VLMLight: Safety-Critical Traffic Signal Control via Vision-Language Meta-Control and Dual-Branch Reasoning Architecture

Maonan Wang[1,2*] **Yirong Chen**[2*] **Aoyu Pang**[1] **Yuxin Cai**[3]
Chung Shue Chen[4] **Yuheng Kan**[5] **Man-On Pun**[1†]
[1]The Chinese University of Hong Kong, Shenzhen, China
[2]Shanghai AI Laboratory, Shanghai, China
[3]Nanyang Technological University, Singapore
[4]Nokia Bell Labs, Paris, France
[5]Fourier Intelligence, Shanghai, China
{maonanwang,aoyupang}@link.cuhk.edu.cn chenyirong@pjlab.org.cn
caiy0039@e.ntu.edu.sg chung_shue.chen@nokia-bell-labs.com
kanyuheng@gmail.com simonpun@cuhk.edu.cn

Abstract

Traffic signal control (TSC) is a core challenge in urban mobility, where real-time decisions must balance efficiency and safety. Existing methods—ranging from rule-based heuristics to reinforcement learning (RL)—often struggle to generalize to complex, dynamic, and safety-critical scenarios. We introduce **VLMLight**, a novel TSC framework that integrates vision-language meta-control with dual-branch reasoning. At the core of VLMLight is the first image-based traffic simulator that enables multi-view visual perception at intersections, allowing policies to reason over rich cues such as vehicle type, motion, and spatial density. A large language model (LLM) serves as a safety-prioritized meta-controller, selecting between a fast RL policy for routine traffic and a structured reasoning branch for critical cases. In the latter, multiple LLM agents collaborate to assess traffic phases, prioritize emergency vehicles, and verify rule compliance. Experiments show that VLMLight reduces waiting times for emergency vehicles by up to **65%** over RL-only systems, while preserving real-time performance in standard conditions with less than **1%** degradation. VLMLight offers a scalable, interpretable, and safety-aware solution for next-generation traffic signal control.

1 Introduction

Efficient management of urban traffic is a critical global challenge, with congestion leading to significant economic losses, environmental damage, and a decreased quality of life [1]. Traffic signal control (TSC) at intersections plays an essential role in regulating traffic flow and reducing delays [2]. Traditional TSC methods, such as Webster's method [3], MaxPressure control [4], and Self-Organizing Traffic Lights (SOTL) [5], rely on predefined rules and domain-specific heuristics. While reliable under steady conditions, these rule-based approaches cannot cope with the dynamic, stochastic, and non-stationary realities of modern road networks. As a result, they struggle to respond to unexpected changes in traffic patterns, where real-time, safety-critical decision-making is required to prevent delays and ensure public safety.

*Equal contribution
†Corresponding author: simonpun@cuhk.edu.cn

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

To overcome the rigidity of rule-based approaches, recent studies have explored reinforcement learning (RL) as a more flexible paradigm for TSC [6, 7, 8, 9, 10]. By interacting with the environment and learning from feedback, RL agents can dynamically adjust signal phases in response to changing traffic conditions, offering improved adaptability and long-term optimization. However, despite their success in routine scenarios, these methods still rely on simplified, vectorized state representations, such as queue lengths [11, 12, 13] or intersection pressure [14, 15], and use static reward formulations [16, 17]. Such abstractions omit the semantically rich cues needed for context-dependent decisions. In safety-critical situations, such as granting the right-of-way to ambulances, RL agents trained solely on minimizing delay or maximizing throughput can deviate from the real-world priorities. Although some research [18, 19] has started to consider special vehicles, these methods still face two key challenges: first, determining the trade-off between emergency and regular traffic; second, integrating these priority mechanisms with existing RL-based controllers that differ in agent architecture and reward formulation. Consequently, current RL approaches still fall short when rapid, semantically informed intervention is required.

The emergence of large language models (LLMs) [20] has opened new possibilities for incorporating high-level reasoning and generalization into traffic signal control systems [21, 22]. Recent efforts have explored the use of LLMs to handle edge cases and long-tail scenarios that are difficult for RL-based policies to address [23, 24, 25]. However, current LLM-based approaches often rely on templated or manually crafted textual descriptions of traffic scenes, which lack the richness and fidelity needed to capture complex, real-world dynamics, leading to significant information loss, especially in visually grounded or ambiguous situations. Inference speed poses another hurdle: multi-step reasoning typically makes LLMs orders of magnitude slower than RL controllers, rendering LLMs impractical as standalone controllers in traffic environments where both latency and precision are essential. Moreover, current TSC simulators, such as the widely used SUMO [26] and CityFlow [27], can only provide statistical data for traffic intersections and cannot render real-time images, limiting the possibility to fuse the visual domain with language-based reasoning for traffic signal control.

