SUMOFlow AI: AI-Driven Traffic Optimization for Next-Generation Smart Cities: A YOLO-SUMO Integration Approach

Team Number: 46

Team Members:

- Rana Waleed (202201737)
- Roaa Raafat (202202079)
- Mariam Alhaj (202200529)

Program: Data Science and Artificial Intelligence (DSAI)

Supervisor: Dr. Mohamed Maher

Department & School: CSAI School, Zewail City of Science, Technology and Innovation

Academic Year: 2025/2026

GitHub Repository: https://github.com/ranna-waleed/sumoflow-ai-traffic-optimization

Abstract

Urban traffic congestion represents a major challenge for modern cities, resulting in significant economic losses, increased travel times, and elevated carbon dioxide (CO_2) emissions. Traditional fixed-time traffic signal systems rely on static schedules that fail to respond to dynamic traffic conditions, leading to inefficient intersection management and unnecessary vehicle idling. This project proposes **SUMOFlow AI**, an intelligent traffic optimization system that integrates computer vision-based vehicle detection with microscopic traffic simulation.

The proposed system establishes a closed-loop control framework by combining the **SUMO (Simulation of Urban MObility)** platform with multiple deep learning-based object detection models, including **YOLOv8, Fast R-CNN, and Faster R-CNN**. Real-time vehicle detection data is transformed into structured traffic state information, which is then processed by an optimization engine to dynamically adjust traffic signal phases via the TraCI interface. By jointly optimizing traffic efficiency and environmental impact, the system aims to reduce average waiting time, queue length, and vehicle emissions. The effectiveness of the proposed approach is validated through simulation-based experiments and comparative performance evaluation against traditional fixed-time control strategies.

---

1. Introduction

1.1 Problem Statement

Smart cities largely rely on inefficient, fixed-time traffic lights that operate on historical averages rather than real-time conditions. This rigidity results in traffic congestion, unpredictable travel times, and significant economic loss. Furthermore, idling vehicles at poorly managed intersections contribute heavily to urban **Co2 emissions** and fuel wastage. The constraints of existing infrastructure make it difficult to deploy expensive sensor networks (e.g., inductive loops) at every intersection, creating a need for a vision-based, software-driven solution that can adapt to dynamic flow patterns.

1.2 Motivation

We chose this project for two main reasons: **urban efficiency** and **environmental sustainability**.

- **Stakeholders:** City traffic authorities, smart-city developers, and daily commuters benefit from reduced congestion.

- **Impact:** Solving this problem reduces the "stop-and-go" waves that cause peak emission output in vehicles. With the rapid progress in Computer Vision (YOLO) and simulation tools, it is now feasible to build a cost-effective, high-accuracy control system that does not require invasive roadwork, offering a "green" solution for next-generation cities.

1.3 Proposed Solution Overview

SUMOFlow AI is a hybrid traffic control system. It utilizes **YOLO** (You Only Look Once) to detect and count vehicles from video feeds in real-time. This detection data is transformed into a "state vector" (queue lengths, waiting times) which is fed into an optimization engine. The engine determines the optimal green-light duration and actuates the traffic lights within the **SUMO** simulation environment using the **TraCI** (Traffic Control Interface) Python API. The system creates a continuous feedback loop that learns and adapts to changing traffic densities.

---

2. Literature Review and Market Survey

2.1 Related Academic / Technical Work

A comprehensive review of existing academic research was conducted to establish the theoretical and technical foundation of SUMOFlow AI.

Ayegbusi et al. (2025) proposed a deep learning-based traffic signal control approach using YOLO for vehicle detection. Their work demonstrated the effectiveness of object detection for traffic density estimation but focused primarily on detection accuracy rather than full closed-loop signal optimization. In contrast, SUMOFlow AI extends this concept by integrating detection results directly into an adaptive signal control loop within a simulation environment.

Jahangir et al. (2024) presented a comparative evaluation of YOLOv8 and Faster R-CNN for traffic object detection. Their findings highlighted the trade-off between inference speed and detection accuracy, where YOLO achieved superior real-time performance while Faster R-CNN provided higher accuracy at increased computational cost. Based on these findings, we decided to evaluate and benchmark multiple detection models within the same system.

