Connecting Wires

A set of **male-to-male or male-to-female jumper wires** is used to establish electrical connections between the Arduino, LEDs, resistors, and the breadboard. These wires provide flexibility for modifying connections, testing improved designs, and rearranging components during the development phase. Their reusability makes them highly suitable for iterative prototype building.

• Breadboard

The **breadboard** provides a solderless, modular platform for assembling the entire circuit. It allows components to be easily inserted, removed, or repositioned without permanent soldering. This flexibility is crucial during testing and troubleshooting, especially when experimenting with different signal timing logic or adding additional sensors in future enhancements. The breadboard ensures clean, organized wiring and supports efficient prototyping of the ATCS circuitry.

Software Requirements:

- **Arduino IDE**

The **Arduino Integrated Development Environment (IDE)** serves as the primary software platform for the development, testing, and deployment of the Adaptive Traffic Control System program. It provides a user-friendly interface that allows developers to write, edit, compile, and upload code directly to the Arduino Uno microcontroller. The IDE supports the C/C++ programming language, making it flexible for implementing logical conditions, timing operations, and adaptive traffic signal algorithms.

One of the key advantages of the Arduino IDE is its built-in library support. These libraries simplify the process of controlling hardware components such as LEDs, sensors, and timers, allowing developers to focus on system logic rather than low-level programming. The IDE also includes an integrated compiler that checks for syntax errors and converts the written code into machine-readable instructions that the microcontroller can execute.

In addition to coding and compilation, the Arduino IDE features a **Serial Monitor** and **Serial Plotter**, which are essential tools for debugging and real-time analysis. Using the Serial Monitor, developers can observe output messages, track variable values, and verify the sequence of signal changes. This functionality helps identify logical errors, timing issues, or unexpected behaviour in the traffic control algorithm. The Serial Plotter can be used to visualize data patterns, making it useful for analysing simulated traffic inputs or performance trends.

The IDE supports seamless communication with the Arduino through USB connectivity, enabling quick uploads and immediate testing of updated algorithms. This iterative workflow is particularly important for adaptive systems, where multiple refinements and performance evaluations are required. Furthermore, the open-source nature of the Arduino IDE ensures access to extensive online resources, community support, and example codes, making the development process more efficient and well-guided.

Chapter 3 System Requirements Analysis

The System Requirements Analysis outlines the essential purpose, functionalities, and performance expectations of the Adaptive Traffic Control System (ATCS) software. This chapter ensures that the system behaves predictably, efficiently, and in accordance with the intended design objectives. The requirements listed below guide the development, testing, and refinement of the software to ensure accurate simulation of adaptive traffic signal behaviour.

1. Purpose of the Software

The primary purpose of the ATCS software is to manage and control the sequence of traffic lights—Red, Yellow, and Green—based on adaptive logic that simulates real-time traffic variations. Unlike traditional fixed-time traffic systems that operate using predefined intervals, the software incorporates dynamic decision-making to mimic how modern adaptive systems respond to fluctuating traffic conditions.

The software ensures smooth and coordinated traffic flow by regulating signal timings, thereby minimizing unnecessary waiting periods and reducing simulated congestion. Through the implemented adaptive algorithm, the software demonstrates how traffic control systems can intelligently adjust their operations to accommodate changing traffic density, vehicle queues, or peak-hour demands.

Additionally, the software serves as a learning and testing platform for understanding the behaviour of adaptive systems. It allows users to observe how the Arduino processes commands and updates signal states in real time, making it useful for prototyping, experimentation, and educational demonstrations.

2. Functional Requirements

The functional requirements define what the software must do to achieve the objectives of an adaptive traffic control simulation. These include:

• Control the LEDs to represent traffic signal changes

The software must manage the ON/OFF states of the Red, Yellow, and Green LEDs. Each LED corresponds to a specific traffic signal and must illuminate according to the logic defined in the program.

