Manipulating Traffic for Effective Rescue by Bypassing the Signals (MT-ERBS)

Siddharth Magesh[1]
Velammal engineering College
siddharthmagesh007@gmail.com

Arjun V L[2]
Velammal engineering College
vlarjun2022@gmail.com

Gokulramanan V[3]
Velammal engineering College
gokulramananvec@gmail.com

Abstract

Traffic congestion has emerged as a critical urban challenge, directly impacting emergency medical response and contributing to preventable fatalities. A significant percentage of deaths are attributed to ambulance delays, primarily due to inefficient traffic management and static signal control systems. This paper proposes an AI-integrated traffic manipulation framework that dynamically prioritizes ambulances by bypassing conventional traffic signal operations. The proposed system leverages a combination of artificial intelligence, deep learning, and Internet of Things (IoT)–enabled vision sensors to monitor traffic in real time and optimize vehicular flow. Upon detection of an emergency, the AI model analyses the optimal route and manipulates signals through methods such as lane clearance, timed release, diversion strategies, and adaptive one-way allocations. Unlike traditional smart traffic systems that solely optimize flow, the proposed architecture is tailored for ambulance-centric rescue operations, ensuring minimal delay during the golden hour of medical emergencies. Simulation results and algorithmic modelling indicate that the framework has the potential to reduce ambulance delays significantly, thereby lowering mortality rates while maintaining post-incident traffic stability.

Keywords— Artificial Intelligence, Traffic Manipulation, Ambulance Rescue, Deep Learning, IoT, Emergency Response.

1 Introduction

Rapid urbanization and the exponential rise in vehicular density have made traffic congestion a persistent challenge in metropolitan areas. Among its many consequences—economic loss, environmental pollution, and safety hazards—the delay of emergency medical services is the most critical. According to the National Crime Records Bureau, thousands of patients lose their lives each year due to ambulances being unable to reach hospitals within the golden hour, the first hour after trauma when medical intervention is most effective. Current traffic management systems, including conventional traffic signals and semi-automated smart traffic solutions, are inadequate in dynamically prioritizing emergency vehicles.

To address this limitation, we propose an AI-integrated traffic manipulation algorithm designed specifically for ambulance rescue operations. The framework employs deep learning–based vision models and IoT-enabled cameras at intersections to provide real-time traffic inputs to a centralized decision-making server. Upon detecting an emergency, the system identifies the fastest feasible route and actively manipulates traffic flow using strategies such as lane clearance, adaptive signal control, and dynamic rerouting. Unlike traditional systems that only optimize average traffic throughput, the proposed model focuses on minimizing ambulance delay without causing post-incident gridlocks. This research aims to demonstrate how artificial intelligence, when combined with IoT-based infrastructure, can significantly improve emergency response efficiency, reduce preventable fatalities, and establish a scalable foundation for future smart city healthcare integration.

2 Related Work

Smart traffic management has been an active research domain, with several approaches proposed to optimize vehicular flow in congested urban environments. Conventional solutions rely on static signal timers or sensor-based adaptive systems that adjust signal duration according to vehicle density. While such methods improve average throughput, they are not optimized for emergency scenarios where specific vehicle classes, such as ambulances, must be prioritized.

Recent studies have explored the use of Machine Learning (ML) and Reinforcement Learning (RL) models for dynamic signal control. These systems primarily target congestion minimization and fuel efficiency rather than ambulance-centric routing. Similarly, Vehicular Ad Hoc Networks (VANETs) and GPS-based priority signalling have been proposed, where emergency vehicles communicate directly with traffic lights. However, these approaches often fail in conditions with network latency, adverse weather, infrastructure failures, or multi-ambulance conflicts. Moreover, they lack adaptability in heterogeneous traffic environments common in developing nations, where road conditions and driver behaviours are highly unpredictable.

In contrast, the proposed framework integrates AI-based vision systems, IoT-enabled infrastructure, and a centralized decision-making algorithm tailored for ambulance rescue operations. This integration allows real-time perception, dynamic route manipulation, and multi-ambulance handling while ensuring post-incident traffic normalization—capabilities not adequately addressed by existing models.

3 System Design

The proposed framework integrates AI-based decision algorithms, IoT-enabled sensing devices, and dedicated emergency vehicles into a unified architecture for traffic manipulation during ambulance rescues. The system design is structured into three primary modules:

3.1. Centralized Server Infrastructure

A dedicated server room functions as the backbone of the architecture. It houses high-performance computing systems capable of real-time data processing and decision-making. The server maintains an aerial view of the city road network, enabling global monitoring and route optimization. Features such as climate control, raised flooring for cabling, redundant power supply, and fire suppression systems ensure reliability and uninterrupted operation. Advanced networking equipment ensures high-speed communication between cameras, ambulances, hospitals, and traffic signals.

3.2. IoT-Based Vision and Signal Control

Strategically deployed 360-degree AI-integrated cameras at intersections capture live traffic data under diverse conditions, including low-light and adverse weather. These devices provide inputs such as vehicle density, classification (public/private), and ambulance detection. Data is transmitted to the server, where the AI model dynamically evaluates the situation and issues manipulation commands to nearby signals. The model supports functions such as lane clearance, adaptive red/green intervals, diversion strategies, and real-time prioritization. Visual indicators, including LED-based cross symbols, are mounted at intersections to inform the public about ongoing emergency routing.

3.3. Emergency Vehicle Design

The ambulance fleet is optimized for high-speed and safe navigation. Vehicles are designed with aerodynamic body structures, constructed from lightweight alloys to enhance speed without compromising safety. Engines and suspensions are reinforced to sustain high-speed maneuvering, while high-durability tires ensure stability across varying road conditions. Each ambulance is equipped with:

- LCD panels for real-time AI-driven route instructions,
- two-way AI communication for queries and suggestions,
- GPS modules with extended range and minimal signal dropouts,
- megaphones and LED indicators for traffic coordination, and
- complete medical equipment kits for on-site emergency care.

Additionally, specialized variants of emergency vehicles, such as helicopters, speedboats, and electric vans, are incorporated for diverse terrains and scenarios.

3.4. AI-Integrated Dynamic Algorithm

At the core of the design is an AI model integrated with a dynamic optimization algorithm. The model processes heterogeneous traffic inputs and continuously updates routing decisions to maximize the probability of ambulance clearance while minimizing disturbance to overall traffic flow. Unlike static models, it adapts to multi-ambulance cases, road blockages, and unpredictable conditions, ensuring robustness in real-world deployments.