In addition, foundational traffic simulation principles were derived from the German Aerospace Center (DLR) SUMO documentation, which details microscopic traffic modeling, car-following behavior, lane-changing logic, and emission estimation. These principles form the basis for accurate traffic state modeling and environmental impact analysis within SUMOFlow AI.

2.2 Existing Systems and Market Solutions

1. **Giza Systems ITS (NAC):** A comprehensive system used in the New Administrative Capital. It utilizes Big Data and unified incident management but requires significant capital expenditure (CAPEX) for physical infrastructure and platform integration.

2. **Dubai RTA (UTC-UX Fusion):** A high-end system utilizing AI and predictive analytics. While powerful, it relies on heavy infrastructure (sensors, digital twins) and high operational costs.

3. **Standard Fixed-Time Systems:** The current baseline in most cities. These are low-cost but highly inefficient, unable to adapt to accidents or spontaneous congestion peaks.

2.3 Comparative Analysis

Feature	**SUMOFlow AI**	**Giza Systems ITS**	**Standard Fixed-Time**
Adaptability	High (Real-time Video)	High (Sensors/Data)	None (Static)
Cost	Low (Software/Camera	High (Infrastructure	Low

	based)	heavy)	
Deployment	Modular/Simulation-First	Complex Integration	Simple
Optimization Goal	Traffic Efficiency + **Emissions**	Traffic Flow	Safety/Order
Research Flexibility	High	Limited	None

Table I summarizes the key differences between the proposed SUMOFlow AI system, a commercial ITS solution, and a traditional fixed-time traffic control approach.

2.4 Identified Gap

Existing commercial solutions offer advanced traffic control but require expensive infrastructure and large-scale deployment, limiting accessibility and experimentation. Academic studies, while demonstrating the effectiveness of computer vision models such as YOLO, often lack full integration with traffic simulation and real-time control mechanisms. Furthermore, environmental optimization metrics such as CO_2 emissions are frequently overlooked.

SUMOFlow AI bridges this gap by providing a low-cost, simulation-based, modular framework that integrates multiple object detection models with real-time signal optimization while explicitly incorporating environmental performance metrics. The system enables safe experimentation, comparative evaluation, and scalable deployment without reliance on physical infrastructure.

---

3. Requirements Analysis

3.1 Functional Requirements

- **FR-1 Vehicle Detection:** The system must detect vehicles (cars, trucks, buses) from video frames with a confidence threshold > 0.5 using YOLOv8.

- **FR-2 Traffic Counting:** The system must count vehicles per lane and estimate queue lengths based on detection bounding boxes.

- **FR-3 Simulation Control:** The system must establish a bidirectional connection with SUMO via TraCI to query vehicle data and set traffic light phases.

- **FR-4 Optimization Logic:** The system must calculate the optimal green phase duration based on current queue lengths to minimize waiting time.

- **FR-5 Reporting:** The system must generate CSV reports detailing wait times, queue lengths, and emission levels for comparison.

3.2 Non-Functional Requirements

Performance & Accuracy

- **NFR-1 Real-Time Latency:** The detection and decision-making loop must occur within the simulation time-step limit (e.g., < 100 ms) to ensure real-time responsiveness.
- **NFR-2 AI Model Accuracy:** The YOLO model should achieve a mean Average Precision (mAP@0.5) of at least **80%** on traffic datasets to ensure reliable vehicle counts.

Modularity & Scalability

- **NFR-3 System Modularity:** The system architecture must be modular to allow for swapping detection models (e.g., replacing YOLO with Faster R-CNN) without altering the optimization engine code.
- **NFR-4 Scalability:** The system must support the simulation of multiple intersections simultaneously without a linear degradation in processing speed.

Usability

1. **NFR-5 Reporting Usability:** The generated CSV reports and visualizations must be automatically

formatted and legible for analysis by non-technical stakeholders (e.g., traffic planners).

Sustainability (Project Specific)

- **NFR-6 Environmental Optimization:** The optimization algorithm must explicitly include C02 emission reduction as a weighted variable in its objective function, not just traffic flow.

---

4. System Design

4.1 Overall System Architecture

The system follows a closed-loop feedback architecture.