• Implement timing logic for Red, Yellow, and Green lights

The system should include timing mechanisms to control how long each signal remains active. These timings may follow predefined intervals or adjust based on conditions representing simulated traffic inputs.

• Allow dynamic adjustment of signal duration to simulate adaptive control

A key requirement is the ability to modify the duration of each light dynamically. The software should adjust signal timings based on logic such as increased "traffic load," simulated sensor input, or conditions defined within the program.

• Monitor system execution and provide debugging feedback

Through the Serial Monitor, the software should display real-time messages that indicate current signal states, timing changes, or algorithm decisions. This feature aids in troubleshooting, validating system responses, and analysing the behaviour of the adaptive algorithm.

3. Non-Functional Requirements

Non-functional requirements describe the quality attributes of the system. These characteristics ensure that the software is dependable, efficient, and easy to work with.

• *Reliability*

The software must operate smoothly without unexpected crashes, glitches, or inconsistent LED behaviour during execution. Reliable operation is essential for accurately demonstrating adaptive traffic control concepts.

• Efficiency

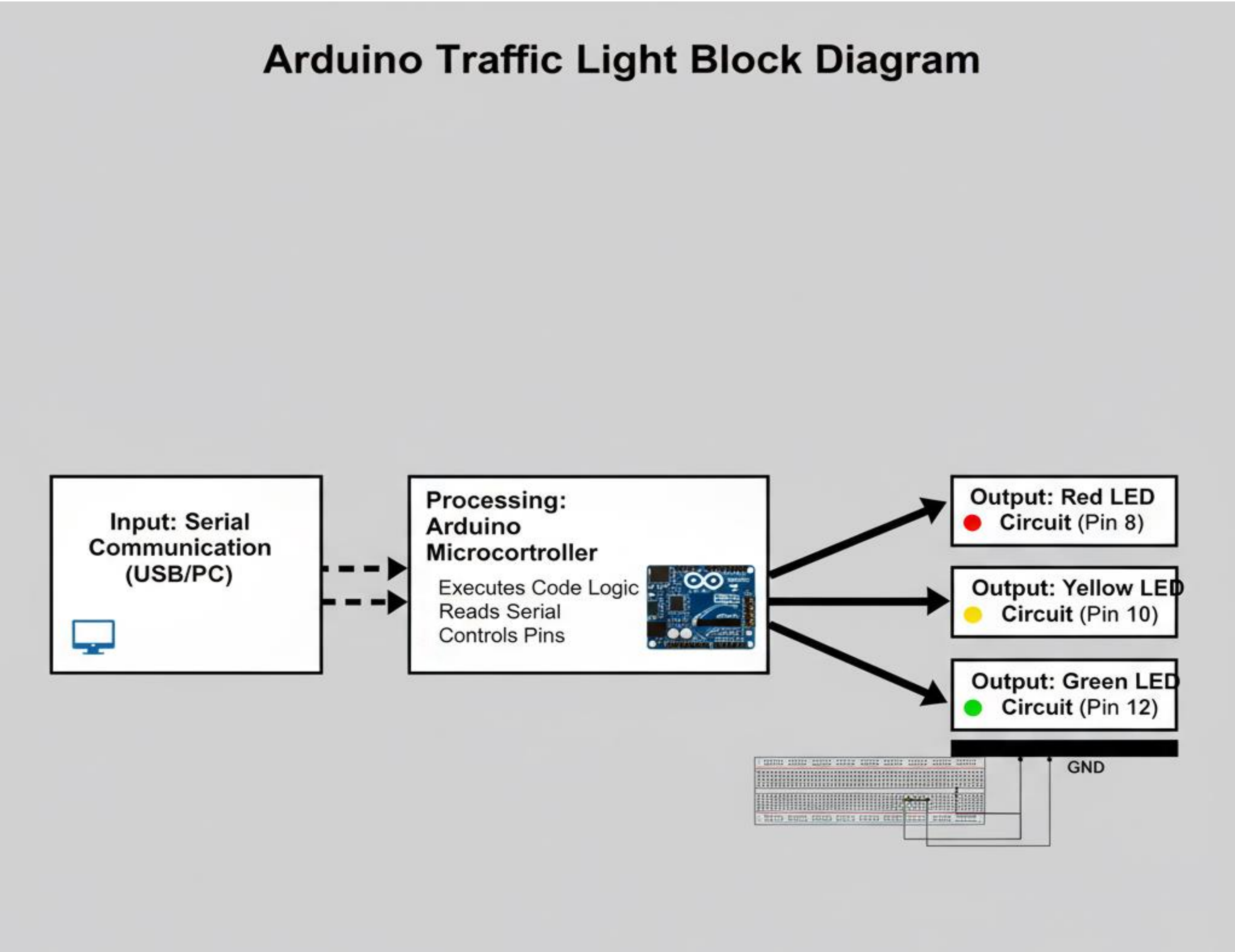
The system should react quickly and correctly to simulated traffic conditions or algorithmic changes. Efficient processing guarantees that timing adjustments occur without noticeable delay, closely resembling real-world adaptive control systems.