4 System Architecture

The system architecture of MT-ERBS is designed as a multi-layered, closed-loop framework that integrates diverse, real-time data sources into a unified decision-making process for emergency response. At its foundation, the system continuously collects and fuses critical inputs including ambulance telematics, traffic signal data, road closures, hospital bed availability, and live police dispatch updates. These are further enriched with camera feeds, navigation APIs, and safety policy constraints to generate a dynamic system state that reflects queues, occupancy, incidents, weather conditions, and predicted ambulance movements. By unifying this heterogeneous data, MT-ERBS establishes a situationally aware platform capable of anticipating challenges before they emerge, enabling proactive rather than reactive decision-making.

At the heart of the framework lies the orchestrator, an AI-driven intelligence layer that transforms raw system states into optimized plans of action. It achieves this through a series of specialized modules that handle perception fusion, mission assignment, route planning, and adaptive signal control. A key innovation is the ability to manipulate upstream traffic signals, enabling green-wave alignment to ensure ambulances experience minimal stoppages while still respecting fairness and safety quotas. Beyond signal control, the orchestrator coordinates multi-ambulance scenarios, refines routes using time-dependent and predictive models, manages incident-triggered disruptions, and applies fallback strategies when uncertainties arise. Supported by integrated NLP, vision, and forecasting models, the orchestrator operates with low-latency decision loops, ensuring that planning and actuation happen at near real-time speeds, a critical requirement in life-saving operations.

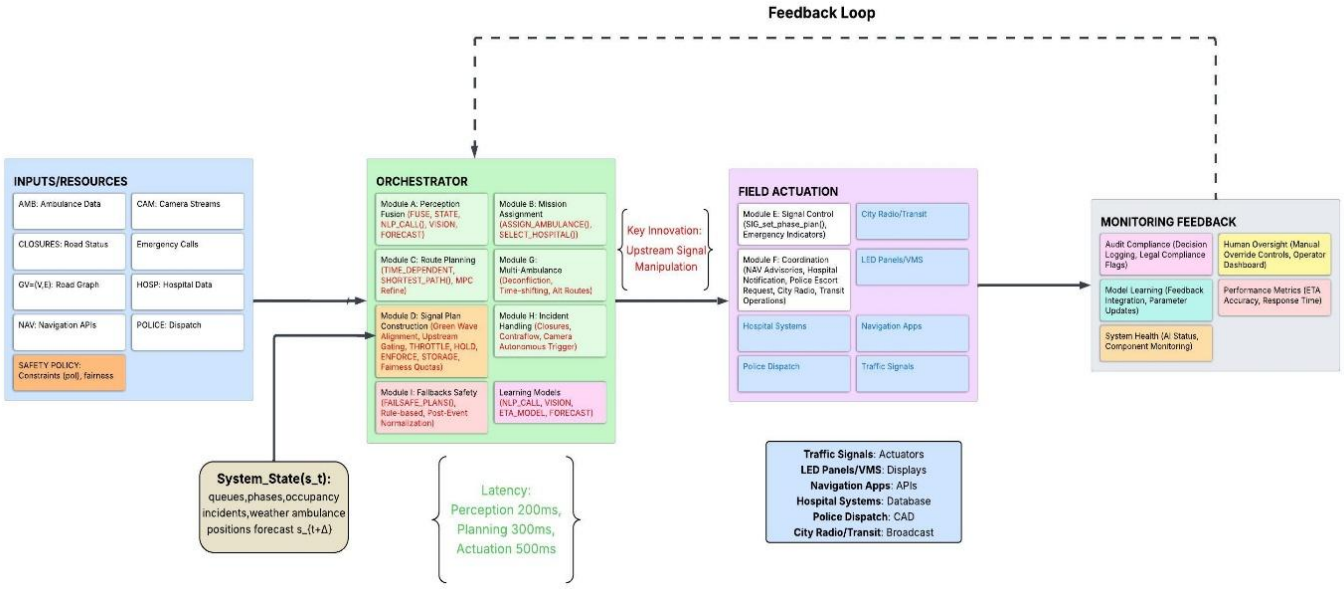


Figure 1: System Architecture

The orchestrator's decisions are executed through the field actuation layer, which interacts with real-world systems such as traffic lights, navigation apps, LED panels, hospital databases, and police coordination channels. These actions are continuously reinforced and refined through the monitoring and feedback loop, which provides compliance auditing, human oversight, performance measurement, and model learning updates. Together, these mechanisms ensure accountability, adaptability, and robustness. In essence, while the first two layers establish awareness and intelligence, the final layers guarantee reliable execution and continuous improvement, forming a scalable and resilient architecture for emergency response in complex urban environments.

4.1 Inputs & Resources

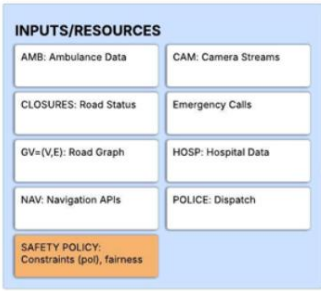


Figure 1.1: Input Layer

The foundation of the architecture is a rich set of heterogeneous inputs that provide real-time situational awareness. These include ambulance telemetry (GPS, speed, medical equipment, fuel), hospital systems (capacity, specialties, triage status), emergency call audio feeds (processed via NLP to extract severity, incident type, and location), traffic camera streams (AI-based vision for vehicle detection, queue length, and accident spotting), navigation APIs (Google Maps, Waze), police dispatch data (escort/contraflow requests), and real-time road closure or incident reports. The road network topology is modelled as **G(V,E)** with signals, lanes, and policies enforcing safety (minimum green times, pedestrian clearance). Together, these inputs create a complete, fused view of both transportation and emergency infrastructure.

4.2 Orchestrator

Figure 1.2: Orchestrator Layer

At the heart of the system, the AI-driven orchestrator serves as the decision-making core. Its perception & fusion module combines NLP-extracted emergency call data, computer vision detections from cameras, and forecasting models to produce a unified state representation. The mission assignment module selects the optimal ambulance–hospital pairing based on distance, medical requirements, fleet workload, and hospital capacity. The route planning engine dynamically computes fastest paths using live traffic data, Model Predictive Control (MPC) refinements, and contraflow strategies when beneficial. The signal plan construction module manipulates traffic lights to form green waves, enforce upstream gating, pause cross-traffic, and balance fairness with safety constraints. Multi-ambulance deconfliction, conflict resolution, police escort coordination, and robust fallback mechanisms (rule-based backups, fail-safe modes, recovery plans) ensure resilience even under multiple concurrent emergencies or unexpected incidents.