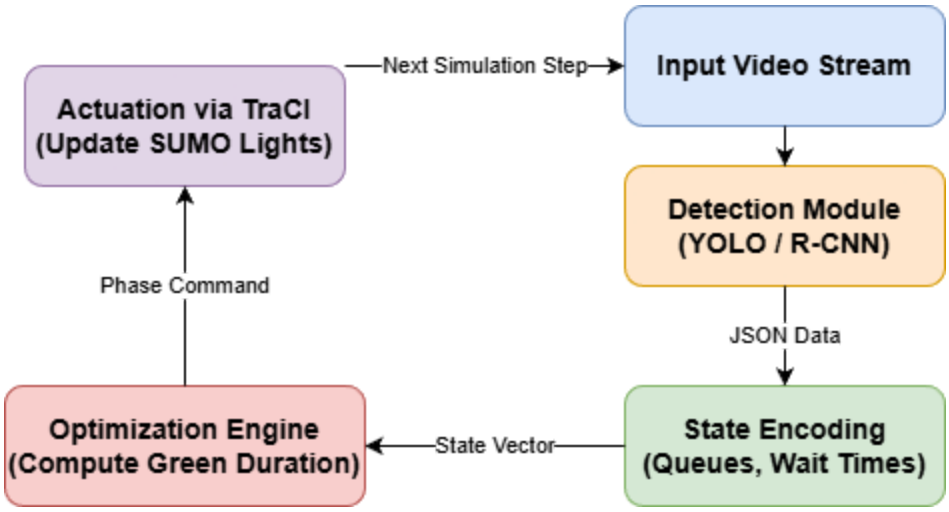

Fig. I : Overall System Architecture of SUMOFlow AI

4.2 Component Breakdown

1. **Input Layer:** Handles video file streams and SUMO network configurations (.net.xml, .rou.xml).

2. **Detection Module:** Uses **YOLOv8** for real-time frame processing and object detection. It is designed to be swappable with **Faster R-CNN** for comparative analysis.

3. **Processing Layer (State Encoding):** The **Vehicle Tracking & Counting Module** converts raw bounding boxes into structured traffic metrics (queue lengths, lane occupancy).

4. **Optimization Engine:** Analyzes the traffic state vector and determines the optimal phase duration using algorithmic logic.

5. **Control Layer (TraCI):** Acts as the bridge, sending the new phase durations to the **SUMO Simulator** to update the traffic lights.

4.3 Design Decisions and Rationale

- **Why SUMO?** It is open-source, highly portable, and allows for microscopic emission modeling, which is crucial for our environmental goals.

- **Why YOLOv8?** It offers the best trade-off between speed and accuracy for real-time applications compared to two-stage detectors like Faster R-CNN.

- **Why Python?** It has robust library support for both AI (PyTorch) and the SUMO interface (TraCI), simplifying integration.

4.4 Program-Specific SO(6) Focus: DSAI (Data Science & AI)

This section details the data lifecycle and modeling approach specific to the DSAI discipline.

- **Data Lifecycle:**
  - **Collection:** Raw video frames are extracted from the simulation or input video files.
  - **Preprocessing:** Frames are resized and normalized. Regions of Interest (ROI) are defined to focus only on lanes approaching the intersection, filtering out irrelevant visual noise.
  - **Transformation:** Unstructured visual data (pixels) is transformed into structured numerical data (Queue Length Q, Waiting Time W) via the tracking module.

- **AI Models & Justification:**
  - We employ **YOLOv8** as the primary perception model due to its single-shot detection

capability, ensuring low inference latency required for the control loop.
    - We are also implementing **Faster R-CNN** and **Fast R-CNN** to conduct a comparative study on detection accuracy versus speed, validating our choice of YOLO for the final deployment.

- **Evaluation Metrics:**
    - **Model Performance:** mAP@0.5 (mean Average Precision) and FPS (Frames Per Second).
    - **System Performance:** Average Waiting Time (seconds), Queue Length (vehicles), and Total C02 Emissions (mg/s).

4.5 Resolution of Supervisor Feedback

During the initial proposal review, the supervisor recommended expanding the project's scope beyond a single detection model to demonstrate a more rigorous engineering approach. Specifically, the feedback advised against relying solely on **YOLO** and suggested incorporating additional AI architectures to validate our design choices.

Resolution:
In response, we expanded our Detection Module (Section 4.2) to include Fast R-CNN and Faster R-CNN alongside YOLOv8.