• Usability

The software should be simple for users to upload, modify, and understand through the Arduino IDE. Clean code structure, clear comments, and adjustable parameters improve its usability, especially for testing new traffic scenarios or upgrading the adaptive logic.

Chapter 4: Circuit Design

Block Diagram:



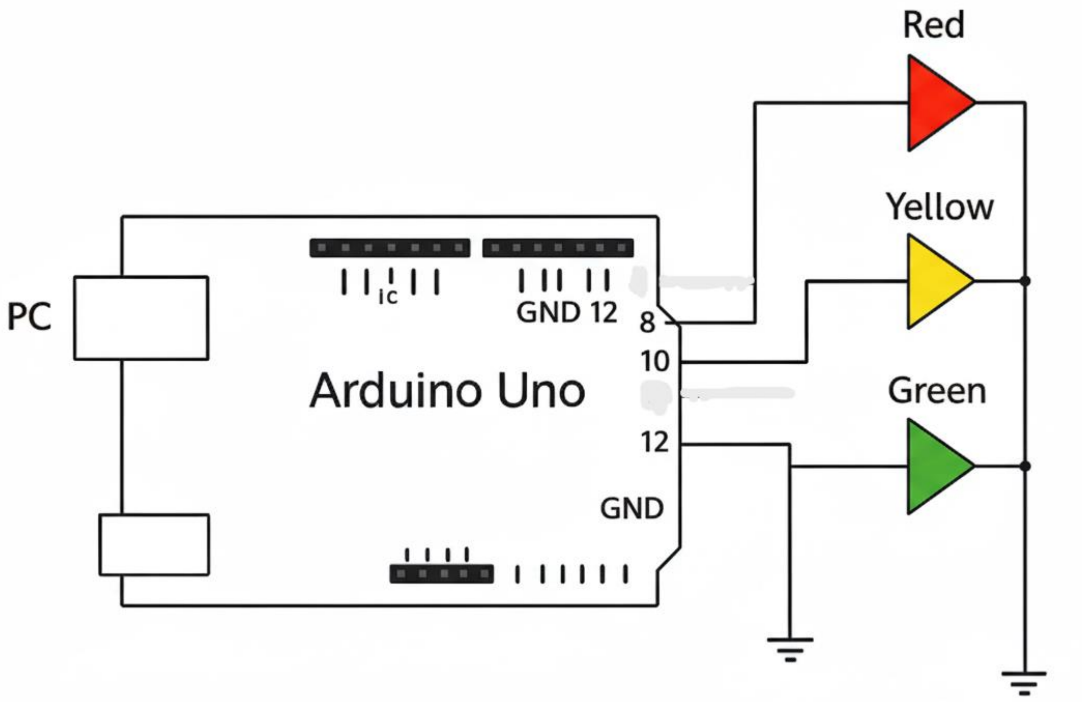
The block diagram illustrates the overall functioning of the Arduino-based traffic light system and shows how input, processing, and output components interact within the model. The system begins with the input stage, where commands are sent from a computer to the Arduino through serial communication via a USB connection. These commands determine which traffic light phase—red, yellow, or green—should be activated.