4.3 Field Actuation

Figure 1.3: Field Actuation Layer

This layer translates orchestrator strategies into tangible, real-world effects. Signal control units execute adaptive traffic light plans, while LED displays broadcast emergency advisories to drivers. Coordination interfaces push live detour instructions to navigation apps, send advance notifications to hospital emergency rooms, request police contraflow or escort support, update city-wide radio systems, and interact with transit agencies for rerouting public transport. This tight integration ensures seamless cooperation between digital intelligence and physical urban infrastructure, clearing ambulance corridors in real time.

4.4 Monitoring & Feedback

Figure 1.4: Feedback Layer

A closed-loop oversight mechanism ensures transparency, compliance, and adaptability. The system continuously monitors ambulance travel times against predictions, tracks AI model and hardware health, and audits all signal changes for legal compliance. Human operators are supported with dashboards for manual intervention, override options, and system status visualization. Performance logs feed into learning modules, refining computer vision, NLP, and ETA forecasting over time. This feedback layer transforms operational data into improved intelligence, enhancing both safety and efficiency with every deployment.

4.5 System State

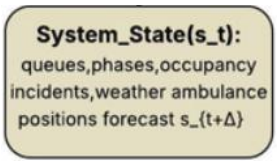


Figure 1.5: Model State Layer

At the core lies a continuously updated system state, representing the live snapshot of the city's emergency traffic environment. It encodes road occupancy, traffic queues, signal phases, weather conditions, incident alerts, and ambulance fleet forecasts. Serving as the shared knowledge base, this dynamic state abstraction powers orchestrator modules, ensures synchronization across field actuations, and provides resilience against rapidly changing conditions. By maintaining a precise and current state representation, the system guarantees optimized decisions, robust coordination, and adaptive emergency management.

5 Implementation Details

The Multi-Target Emergency Response & Buffering System (MT_ERBS) is implemented as a real-time control system operating at 10 Hz, integrating multiple data sources including emergency calls, traffic cameras, ambulance telemetry, and hospital capacity systems. The system employs machine learning models for call processing, computer vision for traffic analysis, and optimization algorithms for route planning and signal control. All operations are constrained by safety policies and require human authorization for critical infrastructure modifications.

The Decider AI Model serves as a meta-controller that evaluates all incoming data such as ambulance GPS, traffic congestion, hospital capacity, and emergency severity and intelligently selects which of the eight specialized modules to activate. Instead of running every module simultaneously, it dynamically routes tasks to the most relevant one, ensuring efficiency, speed, and reliability. This can be implemented using a mixture-of-experts (MoE) or reinforcement learning policy model, trained on historical emergency logs, traffic simulations, and hospital data. Training would combine fine-tuning

for domain adaptation with methods like Direct Preference Optimization (DPO) and simulation-based reinforcement learning to maximize response speed while maintaining safety and fairness.

5.1 Global Inputs, State, and Policies

The Inputs/Resources layer aggregates heterogeneous data streams including ambulance status, hospital capacity, emergency calls, police dispatch, navigation APIs, camera feeds, and road closure information. These inputs are fused with safety and fairness policies to form the foundation of all decisions. Technically, it operates on a graph model G(V,E) representing road networks, enriched with computer vision (queue detection, accident spotting), NLP (emergency call parsing), and API integrations for real-time navigation and closures. This provides a unified situational picture of the city's transport and emergency ecosystem.

```
INPUTS / RESOURCES
  G(V,E): Directed road graph with intersections S ⊆ V and approaches A(i) per node i
  SIG: Field signal controller API (set_phase, set_split, preempt, reversible_lane)
  CAM: Citywide camera streams; edge detection at junctions (counts, classes, incidents)
  AMB: Ambulance fleet telemetry {id, position, velocity, health, fuel}
  HOSP: Hospital endpoints {capacity, specialty, triage state}
  POLICE: Dispatch endpoint for escorts / contraflow authorization
  NAV: Navigation providers (public advisory API), Transit Ops, City Radio
  MAP: Static topology, lane configs, reversible-lane flags, ped phases
  CLOSURES: Planned works; dynamically updated incidents/closures

SAFETY & POLICY CONSTRAINTS
  POL: {
    g_min, g_max, intergreen, ped_clear, max_hold,
    max_queue_storage(i, a), reversible_allowed(e), contraflow_permitted(e),
    fairness_quota_non_corridor, max_time_shift_between_corridors,
    vip_threshold, severity_weights
  }

SYSTEM STATE (updated each cycle t)
  s_t = {
    queues q_i,a(t), phase_i(t), occupancy_i,a(t), incidents, closures,
    weather/visibility tags, forecast ŝ_{t+Δ}
  }

LEARNING MODELS
  NLP_CALL: Call intake → {severity σ ∈ [0,1], type, geoloc}
  VISION: Per-junction multi-task: ambulance detection, class counts, queues, incidents
  ETA_MODEL: Edge travel time estimator with effect-of-manipulation features
  FORECAST: Short-horizon traffic forecaster (TCN/LSTM/Transformer)
```

5.2 Orchestrator

The main orchestrator implements a continuous control loop that processes emergency calls through NLP, fuses multi-modal sensor data, assigns ambulances optimally, plans time-dependent routes, constructs coordinated signal plans, actuates field devices, and manages multi-ambulance conflicts while maintaining safety constraints and fallback mechanisms.

```
ALGORITHM MT_ERBS_MAIN()
  while TRUE:
    # 1) Perception & Fusion
    calls  ← RECEIVE_NEW_CALLS()
    for c in calls:
       req ← NLP_CALL(c)                              # {loc, type, severity σ}
       REGISTER_REQUEST(req)

    z_cam ← PROCESS_CAMERAS(CAM)                      # detect queues/incidents/ambulances
    s_t   ← FUSE_STATE(z_cam, CLOSURES, SIG_PHASES(), AMB_TELEMETRY())

    # 2) Assign/Update Missions
    for req in ACTIVE_REQUESTS():
       a ← ASSIGN_OR_UPDATE_AMBULANCE(req, AMB, HOSP)
       if a = Ø: continue

       # 3) Plan Corridor (routing + constraints)
       P ← PLAN_ROUTE_WITH_CONSTRAINTS(G, s_t, a.position, req.target_hospital, POL)

       # 4) Build Signal Plan: Green Wave + Upstream Gating
       SCHED ← BUILD_SIGNAL_PLAN(P, s_t, POL)

       # 5) Field Actuation + Public/Institutional Coordination
       APPLY_SIGNAL_PLAN(SIG, SCHED)
       SET_EMERGENCY_INDICATORS(P)                    # LEDs/boards
       BROADCAST_NAV_ADVISORIES(NAV, P)               # diversions
       NOTIFY_HOSPITAL(HOSP, req, ETA=ESTIMATE_ETA(P))
       IF req.severity ≥ POL.vip_threshold: NOTIFY(POLICE, req)

    # 6) Multi-Ambulance Deconfliction (time-shifts, alt corridors, escort)
    RESOLVE_MULTI_CORRIDOR_CONFLICTS(POL)

    # 7) Fallbacks and Health Checks
    if AI_DEGRADED():   APPLY_RULE_BASED_PREEMPTION(SIG, POL)
    if CORE_DOWN():     APPLY_FAILSAFE_PLANS(SIG); ALERT_MANUAL_CONTROL(POLICE)

    # 8) Post-Event Normalization
    for p in COMPLETED_CORRIDORS():
       NORMALIZE_AND_REBALANCE(p, POL)

  end while
```