- We have integrated these models into our development plan to conduct a comparative analysis of **inference speed (FPS)** versus **detection accuracy (mAP)**.
- This benchmarking process allows us to scientifically justify our final model choice for the real-time control loop, ensuring the system balances latency with precision rather than selecting a model arbitrarily.
- The architecture was updated to be modular, allowing these detection models to be swapped seamlessly for testing purposes without altering the downstream optimization logic.

---

5. Project Timeline (Next Semester)

The following timeline outlines the implementation plan for the Spring semester. The Fall semester focused on establishing the SUMO simulation infrastructure, while the Spring semester is dedicated to the implementation and integration of the specific AI modules and optimization logic.

5.1 Phase Breakdown and Responsibilities

Phase	Tasks	Weeks	Responsible
1. AI Module Implementation	**1.1 YOLOv8 Integration: Implement and tune YOLO for real-time detection speed.** **1.2 Fast R-CNN Implementation: Develop the Fast R-CNN pipeline for comparative benchmarking.** **1.3 Faster R-CNN Implementation: Develop the Faster R-CNN pipeline for high-accuracy benchmarking.**	**Weeks 1-4**	**Rana (YOLO)** **Roaa (Fast R-CNN)** **Mariam (Faster R-CNN)**

2. Optimization Logic	**Implement the specific optimization algorithm (e.g., PSO or Rule-Based) to compute Green Time based on the detection outputs.**	**Weeks 5-7**	**Mariam (Lead), All Team**
3. Integration	**Connect the AI Perception Modules, Optimization Engine, and SUMO via TraCI into a single automated closed loop.**	**Weeks 8-10**	**All Team**
4. Testing & Validation	**Run simulation scenarios (low/med/high traffic); Collect metrics; Compare AI models against the fixed-time baseline.**	**Weeks 11-13**	**All Team**
5. Final Reporting	**Generate visualizations; Write final thesis; Prepare defense presentation.**	**Weeks 14-16**	**All Team**

Table II: summarizes the phased development plan, task distribution, and responsibility allocation across

the project timeline.

5.2 Deliverables per Phase

Phase	**Key Deliverables**
Phase 1	- Trained YOLOv8, Fast R-CNN, and Faster R-CNN models - Performance benchmarking report - Detection module documentation
Phase 2	- Optimization algorithm implementation - Algorithm validation results - Traffic signal control logic specification
Phase 3	- Integrated system with TraCI interface - Closed-loop control demonstration - System integration test results
Phase 4	- Simulation results across traffic scenarios - Comparative analysis (AI models vs. fixed-time) - Performance metrics (waiting time, queue length, CO_2)
Phase 5	- Final project report - Technical documentation - Defense presentation slides - Demo video

Table III: presents the main deliverables for each project phase, from AI module development to final reporting and demonstrations.

5.3 Gantt Chart

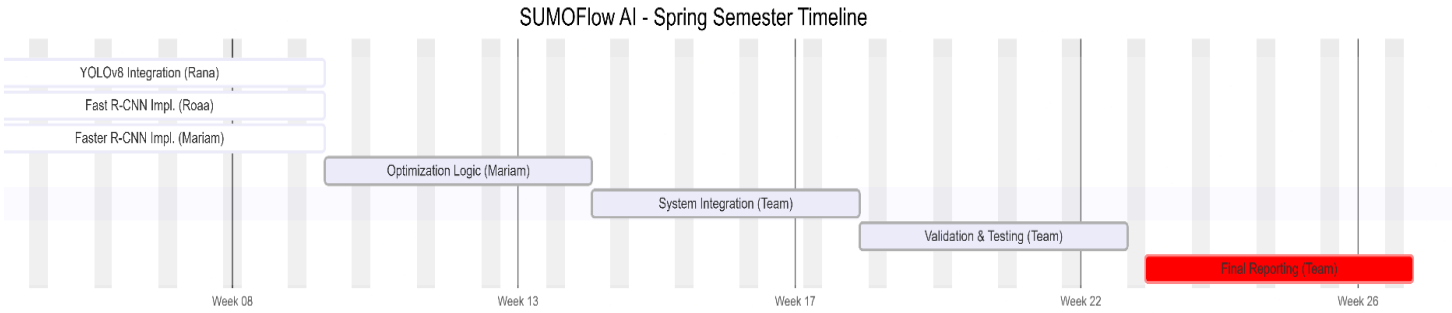

Fig.II: SUMOFlow AI Spring 2025 implementation timeline with overlapping development phases and critical milestones.