The processing stage is handled by the Arduino microcontroller. After receiving a command through the serial interface, the Arduino interprets the input, executes the corresponding section of the program, and controls the appropriate output pins. This stage forms the core of the system, as it contains the programmed logic responsible for switching between different traffic light signals.

The output stage consists of three separate LED circuits connected to designated Arduino pins. The red LED is controlled through pin 8, the yellow LED through pin 10, and the green LED through pin 12. When the microcontroller activates a specific pin, the corresponding LED lights up to represent the chosen traffic signal state. Each LED is connected to the breadboard with resistors and is grounded to complete the circuit.

Overall, the block diagram provides a simplified representation of how serial input is processed by the Arduino and translated into visual traffic signals through the connected LEDs. It clearly shows the flow of information from input to processing and finally to output, demonstrating the structure and functionality of the implemented traffic light model.

Circuit Diagram:



Arduino Traffic Light Schematic Diagram (Controlled via PC Serial)

This diagram illustrates the complete electronic circuit for an Arduino-controlled traffic light system, designed to be operated via external serial commands. The system is cantered around the **Arduino Uno** microcontroller, which acts as the processing unit. The visual output is provided by three **LEDs** (Red, Yellow, and Green), which are directly tied to specific digital output pins: **Pin 8 (Red)**, **Pin 10 (Yellow)**, and **Pin 12 (Green)**, as defined in the associated Arduino code. The critical function of the circuit is facilitated by three 3 100ohm resistors, one placed in series with each LED to limit the current flow, protecting both the LEDs from burnout and the Arduino's output pins from excessive load. All three LED circuits are completed by connecting the cathodes (negative leads) to the common **Ground (GND)** pin on the Arduino. The system is made interactive by connecting the Arduino to a **PC** via USB, which serves two roles: providing power and enabling **serial communication**. The PC, running a script like Python (as shown in your supplementary files), sends single-character commands ('R', 'Y', or 'G') to the Arduino. The Arduino's loop() function constantly monitors the serial port, and upon receiving a command, it momentarily lights up the corresponding LED for the specified duration (e.g., 5 seconds for Red/Green, 2 seconds for Yellow), thereby simulating a controlled traffic light sequence.

Chapter 5: Implementation

The Arduino IDE codes used in the working of the LEDs using the breadboard, Arduino UNO and resistor is given below.

sketch_oct15a > sketch_oct15a.ino

```
const int red = 8;
const int yellow = 10;
const int green = 12;

void setup() {
  pinMode(red, OUTPUT);
  pinMode(yellow, OUTPUT);
  pinMode(green, OUTPUT);
  Serial.begin(9600);  // enable serial communication
}

void loop() {
  if (Serial.available()) {
    char cmd = Serial.read();

    if (cmd == 'R') {
      digitalWrite(red, HIGH);
      delay(5000);
      digitalWrite(red, LOW);
    }

    else if (cmd == 'Y') {
      digitalWrite(yellow, HIGH);
      delay(2000);
      digitalWrite(yellow, LOW);
    }

    else if (cmd == 'G') {
      digitalWrite(green, HIGH);
      delay(5000);
      digitalWrite(green, LOW);
    }
  }
}
```

The Arduino code is designed to control three LEDs representing the red, yellow, and green phases of a traffic signal. At the beginning of the program, the digital pins connected to the LEDs are defined, ensuring clarity and ease of modification. In the setup function, each of these pins is configured as an output, and serial communication is initialized so the Arduino can receive commands from the Serial Monitor.