To address the limitations of both RL- and LLM-based methods, we propose **VLMLight**, a safety-critical traffic signal control framework that integrates vision-language meta-control with dual-branch reasoning. At the core of VLMLight is a novel, custom-built traffic simulator that, for the first time, enables multiview image inputs at urban intersections, allowing policies to reason over rich visual cues such as vehicle types, spatial density, and dynamic motion patterns. Each intersection scene is first processed by VLM to produce interpretable, multi-directional traffic descriptions. These structured scene summaries are then passed to LLM acting as a meta-controller, which determines whether the situation requires low-latency control or high-level reasoning based on semantic understanding of the junction status. For routine traffic, VLMLight invokes a pre-trained RL policy to ensure fast and efficient signal control. In contrast, when a safety-critical condition is detected (e.g., an ambulance requiring priority passage), the system activates a slow deliberative reasoning branch in which multiple specialized LLM agents collaborate through structured dialogue. These agents sequentially perform phase analysis, action planning, and rule compliance verification, simulating a human-like deliberation process. This architecture fuses the speed of learned policies with the interpretability and robustness of language-based reasoning. Empirical results show that VLMLight reduces emergency vehicle waiting time by up to **65%** compared to RL-only baselines. Meanwhile, it maintains comparable performance in standard traffic with less than a **1%** degradation, demonstrating strong potential for trustworthy and scalable deployment.

The key contributions of this work are as follows:

1. We propose VLMLight, a novel traffic signal control framework that integrates vision-language scene understanding for both routine traffic and safety-critical scenarios. The framework leverages the first image-based simulator supporting multi-view visual inputs at intersections, enabling real-time, context-aware decision-making and enhancing flexibility beyond traditional handcrafted traffic states.
2. Guided by an LLM-based meta-controller, VLMLight dynamically selects between a fast RL policy and a structured reasoning module based on real-time junction context. This dual-branch architecture ensures both high efficiency in standard traffic and deliberate, high-level reasoning for safety-critical events, such as prioritizing emergency vehicles.
3. Experimental results show that VLMLight reduces the waiting time for emergency vehicles (e.g., ambulances) by up to **65%** compared to RL-only baselines, while maintaining comparable performance in standard traffic with less than **1%** degradation.

2 Preliminaries

2.1 Traffic Signal Terminology

Figure 1 illustrates a typical four-way intersection. Each intersection comprises two types of approaches: incoming approach with a varying number of income lanes (l_{in}), which carries traffic into the intersection, and outgoing approach with a varying number of outgoing lanes (l_{out}), on which the vehicles can leave the intersection. A *movement*, denoted as m, refers to vehicles going from an incoming approach to an outgoing one. For example, movement m_1 includes l_{in}^2 and l_{in}^3. To regulate traffic flow safely, movements are grouped into *phases*, each representing a set of non-conflicting movements that receive a green light simultaneously. For instance, Phase 1 is defined as $p_1 = \{m_1, m_2\}$. The complete set of feasible signal phases is denoted as $\mathcal{P} = \{p_1, p_2, p_3, p_4\}$.

2.2 Vision-Based TSC Simulator

To overcome the limitations of conventional traffic simulators, we introduce the first vision-enabled traffic signal control simulator that supports multi-view visual inputs at intersections. As illustrated in Figure 1, the simulator allows configurable camera placement to replicate real-world monitoring setups, including a bird's-eye view for monitoring overall traffic flow (left) and directional views simulating roadside perspectives from each approach (right).