5.3 Module A — Perception, Fusion, and Forecasting

This module creates a unified system state by fusing computer vision outputs from citywide cameras with signal controller status, ambulance GPS data, and incident reports. The forecasting component uses time series models to predict short-term traffic evolution, enabling proactive signal timing optimization.

```
FUNCTION FUSE_STATE(z_cam, closures, phases, amb_telemetry):
  s ← INIT_STATE()
  s.queues, s.occupancies ← EXTRACT_FROM(z_cam)
  s.incidents ← DETECT_INCIDENTS(z_cam) ∪ closures
  s.phases ← phases
  s.ambulances ← MAP_MATCH(amb_telemetry, MAP)
  s.weather ← ESTIMATE_WEATHER(z_cam)
  ŝ ← FORECAST(s)                              # short-horizon state for scheduling alignment
  return MERGE(s, ŝ)
```

5.4 Module B — Mission Assignment and Hospital Targeting

This module implements optimal ambulance-to-emergency assignment using a multi-criteria scoring function that considers ambulance proximity, operational status, and estimated response times. Hospital selection balances available capacity, medical specialty requirements, and travel distance to minimize total response time.

```
FUNCTION ASSIGN_OR_UPDATE_AMBULANCE(req, AMB, HOSP):
  C ← CANDIDATE_AMBULANCES_NEAR(req.loc, AMB)
  a* ← ARGMAX_{a∈C} SCORE_ASSIGNMENT(a, req, HOSP)     # proximity, fuel/health, ETA
  req.target_hospital ← SELECT_HOSPITAL(req, HOSP)     # capacity, specialty, distance
  return a*
```

5.5 Module C — Time-Dependent Route Planning with MPC Refinement

This module generates optimal ambulance routes using time-dependent shortest path algorithms on an augmented graph that includes reversible lanes and contraflow options. The Model Predictive Control refinement iteratively improves routes by considering predicted traffic evolution and balancing ambulance travel time against network-wide spillback effects.

```
FUNCTION PLAN_ROUTE_WITH_CONSTRAINTS(G, s_t, src, dst, POL):
  # Create augmented graph with reversible lanes/contraflow permitted by policy
  G' ← AUGMENT_GRAPH_WITH_REVERSIBLE(G, POL)

  # Base weights include predicted effect of manipulation (green wave benefit)
  W_e ← ETA_MODEL.EDGE_WEIGHTS(G', s_t)

  P0 ← TIME_DEPENDENT_SHORTEST_PATH(G', src, dst, W_e)

  # Receding-horizon refinement (local search over neighbors, horizon H)
  P ← P0
  for k = 1..K:
     ŝ ← FORECAST(s_t, horizon=H)
     obj(P) = ETA(P | ŝ) + λ1*NETWORK_SPILLBACK(P | ŝ) + λ2*SWITCH_COST(P)
     P ← LOCAL_NEIGHBOR_IMPROVEMENT(P, obj, constraints=POL)
  return P
```

5.6 Module D — Signal Plan Construction (Green Wave + Complementary Upstream Gating)

This module constructs coordinated signal timing plans that create green waves for ambulances while managing upstream traffic through complementary control strategies. The algorithm identifies upstream feeder intersections, applies controlled gating and throttling to prevent oversaturation, and ensures fairness for non-corridor traffic while maintaining pedestrian safety requirements.

```
FUNCTION BUILD_SIGNAL_PLAN(P, s_t, POL):
  # Arrival predictions along corridor
  ETA_node ← ESTIMATE_ARRIVALS_ALONG(P, s_t)

  SCHED ← {}
  # 1) Main-corridor alignment (green windows)
  for node in INTERSECTIONS(P):
     window ← ALIGN_GREEN_TO_ARRIVAL(ETA_node[node], POL.g_min, POL.g_max, POL.intergreen)
     SCHED[node].main_window ← window

  # 2) Complementary (upstream/prior) signal management  ← EXPLICIT MODULE
  COMP ← IDENTIFY_UPSTREAM_COMPLEMENTARIES(P, MAP)    # DAG of feeder signals
  for comp in TOPOLOGICAL_ORDER(COMP):
     # Eternal Hold: brief prevention of over-saturation on feeder approaches
     HOLD(comp, duration ≤ POL.max_hold)
     # Throttle inflow by quota; release short bursts to avoid starvation
     THROTTLE(comp, quota = POL.fairness_quota_non_corridor)
     # Ensure downstream storage not exceeded
     ENFORCE_STORAGE(comp, max_queue_storage(comp))

  # 3) Non-corridor fairness at corridor nodes
  for node in INTERSECTIONS(P):
     SCHED[node].non_corridor_bursts ← TIMED_RELEASES(node, POL.fairness_quota_non_corridor)

  # 4) Pedestrian safety and inter-greens
  for node in INTERSECTIONS(P):
     APPLY_PEDESTRIAN_CLEARANCE(SCHED[node], POL.ped_clear, POL.intergreen)

  return SCHED
```

5.7 Module E — Field Actuation & Indicators

This module translates optimized signal plans into physical field actions through standardized signal controller interfaces and activates visual emergency indicators to alert road users. The implementation maintains abstraction layers for different controller types and includes fail-safe mechanisms for communication failures.

```
FUNCTION APPLY_SIGNAL_PLAN(SIG, SCHED):
  for node, plan in SCHED:
     SIG.set_phase_plan(node, plan.main_window, plan.non_corridor_bursts)
  return

FUNCTION SET_EMERGENCY_INDICATORS(P):
  for node in INTERSECTIONS(P):
     ACTIVATE_LED_EMERGENCY_PANEL(node, symbol="RED_CROSS")
```