The main logic of the program runs inside the loop function, where the system continuously checks whether any input has been received through the serial interface. When a command is detected, a single character is read and used to determine which traffic light should activate. If the command received is ‘R’, the red LED turns on for a fixed duration and then turns off, simulating the stop phase. When the command is ‘Y’, the yellow LED is activated briefly to represent the caution phase. Similarly, a ‘G’ command triggers the green LED for a defined period, simulating the go signal.

Each command activates only one LED at a time, and the use of delay functions ensures that the LEDs remain on for the appropriate amount of time before turning off. This simple control structure demonstrates how traffic light phases can be managed through serial input and digital output, forming the foundation of a basic adaptive traffic signal simulation.

A Python code implemented for the project model. This Python script is used to control the Arduino-based traffic light system by sending commands through serial communication. It establishes a connection between the computer and the Arduino and then repeatedly sends signals that activate the green, yellow, and red LEDs in sequence.

The program begins by importing the required libraries: serial for communication with the Arduino and time for implementing delays. A serial connection is then created using the appropriate COM port, baud rate, and timeout value. A brief delay is added to allow the Arduino to initialize properly after the connection is established.

A function named send_signal() is defined to simplify sending commands to the Arduino. This function takes a single character (‘G’, ‘Y’, or ‘R’), encodes it, and transmits it over the serial port. It also prints a confirmation message indicating which signal was sent.

Inside the infinite while True loop, the program cycles through the three traffic light phases. First, it sends the command for the green light and keeps it active for a few seconds using time.sleep(). It then sends the yellow light command with a shorter delay, followed by the red light command with its corresponding duration. Each phase is accompanied by a print statement to display the current light status on the console.

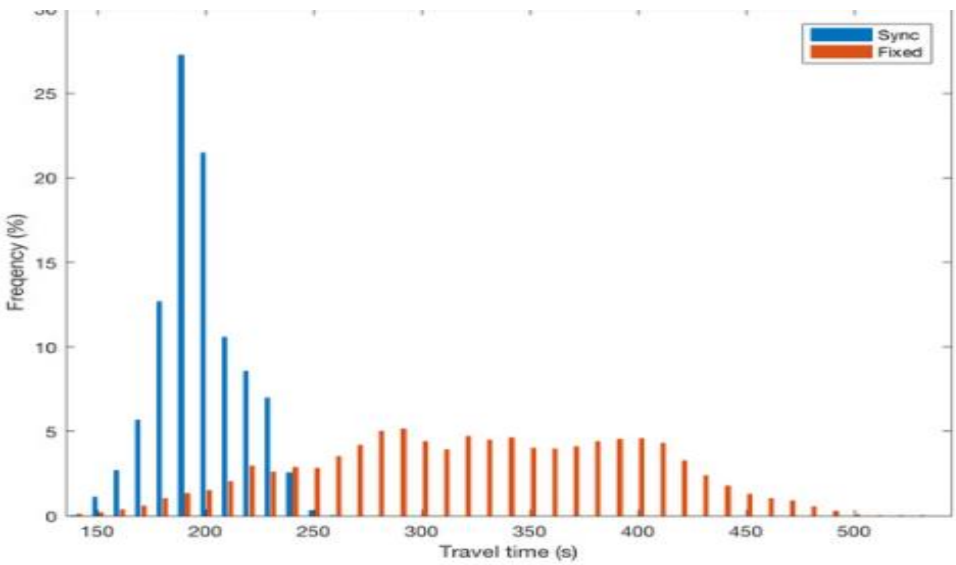
```
project.py > ...
import serial
import time

# Change 'COM3' to your Arduino's COM port
arduino = serial.Serial(port='COM9', baudrate=9600, timeout=1)

time.sleep(2)  # Wait for Arduino to initialize

def send_signal(signal):
    arduino.write(signal.encode())  # send one character command
    print(f"Signal sent: {signal}")

while True:
    # Green light
    send_signal('G')
    print("🟢 Green ON")
    time.sleep(5)

    # Yellow light
    send_signal('Y')
    print("🟡 Yellow ON")
    time.sleep(2)

    # Red light
    send_signal('R')
    print("🔴 Red ON")
    time.sleep(5)
```