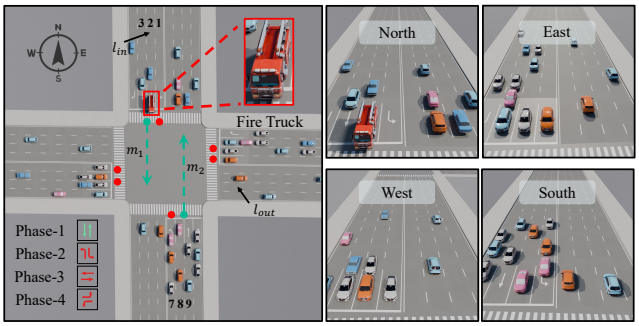


Figure 1: Illustration of a four-way intersection with four signal phases. The simulator supports multi-view visual inputs, including a bird's-eye view (left) and directional views from each approach (right), enabling lane-level observation of vehicle movements. In this example, the North-facing camera captures a fire truck traversing the intersection, highlighting the simulator's ability to support safety-critical reasoning through perceptually grounded traffic understanding.

These views form the perceptual basis for VLMLight. For example, the North-facing camera clearly captures a fire truck navigating through the intersection, and the lane markings provide additional guidance for tracking its trajectory. These visual cues support fine-grained scene understanding, which is essential for adaptive and safety-aware traffic signal control. By enabling real-time observation of dynamic intersection behavior in complex and realistic traffic scenarios, the simulator enhances the system's ability to make informed and context-aware signal decisions under both routine and high-stakes scenarios.

3 Method

We propose **VLMLight**, a vision-language traffic signal control framework that integrates the fast decision-making capabilities of RL with the generalization and structured reasoning strengths of LLMs. The core insight behind VLMLight is that routine traffic scenarios can be efficiently managed by a pre-trained RL policy. In contrast, rare or safety-critical situations, such as the presence of emergency vehicles, require interpretable and high-level reasoning that goes beyond the RL training distribution. As shown in Figure 2, VLMLight consists of four stages: **(1) Scene Understanding**, where a VLM processes multi-view intersection images into natural language descriptions; **(2) Safety-Prioritized Meta-Control**, where an LLM agent analyzes the scene and determines whether to activate the fast RL branch (for routine control) or the deliberative LLM reasoning branch (for complex or critical events); **(3) Routine Control**, where a pre-trained RL policy selects traffic signal actions based on spatio-temporal features; and **(4) Deliberative Reasoning**, where multiple LLM agents collaborate through structured dialogue to assess traffic priorities and verify rule compliance. By expressing perception, reasoning, and control processes in natural language, VLMLight provides a transparent interface that facilitates interpretability and post-hoc analysis across decision stages, thereby bridging low-level control and high-level reasoning in a unified framework. This hybrid

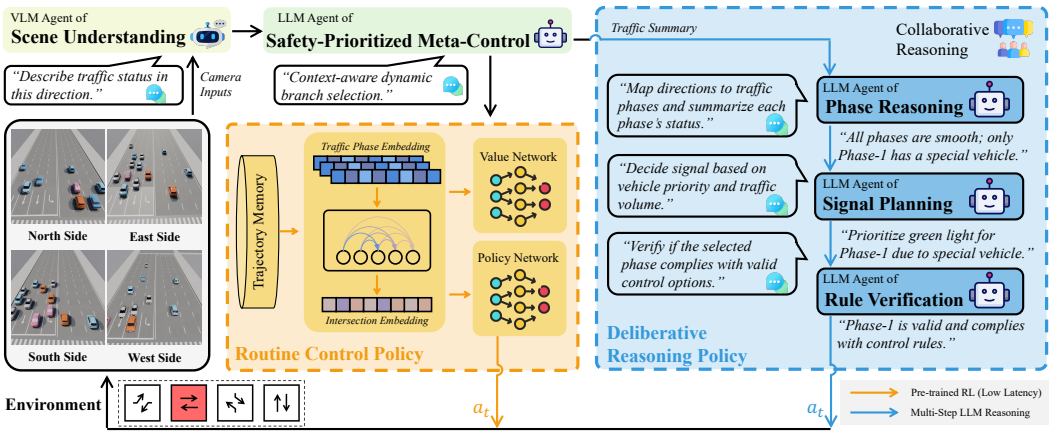


Figure 2: VLMLight architecture. Multi-view intersection images are first parsed by a VLM agent for scene understanding, after which a safety-prioritized LLM meta-controller interprets the scene and selects either a fast RL policy (orange) for routine traffic flow or a collaborative reasoning policy (blue) for safety-critical scenarios. A team of LLM agents—Phase Reasoning, Signal Planning, and Rule Verification—sequentially assess traffic phases, vehicle priority, and rule compliance to determine the final action a_t, giving real-time control and robust handling of complex events.

architecture enables VLMLight to maintain real-time responsiveness in typical scenarios while ensuring trustworthy and explainable decisions under safety-critical conditions.