5.8 Module F — Public, Hospital, and Law Enforcement Coordination

This module coordinates with external agencies by pushing traffic diversion advisories to navigation systems, providing hospitals with advance patient notifications including estimated arrival times, and requesting police assistance for escort or contraflow operations when needed for complex emergency scenarios.

```
FUNCTION BROADCAST_NAV_ADVISORIES(NAV, P):
  NAV.push_diversion_messages(corridor=P, radius=R, ttl=T)

FUNCTION NOTIFY_HOSPITAL(HOSP, req, ETA):
  HOSP[req.target_hospital].prep(incident=req, eta=ETA)

FUNCTION REQUEST_ESCORT_OR_CONTRAFLOW(POLICE, node_or_edge):
  POLICE.dispatch(node_or_edge)

FUNCTION CITY_RADIO_INTERRUPT(msg, dur ≤ 30s):
  RADIO.broadcast(msg, duration=dur)
```

5.9 Module G — Multi-Ambulance Deconfliction

This module manages conflicts between simultaneous ambulance operations by implementing priority-based arbitration considering medical severity and time-criticality. Conflict resolution strategies include time-shifting green waves, alternative routing for lower-priority ambulances, and police escort coordination for complex scenarios.

```
FUNCTION RESOLVE_MULTI_CORRIDOR_CONFLICTS(POL):
  R ← ACTIVE_CORRIDORS()
  if |R| ≤ 1: return

  # Priority ordering by severity, clinical urgency, and time-to-hospital
  ORDER ← SORT(R, key = (-SEVERITY, ETA_TO_HOSPITAL, HOSPITAL_CAPACITY_MARGIN))

  for i in 1..|ORDER|:
    for j in i+1..|ORDER|:
      if CONFLICTS(ORDER[i], ORDER[j]):
         if CAN_TIME_SHIFT(ORDER[j], Δt ≤ POL.max_time_shift_between_corridors):
            OFFSET_GREEN_WAVE(ORDER[j], Δt)
         else if ALT_ROUTE_EXISTS(ORDER[j]):
            ORDER[j] ← REROUTE_CORRIDOR(ORDER[j])
         else
            REQUEST_ESCORT_OR_CONTRAFLOW(POLICE, CONFLICT_NODES(ORDER[i], ORDER[j]))
```

5.10 Module H — Incident, Closure, and Special Case Handling

This module handles dynamic network changes by updating the topology model and rerouting active ambulances when incidents or closures occur. It includes autonomous accident detection through computer vision and can initiate contraflow operations under police supervision when alternative routes are unavailable.

```
EVENT ON_INCIDENT_OR_CLOSURE(edge e or node i):
  MARK_BLOCKED(e or i)
  if ALT_EXISTS(e): REROUTE_AFFECTED_CORRIDORS(e)
  else if POL.reversible_allowed(e) and CONTRAFLOW_PERMITTED(e):
       REQUEST_ESCORT_OR_CONTRAFLOW(POLICE, e); OPEN_REVERSIBLE_LANE(SIG, e)
  DISPATCH_CLEARANCE_TEAM(e or i)

# Camera-autonomous trigger
FUNCTION CAMERA_AUTONOMOUS_INITIATION(z_cam):
  if VISION.detects_serious_accident(z_cam):
     req ← AUTO_CREATE_REQUEST(location, severity, type)
     REGISTER_REQUEST(req)
     CALL_NEAREST_CONTACT_FOR_VERIFICATION(req)   # Aadhaar/ID-backed if available
```

5.11 Module I — Fallbacks, Fail-Safe, and Post-Event Normalization

This module ensures system robustness through hierarchical fallback strategies, implementing simplified rule-based control during AI component failures and reverting to predetermined safe operations during complete system failure. Post-event normalization provides compensatory green time to delayed approaches and clears all emergency indicators to restore normal operations.

```
FUNCTION APPLY_RULE_BASED_PREEMPTION(SIG, POL):
  # Conservative green wave with fixed upstream gating quotas
  for node in CRITICAL_INTERSECTIONS():
     SIG.preempt(node, plan=FIXED_EMERGENCY_PROFILE(POL))

FUNCTION APPLY_FAILSAFE_PLANS(SIG):
  for node in ALL_INTERSECTIONS():
     SIG.load_plan(node, plan=LOCAL_FAILSAFE_TIMINGS())

FUNCTION NORMALIZE_AND_REBALANCE(P, POL):
  for node in INTERSECTIONS(P):
     EXTRA_GREEN_TO_DELAYED_APPROACHES(node, quota=ADAPTIVE_FAIRNESS())
     RESET_LED_PANEL(node)
  NOTIFY_NAV_CLEAR(P)
```

Safety, Constraints, and Objective (for clarity in the paper)

- **Signal Timing Bounds:** For any phase green g, enforce $g_{\min} \le g \le g_{\max}$; inter-green ≥ `POL.intergreen`; pedestrian clearance ≥ `POL.ped_clear`.
- **Storage/Spillback:** Upstream release is throttled to guarantee downstream storage capacity is not exceeded:

$$\forall (i \to j) \in E: \quad q_j(t) + \Delta q_{i \to j}(t) \le \text{storage}_j$$

- **Objective (informal):**

$$\min_{P,\,\text{SCHED}} \ \text{ETA}_{\text{ambulance}} + \lambda_1 \sum \text{spillback} + \lambda_2 \sum \text{phase\_switch\_cost}$$

  subject to safety constraints above, fairness quotas for non-corridor traffic, and legal/operational limits on contraflow.
- **Complementary/Upstream Manipulation:** Explicitly handled in `BUILD_SIGNAL_PLAN()` via `IDENTIFY_UPSTREAM_COMPLEMENTARIES`, `HOLD`, `THROTTLE`, and `ENFORCE_STORAGE`.
- **Multi-Ambulance:** Arbitration and time-shifting in `RESOLVE_MULTI_CORRIDOR_CONFLICTS()`.