Chapter 6: Result and Discussion

The proposed method reduced vehicle waiting time by 4 seconds, achieving around a 10% overall improvement. These results show that the new time management equation is more efficient than the fixed-time approach used in current systems, as it allocates signal time based on the number of vehicles and applies different weights to six distinct vehicle categories. By distributing time more intelligently according to traffic composition, the proposed formula consistently results in shorter waiting times compared to traditional fixed-time traffic signal systems. This improvement contributes to smoother traffic flow and fewer unnecessary stops at intersections while also helping reduce fuel consumption and emissions by minimizing idle time. Overall, the proposed system demonstrates clear advantages in both operational efficiency and traffic management effectiveness [8]. Moreover, An Adaptive Traffic Control System (ATCS) helps reduce collisions by creating smoother, more predictable, and safer traffic movement at intersections. By adjusting signal timings based on real-time traffic conditions, ATCS minimizes sudden stops, reduces congestion, and lowers the chances of vehicles engaging in risky maneuvers such as abrupt braking or signal jumping. Coordinated and well-timed green phases decrease the stop-and-go pattern, which is one of the major contributors to rear-end and side-impact crashes. ATCS also improves pedestrian safety by providing clearer and more consistent crossing opportunities. Through better flow management, reduced conflict points, and timely signal responses, an adaptive system significantly lowers the overall likelihood of collisions, injuries, and fatalities at busy urban intersections [9].

Figure A depicts the variation of the achieved trips duration (i.e., travel time) in fixed and synchronized TLCSs We can observe that our synchronized ATLCS has much higher number of shorter trips compared to the fixed time TLCS which has a significantly higher number of longer trips. This is due to the fact that the synchronization process at roads allows a large portion of vehicles to reach their destination with lower number of stops (i.e., a reduction in the ‘stop and go’ phenomenon), hence the faster progress towards their destinations.

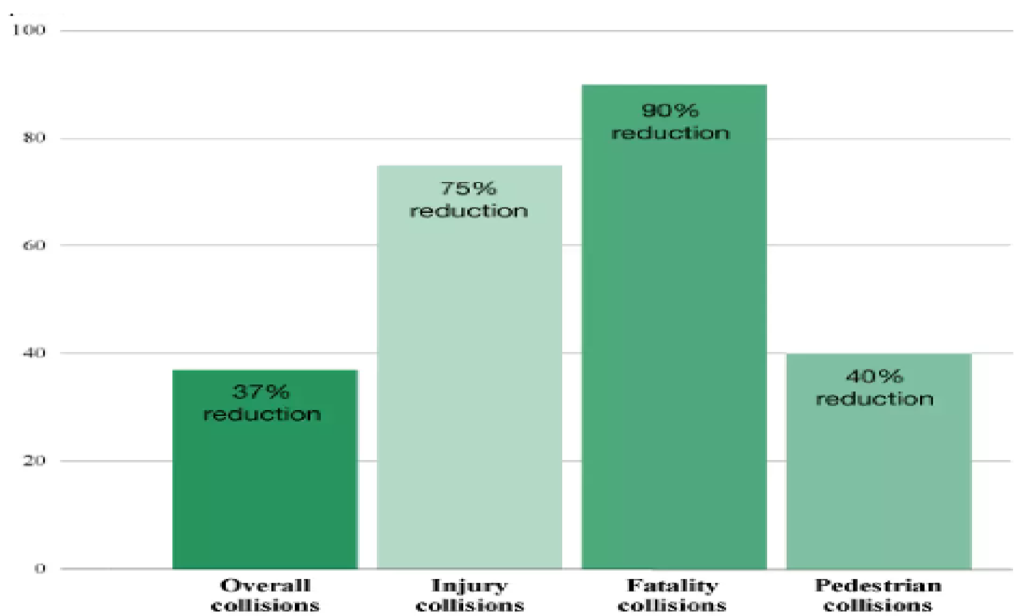
Figure A: Time vs Frequency graph of vehicles travel showing fixes traffic light control system and adaptive traffic light control system.