3.1 Scene Understanding via VLM

Effective TSC requires an accurate and interpretable understanding of intersection conditions. Prior LLM-based approaches often rely on fixed, templated descriptions of traffic state (e.g., queue lengths or average delays), which are limited in expressiveness and fail to capture rich visual semantics, such as vehicle types, spatial configurations, or the presence of special vehicles. To address this, we build a closed-loop traffic simulator that provides real-time image observations from multiple directions of an intersection for perceptual grounding. Based on these visual inputs, a dedicated *Scene Understanding Agent* ($\texttt{Agent}_{\text{Scene}}$) leverages a pretrained VLM to process these inputs and generate natural language traffic scene descriptions.

Given directional images $\{I_1, I_2, \ldots, I_D\}$ from D camera viewpoints (e.g., North, East, South, West), the agent produces a set of textual summaries:

$$\{T_1, T_2, \ldots, T_D\} = \texttt{Agent}_{\text{Scene}}(\{I_1, I_2, \ldots, I_D\}), \tag{1}$$

where each T_i is a textual description of the traffic conditions in direction i, including lane-level semantics, congestion level, and whether any emergency or special vehicles (e.g., ambulances) are present. These free-form yet structured descriptions serve as interpretable inputs for downstream decision modules. As illustrated in Figure 3, this process is demonstrated on a T-junction with three incoming directions. For each I_i, the $\texttt{Agent}_{\text{Scene}}$ generates a corresponding T_i that captures actionable traffic semantics, forming the perceptual basis for mode selection and phase reasoning in later stages.

3.2 Safety-Prioritized Meta-Control

Once the directional scene descriptions $\{T_1, \ldots, T_D\}$ are generated, the system must determine the appropriate decision-making strategy. RL-based policies are efficient for routine traffic but are limited by fixed reward structures, making them unsuitable for unexpected or safety-critical events. In contrast, LLM-based reasoning offers greater flexibility but incurs high latency, which is impractical for real-time control.

To balance responsiveness with adaptability, we introduce a *Safety-Prioritized Meta-Controller*, implemented as a language model agent ($\texttt{Agent}_{\text{ModeSelector}}$). Rather than issuing control actions directly, $\texttt{Agent}_{\text{ModeSelector}}$ interprets the set of textual scene descriptions and decides whether to route