---

6. Challenges and Solutions

6.1 Technical Challenges

- **Challenge: YOLO Accuracy Issues.** Pre-trained models struggled with specific camera angles or occlusions in the simulation, leading to inaccurate queue counts.
    - **Solution:** We plan to tune confidence thresholds and implement temporal filtering (tracking vehicles across multiple frames) to stabilize counts.
- **Challenge: Computational Resources.** Running YOLO inference and SUMO simulation simultaneously requires significant GPU/CPU power.
    - **Solution:** We will leverage cloud resources (Google Colab) for training and testing heavy models, and optimize code efficiency by reducing the frame processing rate (e.g., processing every 5th frame).

6.2 Organizational / Team Challenges

- **Challenge: Skill Gaps.** The team lacked prior experience with the TraCI API and complex tracking algorithms.

- **Solution:** We mitigated this by conducting structured weekly learning sessions, where we collaboratively learned SUMO network creation and TraCI basics.

---

7. Work Summary (Fall Semester)

During the Fall semester, the team focused on establishing the simulation infrastructure required to support the AI modules in the next phase:

- **Research & Requirements:** Conducted a deep dive into traffic simulation theory (SUMO), reviewed object detection architectures (YOLO vs R-CNN series), and defined functional/non-functional requirements.
- **System Design:** Finalized the system architecture, data flow diagrams, and component breakdown structure, incorporating supervisor feedback to include multiple AI models for benchmarking.
- **Prototyping (SUMO Focus):**
  - **Network Creation:** Designed and built a custom 4-way intersection road network using **SUMO NetEdit** to serve as the training environment.
  - **Traffic Logic:** Configured traffic light logic, lane connections, and right-of-way rules.
  - **Simulation Data:** Generated route files (.rou.xml) to simulate realistic vehicle flows and verify the "car-following" models.
  - **Evidence:** Screenshots of the NetEdit interface and road network topology were produced and included in internal reports.
- **Timeline & Planning:** Established a detailed roadmap for the Spring semester, assigning **specific AI architectures (YOLO, Fast R-CNN, Faster R-CNN)** to individual team members to ensure parallel development and rigorous comparative analysis.
- **Individual Team Member Contributions:**

  Rana Waleed (202201737): Led the literature review on YOLO architectures and real-time object detection, designed the Detection Module architecture, and prepared the YOLOv8 integration plan. Contributed to requirements analysis and co-authored the Abstract and Introduction sections.

Roaa Raafat (202202079): Conducted the market survey of existing traffic management systems, developed the comparative analysis table, and designed the Processing Layer (State Encoding) specifications. Contributed to non-functional requirements and co-authored the Literature Review and Market Survey sections.

Mariam Alhaj (202200529): Led the SUMO network design and prototyping, configured traffic light logic and route files, and designed the Optimization Engine architecture. Led the incorporation of supervisor feedback and co-authored the System Design and Project Timeline sections.

All Team Members: Participated collaboratively in weekly SUMO learning sessions, system architecture refinement, report compilation and editing, supervisor meetings, and challenge identification and mitigation planning.

References

[1] "SUMO Documentation," German Aerospace Center (DLR), Institute of Transportation Systems. [Online]. Available: https://sumo.dlr.de/docs/index.html.

[2] "YOLO Documentation - Ultralytics," Ultralytics, 2024. [Online]. Available: https://yolo-docs.readthedocs.io/en/latest/.

[3] O. A. Ayegbusi et al., "A Deep Learning-based Model for Traffic Signal Control using the YOLO Algorithm," International Journal of Computer Applications, vol. 186, no. 63, pp. 43-54, Jan. 2025.

[4] M. T. Jahangir, Tahreem Fatima, and Qandeel Fatima, "Evaluating Faster R-CNN and YOLOv8 for Traffic Object Detection and Class-Based Counting," IJIST, vol. 6, no. 4, pp. 1606-1620, Oct. 2024.

[5] "Python 3 Documentation," Python Software Foundation, 2025. [Online]. Available: https://docs.python.org/3/.

[6] C. Gershenson, “Self-organizing traffic lights,” *Complex Systems*, vol. 16, no. 1, pp. 29–53, 2005.