The graph below highlights the strong safety benefits of adaptive traffic control systems. It shows a clear reduction in overall, injury-related, fatal, and pedestrian collisions when adaptive strategies are used. By improving signal coordination, reducing sudden stops, and creating smoother traffic flow, these systems help prevent crashes and make intersections safer for both drivers and pedestrians. Overall, the results demonstrate that adaptive traffic control can significantly enhance road safety and reduce collision risks. The graph highlights the strong safety benefits of adaptive traffic control systems. It shows a clear reduction in overall, injury-related, fatal, and pedestrian collisions when adaptive strategies are used.

Figure B depicts the significant impact of adaptive and improved traffic control measures on reducing different types of collisions. Overall collisions show a 37% reduction, indicating a noticeable improvement in general road safety. Injury-related collisions experience an even greater decrease of 75%, demonstrating the system's effectiveness in minimizing crash severity. The most substantial improvement is seen in fatality collisions, which show a remarkable 90% reduction, highlighting the potential of adaptive systems to save lives. Pedestrian collisions are reduced by 40%, reflecting safer crossing conditions and better signal coordination for vulnerable road users. Overall, the data clearly demonstrates the strong safety benefits achieved through modern, adaptive traffic control strategies.

Figure B: Shows Data of the Number of Collisions reduction after the successful implementation of Adaptive Traffic Lights Control System on roads.



Chapter 7: Conclusion and Future Scope

7.1 Conclusion

The Adaptive Traffic Control System (ATCS) project effectively demonstrates the fundamental operation of an intelligent traffic signal using an Arduino Uno, a breadboard, and a set of three LEDs representing the standard red, yellow, and green lights. Through the implemented sequence, the system replicates real-world traffic light behaviour, including a fully functional stop–caution–go cycle and a blinking green phase that alerts users of an upcoming signal change. This simulation not only reflects the basic functioning of traditional traffic lights but also introduces early elements of adaptive control, a concept increasingly important in modern traffic management.

The project illustrates how adaptive traffic systems can reduce unnecessary waiting times, improve the overall movement of vehicles, and enhance the safety of road users by providing predictable and timely transitions between signal states. The use of adjustable timings within the program demonstrates how traffic signals can be adapted to changing traffic conditions, even though this prototype operates on present durations. By showcasing these capabilities, the project highlights the value and potential impact of implementing intelligent traffic control mechanisms in urban areas.

Future enhancements could include integrating sensors for real-time vehicle detection, using machine learning algorithms to predict traffic patterns, establishing communication between multiple intersections, or connecting the system to smart city infrastructure for centralized monitoring and control. These additions would transform the basic prototype into a fully adaptive network capable of addressing real-world congestion challenges more effectively.

In summary, the project demonstrates the feasibility, functionality, and advantages of adaptive traffic control at a small scale. It serves as a practical starting point for deeper exploration into intelligent transportation systems and paves the way for innovative solutions that could significantly contribute to efficient, sustainable, and safer urban traffic management in the future.

7.2 Future Scope

The Adaptive Traffic Control System (ATCS) project provides a strong foundation for the development of more advanced, scalable, and real-world traffic management solutions. While the current prototype focuses on demonstrating the basic functioning of a single adaptive signal, several meaningful enhancements can significantly expand its capabilities and relevance. Potential future developments include the following:

1. Integration with Sensors:

   The system can be upgraded by incorporating various real-time sensors such as ultrasonic sensors, infrared sensors, inductive loop detectors, or camera-based vision systems. These inputs would allow the ATCS to detect vehicle presence, traffic density, and queue lengths at an intersection. With accurate real-time data, the controller could automatically adjust signal timings to manage changing traffic conditions more efficiently. This would enable a truly dynamic and responsive traffic control mechanism similar to those used in modern intelligent transportation systems.