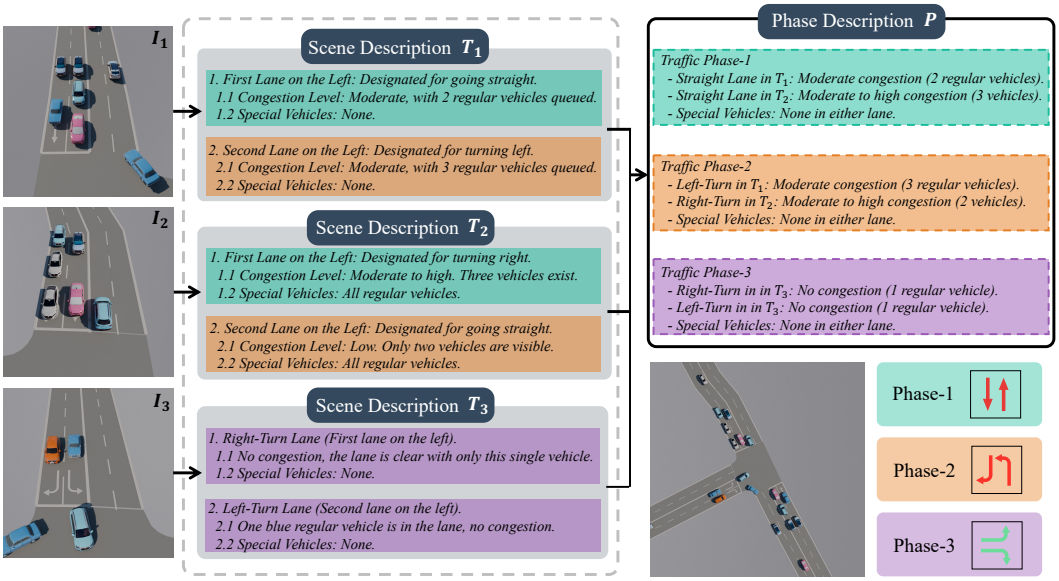


Figure 3: Illustration of $\texttt{Agent}_{\text{Scene}}$ in VLMLight. Given three-view images from a T-junction (left), a VLM-based Scene Description Agent generates directional-level textual summaries (T_1, T_2, T_3), describing lane semantics, congestion, and special vehicle presence. These summaries are then aggregated into phase-level descriptions (P) based on predefined signal phase mappings.

the system to the fast RL routine policy or the deliberative LLM reasoning module. If the scene is consistent with the assumptions of the RL training objective, such as the absence of special vehicles or traffic accidents, the system proceeds with low-latency control via the RL branch. However, if the description includes critical elements such as emergency vehicle detection, conflicting traffic goals, the meta-controller activates the structured reasoning path. This conditional routing allows VLMLight to respond efficiently in common scenarios while retaining the capability to reason explicitly under rare, high-stakes conditions.

3.3 Routine Control Policy

Under normal traffic conditions, VLMLight engages a fast RL decision branch designed for low-latency, high-throughput control. The current intersection state is constructed using recent statistics across up to 12 traffic movements (e.g., vehicle flow, occupancy, signal status), aggregated over the past five time steps to form a spatio-temporal input tensor.

This input is encoded by a lightweight Transformer-based module that first captures spatial dependencies among concurrent movements and then models temporal dynamics across frames. The resulting representation is used to select a traffic signal phase from a predefined discrete action set $\mathcal{A}$.

The RL policy is trained using Proximal Policy Optimization (PPO) [28], with a reward function designed to minimize intersection-level congestion and delay. This routine path enables fast, reactive control without invoking high-level reasoning, making it well-suited for the majority of non-critical traffic scenarios. Implementation details, including the agent design, state encoder, and PPO objective, are provided in Appendix A.1.

3.4 Deliberative Reasoning Policy

When traffic conditions deviate from routine patterns, such as the presence of emergency vehicles, conflicting priorities, or abnormal congestion, VLMLight activates its slow, deliberative reasoning branch. This path engages a team of specialized LLM agents that collaborate through structured natural language dialogue to generate a context-aware signal decision. The process unfolds in three stages: *Phase Reasoning*, *Signal Planning*, and *Rule Verification*.

Phase Reasoning The first step is to convert directional scene descriptions into phase-level traffic summaries aligned with the TSC control action space. As shown in Figure 3, given directional-level

descriptions $\{T_i\}$, the $\texttt{Agent}_{\text{Phase}}$ reorganizes the information using the predefined mapping from traffic movements to signal phases. For instance, Traffic Phase-1 will combine straight-going lanes from T_1 and T_2. The agent aggregates relevant semantic lane-level details (e.g., vehicle types, congestion levels) to form coherent descriptions $\{P_i\}$ for each candidate phase. This transformation bridges low-level visual perception and high-level traffic control, enabling the system to make decisions directly in the discrete space of traffic phases instead of raw directional inputs.

Signal Planning Next, $\texttt{Agent}_{\text{Plan}}$ evaluates the candidate phase descriptions $\{P_i\}$ in light of the current control objectives, such as minimizing delay, prioritizing emergency vehicles, or balancing directional load. Based on this reasoning, it selects the most appropriate phase $a_t^{\text{LLM}} \in \mathcal{A}$ and generates a textual explanation justifying the decision. This rationale provides interpretability and enables auditability of the agent's decision-making process.

Rule Verification Finally, $\texttt{Agent}_{\text{Check}}$ verifies whether the chosen action a_t^{LLM} complies with the current feasible phase set $\mathcal{A}_t$. If the action is valid, the system proceeds with execution. If not, the agent selects an alternative from $\mathcal{A}_t$ that best aligns with the original intent while maintaining safety and consistency. The final decision is then formatted as a JSON object for downstream execution.