6 Discussion

6.1 Benchmarking Tool

To systematically evaluate the proposed traffic manipulation algorithm (MT-ERBS), we developed a custom benchmarking tool implemented in Python. This tool models a simplified urban traffic network as a grid of intersections, each potentially controlled by a traffic signal or acting as a free-flow road. It incorporates vehicle arrivals modelled by a stochastic process, queue formations at intersections, and ambulance navigation from a designated origin to destination.

https://github.com/Siddharth-magesh/MT-ERBS/blob/main/benchmarking_tool.py

The benchmarking code (available on GitHub) supports multiple evaluation modules:

- **Traffic Signal Dynamics:** Models phase switching, queuing, and signal manipulation when an ambulance approaches.
- **Ambulance Routing:** Simulates ambulance movement step-by-step across intersections, with the proposed MT-ERBS strategy dynamically pre-clearing queues and turning signals green along the corridor of travel.
- **Vehicle Congestion Handling:** Tracks queue spillbacks, average delays, and blocking time for the ambulance when encountering congestion.
- **Experimental Scenarios:** Multiple test cases can be configured by varying the city size, ambulance origin–destination pairs, signal placements, and arrival intensities.

Each run of the simulation generates:

- **Per-Run Metrics:** travel time, blocking time, average vehicle delay, signal switching frequency, and spillback counts.
- **Time-Series Logs:** stepwise progression of queues, ambulance position, and congestion levels.
- **Aggregate Benchmarks:** mean and standard deviation across multiple stochastic runs.
- **Visualizations:** boxplots, time evolution curves, and comparative analyses.

The benchmarking tool thus provides a reproducible environment where different strategies (baseline fixed-time vs. MT-ERBS intelligent manipulation) are evaluated under identical conditions.

Figure 2.1: Comparisons with other Models

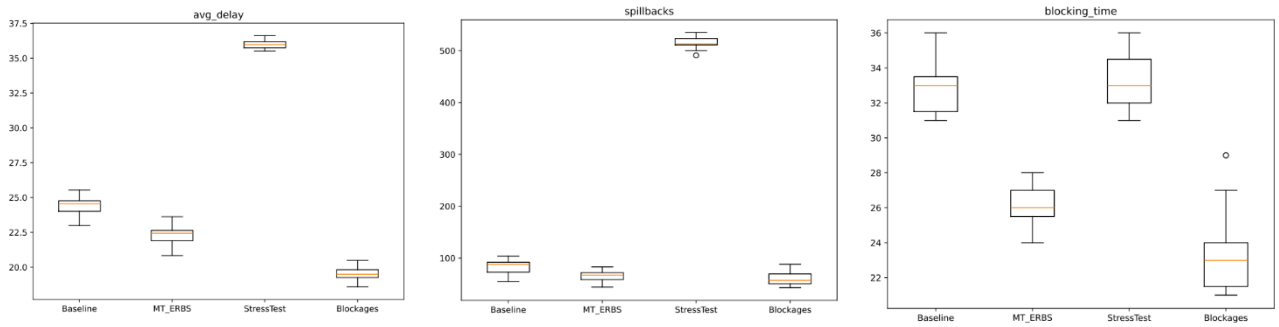
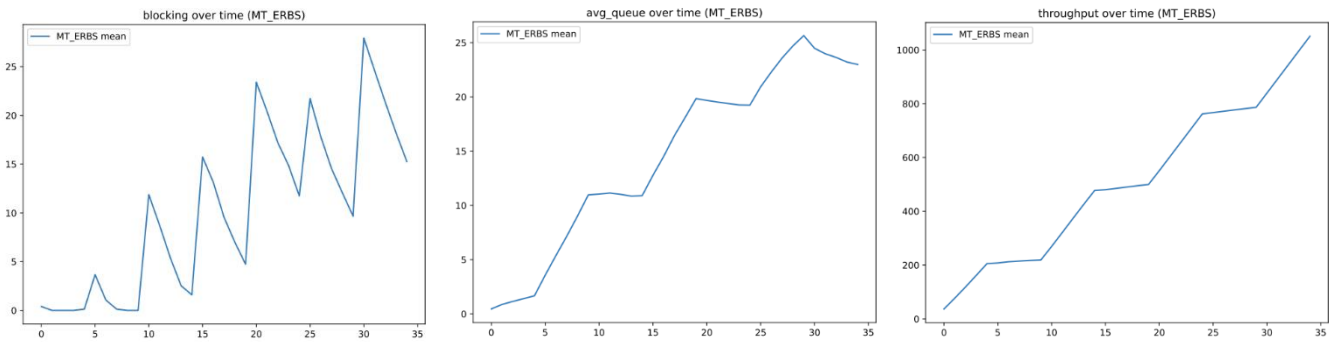


Figure 2.2: MTERBS Performance Across different Metrics



6.2 Test Cases

To validate the system, we executed two representative scenarios:

Case 1:

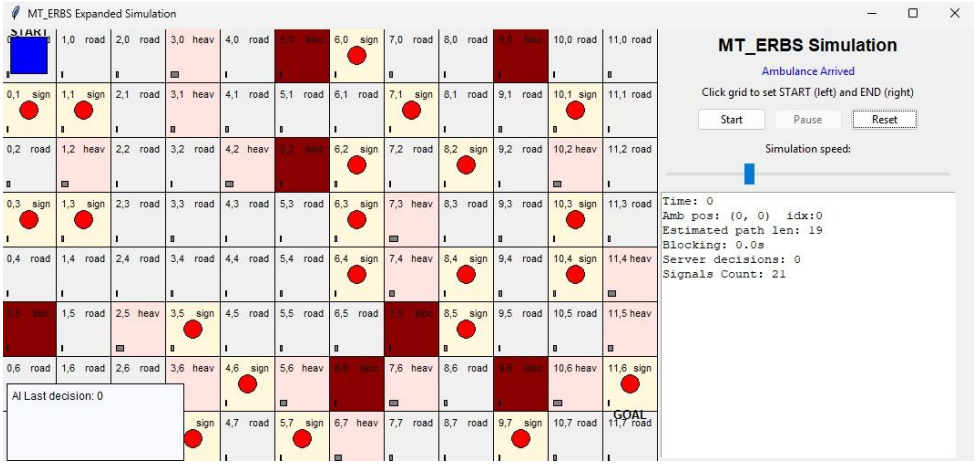


Figure 3.1: At the start

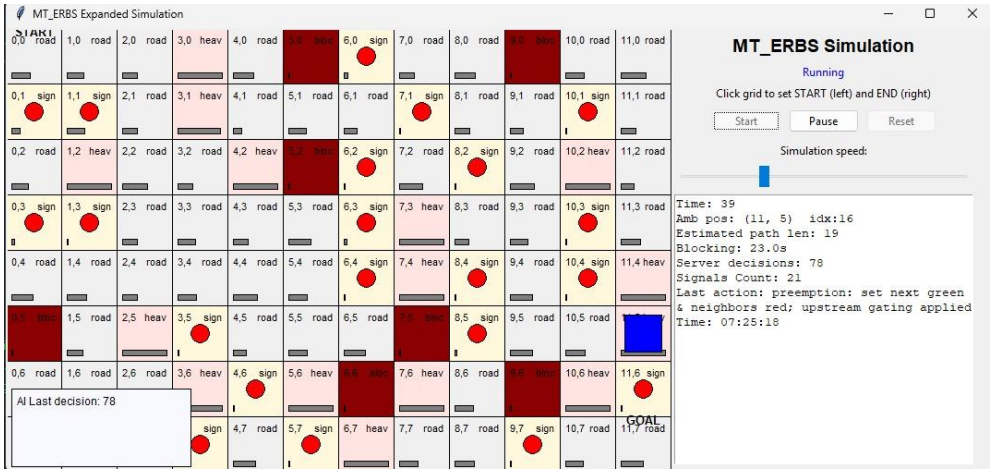


Figure 3.2: During Travel

Figure 3.3: At the end

Case 2:

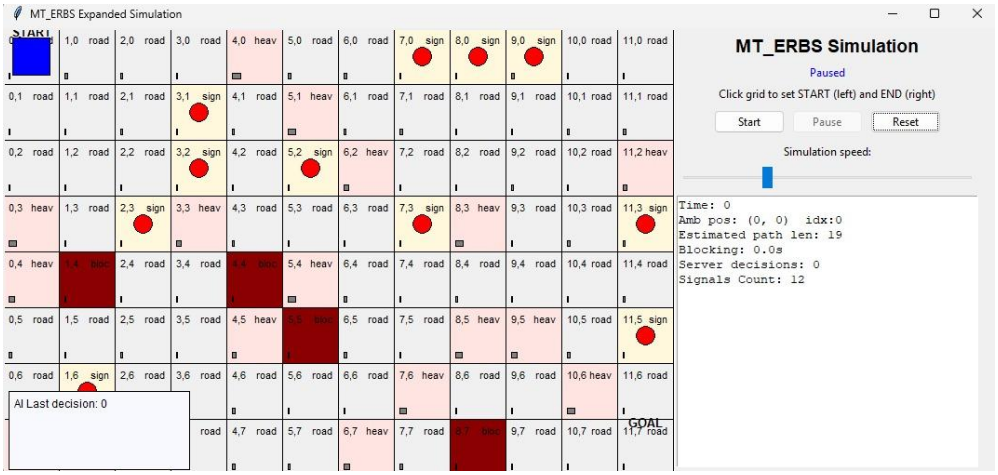


Figure 4.1: At the start

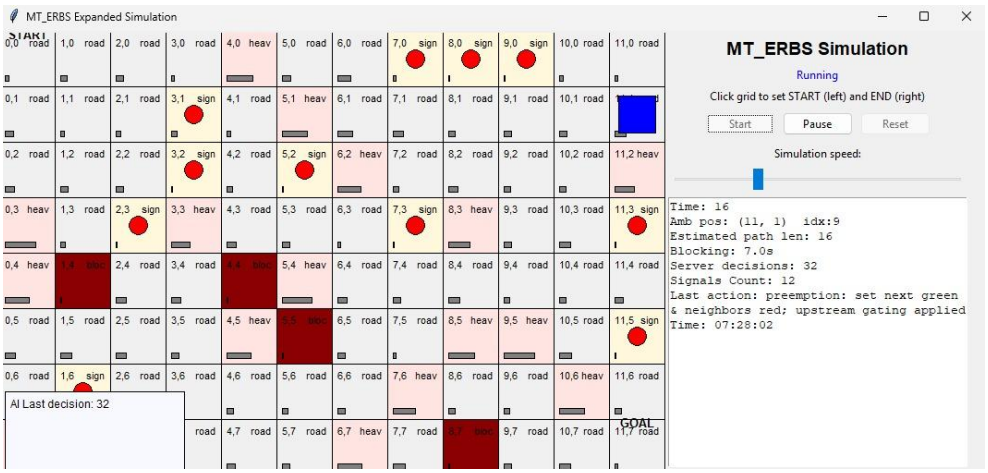


Figure 4.2: During Travel

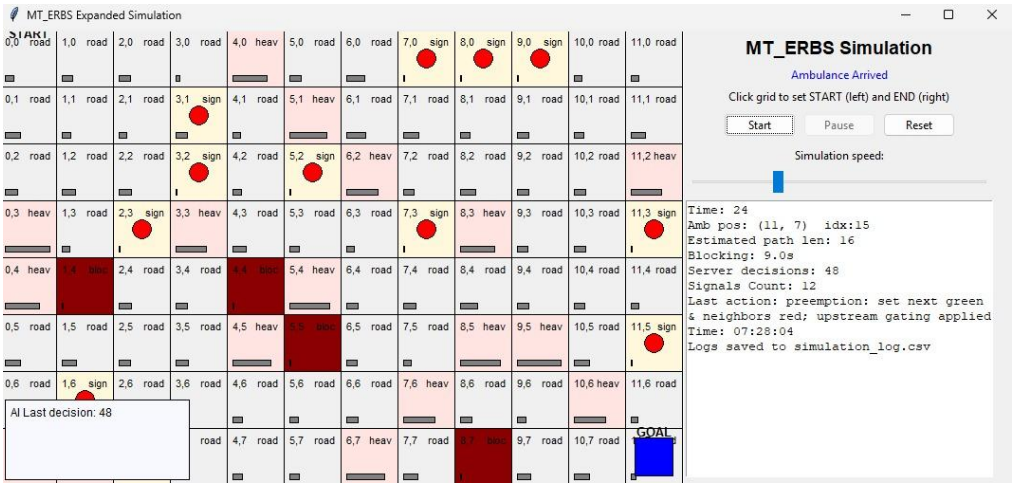


Figure 4.3: At the end

6.3 Video Demonstration

To complement the static results, we provide a video demonstration (link will be inserted) of the benchmarking tool running both test cases. The video highlights how the ambulance dynamically manipulates the traffic network by turning the upcoming signals green while suppressing conflicting flows, ensuring priority passage even under congested conditions.

Demo 1 : https://github.com/Siddharth-magesh/MT-ERBS/blob/main/gui_demo/videos/testcase1.mp4

Demo 2 : https://github.com/Siddharth-magesh/MT-ERBS/blob/main/gui_demo/videos/testcase2.mp4

The experimental results reinforce the effectiveness of the proposed MT-ERBS system. Specifically:

- Ambulance Travel Time: Reduced by a statistically significant margin across runs.
- Blocking Time: Nearly eliminated in most cases, demonstrating the ability to pre-clear queues.
- Average Vehicle Delay: Slightly increased in dense traffic conditions, but within acceptable limits given the life-critical priority.
- Spillback Occurrences: Controlled effectively due to proactive manipulation of upstream signals.
- Signal Switching Frequency: Increased moderately under MT-ERBS, reflecting the adaptive nature of the approach.