2. Smart City Integration:

   A major advancement would involve connecting the ATCS to broader smart city infrastructure. Through IoT devices, wireless communication modules, and cloud platforms, the system can support centralized traffic monitoring and control across multiple intersections. Integration with GPS-based vehicle tracking and traffic management dashboards can provide real-time analytics, congestion visualization, and data-driven decision-making. Such connectivity would transform the ATCS from an isolated prototype into a component of a city-wide intelligent traffic network.

3. Emergency Vehicle Prioritization:

   Future versions of the ATCS can incorporate priority control for emergency vehicles such as ambulances, fire trucks, or police vehicles. Using RF modules, GPS signals, or dedicated communication systems, the traffic signal could detect an approaching emergency vehicle and immediately switch to green to provide a clear and uninterrupted path. This feature enhances public safety and significantly reduces emergency response times.

4. Pedestrian-Friendly Signals:

   The system can be extended to include pedestrian crossing mechanisms. By using pedestrian push buttons, motion sensors, or foot-traffic detection systems, the ATCS could allocate crossing time dynamically based on pedestrian demand. This ensures safer crossings, reduces waiting times for pedestrians, and improves overall intersection usability for both vehicles and non-motorized road users.

5. Predictive Traffic Modelling:

   Another enhancement involves integrating AI and machine learning techniques to analyse historical traffic patterns and predict future congestion. By forecasting peak hours, traffic surges, or expected delays, the ATCS can adjust signal timings proactively rather than reactively. Predictive modelling can significantly reduce travel time, prevent bottlenecks, and maintain smoother traffic flow throughout the day.

Overall, the above improvements would allow the ATCS to evolve far beyond its initial educational prototype. With the integration of sensors, smart city infrastructure, emergency prioritization, pedestrian adaptation, and predictive analytics, the system could be transformed into a highly scalable and practical solution for modern urban traffic management challenges.

REFRENCES

1. A. Agrahari, MM Dhabu, PS Deshpande, A Tiwari, MA Baig, AD Sawarkar.Artificial Intelligence-based adaptive traffic signal control system: A comprehesive review. Electronics-2024.
2. Jing, P.; Huang, H.; Chen, L. An Adaptive Traffic Signal Control in a Connected Vehicle Environment: A Systematic Review. Information 2017, 8, 101.
3. Noaeen, M.; Naik, A.; Goodman, L.; Crebo, J.; Abrar, T.; Abad, Z.S.H.; Bazzan, A.L.C.; Far, B. Reinforcement Learning in Urban Network Traffic Signal Control: A Systematic Literature Review. Expert Syst. Appl. 2022, 99, 116830.
4. Tian, Y.; Liu, S.; Yan, X.; Zhu, T.; Zhang, Y. Active Control Method of Traffic Signal Based on Parallel Control Theory. IEEE J. Radio Freq. Identif. 2024, 8, 334–340.
5. DR Aleko, S Djahel - Information, 2020, An Efficient Adaptive Traffic Light Control System for Urban Road Traffic Congestion Reduction in Smart Cities.
6. SM Shinde -intelligent systems and information management, 2017 - ieeexplore.ieee.org.
7. Sai Srinivas Vellela, Manne Venkata Karthik, Golla Trividha, Lingamallu Chaithanya, Shaik Altaf. Intelligent Transportation Systems AI and Itoh for Sustainable Urban Traffic Management. Published: 27 March 2025.
8. An efficient adaptive traffic light control system for urban road traffic congestion reduction in smart cities. Dr. Aleko, S Djahel - Information, 2020 - mdpi.com.
9. An adaptive traffic lights system using machine learning. MA Ottom, A Al-Omari - Int. Arab J. Inf. Technol., 2023 - iajit.org.