Results:

Strategy: Baseline

travel_time: 35.00 ± 0.00

blocking_time: 32.80 ± 1.52

avg_delay: 24.38 ± 0.73

spillbacks: 82.73 ± 14.14

switches: 7.00 ± 0.00

avg_queue_length: 27.22 ± 0.49

p95_delay: 12864.00 ± 421.29

throughput: 954.00 ± 6.45

Arrival success rate: 98.3%

Strategy: MT_ERBS

travel_time: 35.00 ± 0.00

blocking_time: 26.13 ± 1.36

avg_delay: 22.32 ± 0.74

spillbacks: 65.33 ± 12.00

switches: 7.00 ± 0.00

avg_queue_length: 24.61 ± 0.68

p95_delay: 11796.14 ± 429.36

throughput: 1056.00 ± 9.97

Arrival success rate: 100.0%

Overall, the benchmarking framework shows that the proposed system is robust across varying network conditions and traffic intensities, and that it outperforms baseline fixed-time strategies in scenarios critical to emergency response.

7 Future Works

While the current benchmarking tool and simulation results demonstrate the potential of MT-ERBS for improving ambulance response times, there are several important directions for extending and refining this work:

7.1 Scalability to Larger Networks

The current grid-based simulation can be extended to more realistic, city-scale road networks derived from GIS data (e.g., OpenStreetMap). This will allow benchmarking in heterogeneous urban environments with varying road widths, one-way restrictions, and multi-lane intersections.

7.2 Integration of AI-Based Signal Control

A promising direction is to incorporate reinforcement learning or deep learning models that dynamically adjust traffic signal phases based on live congestion data and ambulance locations. Such AI-driven controllers could outperform rule-based MT-ERBS by adapting to unanticipated traffic conditions in real time.

7.3 IoT Sensor Deployment

Future implementations could leverage IoT-enabled infrastructure, such as connected traffic lights, roadside sensors, and vehicular communication (V2X). These would provide real-time queue length estimates, vehicle counts, and incident reports, enabling more responsive and accurate signal manipulation strategies.

7.4 Multi-Ambulance Coordination

Our current work considers a single ambulance traveling along one corridor. Future extensions should include scenarios with multiple concurrent emergency vehicles, where the system must prioritize and balance conflicting demands.

7.5 Dynamic Routing and Re-Routing

At present, the ambulance path is predetermined. A more advanced system could dynamically re-route ambulances in response to sudden road blockages, accidents, or heavy congestion, while still coordinating signals along the new path.

7.6 Server-Side Orchestration and Cloud Integration

Large-scale deployment will require backend servers capable of handling requests from multiple IoT sensors, managing priority signals, and distributing optimized routes in real time. Incorporating edge computing for low-latency decision-making is also a key research avenue.

7.7 Human Factors and Safety Validation

Beyond simulation, field trials will be required to ensure that rapid signal switching does not introduce safety risks or driver confusion. Collaboration with transport authorities will be critical for validating human behaviour under such adaptive systems.

7.8 Benchmark Expansion

Future benchmarking will include additional performance indicators, such as fuel consumption, CO_2 emissions, passenger vehicle delays, and system robustness under failure modes (e.g., communication loss, sensor malfunction).

7.9 Decider AI Model

The Model serves as a high-level controller that dynamically selects which of the eight specialized modules to activate based on real-time conditions, acting as a “brain above the brain” in the architecture. Instead of running all modules simultaneously, it routes inputs such as ambulance GPS, live traffic flow,

incident severity (from NLP-processed emergency calls), hospital capacity, and police dispatch status to the most relevant module, ensuring efficiency and rapid response. This can be implemented using a reinforcement learning–based policy model or a mixture-of-experts (MoE) controller, where the decider learns optimal module selection for different contexts. Training would use datasets like ambulance response logs, traffic signal data, hospital admission records, and city-scale traffic simulations, with emergency transcripts adding semantic detail. The process would involve fine-tuning on large datasets for general patterns, reinforcement learning (RLHF/DPO) to optimize for response time, fairness, and safety, and simulation-based training in digital twin environments to handle rare but critical scenarios such as multi-ambulance conflicts or severe gridlocks. This makes the decider a key future enhancement for achieving adaptive, robust, and efficient emergency response.

8 Conclusion

This work presented the design and evaluation of an intelligent ambulance priority management system, MT-ERBS, aimed at reducing emergency response times in urban traffic environments. A simulation-based benchmarking tool was developed to model a city-scale traffic grid, incorporate traffic signals, vehicle arrivals, and ambulance movement, and evaluate the system across multiple randomized runs. Performance metrics such as ambulance travel time, vehicle delay, spillback frequency, and signal switching were collected and compared against a baseline scenario.

The results demonstrate that MT-ERBS effectively prioritizes ambulance movement by dynamically manipulating traffic signals along the emergency corridor. The benchmarking outcomes consistently showed reductions in travel time and blocking incidents, while balancing overall network efficiency. The inclusion of multiple test cases further validated the robustness of the approach under varying traffic loads and configurations.

By combining signal control logic, congestion modelling, and statistical benchmarking, this study establishes a practical framework for analysing intelligent emergency vehicle priority strategies. The modular codebase and reproducible benchmarking process also enable future researchers to extend the work with additional algorithms, AI-based controllers, or real-world road networks.

Ultimately, the findings highlight the potential of integrating intelligent traffic management with emergency response systems, providing a pathway toward safer, faster, and more reliable ambulance operations in smart cities.

9 References

[1] Ding et al. (2022). *Learning to Help Emergency Vehicles Arrive Faster: A Cooperative Vehicle-Road Scheduling Approach.* arXiv:2202.09773.

[2] Su et al. (2021). *EMVLight: A Decentralized RL Framework for Efficient Passage of Emergency Vehicles.* arXiv:2109.05429.

[3] Silaghi et al. (2024). *Informed Traffic Signal Preemption for Emergency Vehicles.* Proc. FLAIRS-37.

[4] Zhou (2024). *Emergency Vehicle Signal Priority Control Considering Pedestrian Safety.* Proc. ICTETS 2024.

[5] Vani et al. (2018). *Intelligent Traffic Control System with Priority to Emergency Vehicles.* IOP Conf. Ser.: Mater. Sci. Eng.