4 Experiments

4.1 Experimental Setup

Experiment Settings All experiments are conducted in our custom-built TSC simulator, which integrates SUMO [26] to simulate vehicle dynamics and supports multi-view image rendering for vision-based perception. Each traffic episode adheres to standard urban signal timing, including a green phase (minimum duration of 10 s), a 3-second yellow phase, and a red phase. A minimum headway of 2.5 meters is enforced to reflect safe urban driving behavior. For vision-language processing, we use Qwen2.5-VL-32B [29] to generate structured traffic scene descriptions from multi-view images. In safety-critical scenarios, high-level reasoning is performed by Qwen2.5-72B [30] via structured multi-agent dialogue among three LLM agents. While Qwen is the primary backbone for our experiments, VLMLight is modular and compatible with alternative VLM and LLM models. We analyze the impact of model choices in Appendix C.2.

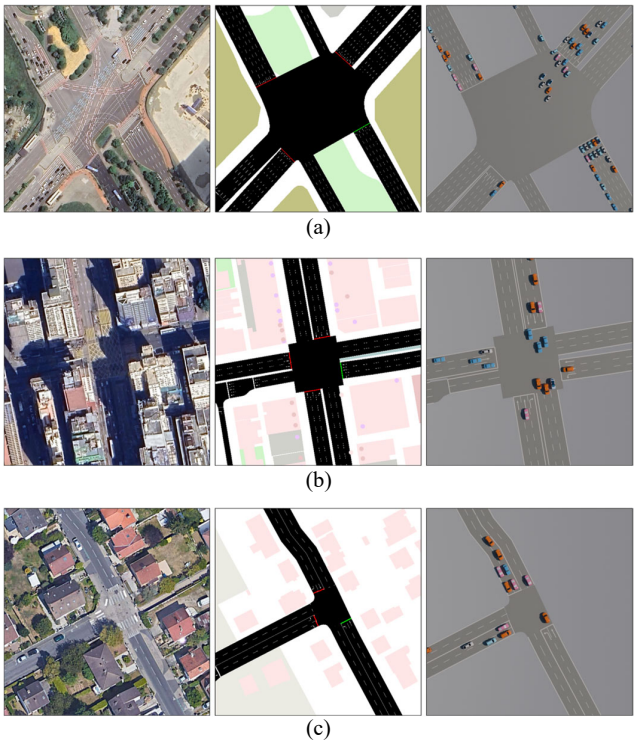


Figure 4: Three real-world intersections, each shown with three image modalities: (a) Songdo (South Korea), (b) Yau Ma Tei (Hong Kong), and (c) Massy (France). For each site, the satellite view is on the left, SUMO simulation in the middle, and our simulator rendering on the right.

Dataset We evaluate VLMLight on traffic data collected from three real-world intersections located in Songdo (South Korea), Yau Ma Tei (Hong Kong), and Massy (France), as illustrated in Figure 4. These locations represent diverse urban settings. Songdo, a newly developed urban district, features larger intersections with up to five lanes per direction. Yau Ma Tei, situated in a dense urban core, has narrower roads and movement

restrictions (e.g., no left or right turns in certain directions). Massy provides a contrasting layout with a T-junction and distinct lane configurations. The diversity of intersection types—including crossroads (Songdo and Yau Ma Tei) and a T-junction (Massy)—as well as differences in size and movement patterns, enables more comprehensive evaluation of the method's generalizability across varied traffic scenarios. Each intersection is equipped with multi-directional cameras capturing 30 minutes of traffic data. The first 20 minutes are used for training the RL policy, while the remaining 10 minutes are reserved for testing. This setup allows us to evaluate the generalizability of VLMLight across varying topologies, lane geometries, and movement constraints. Additional dataset details are provided in Appendix B.2.

Evaluation Metrics We evaluate TSC system performance using four metrics that jointly assess traffic efficiency and emergency vehicle handling. For overall traffic flow, we report the Average Travel Time (ATT), defined as the mean duration for vehicles to reach their destinations, and the Average Waiting Time (AWT), which represents the average duration that vehicles remain nearly stationary (speed < 0.1 m/s), typically due to signal-induced delays. To specifically evaluate emergency vehicle treatment, we measure Average Emergency Travel Time (AETT) and Average Emergency Waiting Time (AEWT). These reflect the system's responsiveness to high-priority traffic. Together, these metrics capture both global intersection efficiency and the framework's ability to prioritize safety-critical scenarios.

Compared Methods To evaluate the performance of VLMLight, we compare it against both traditional and RL-based TSC methods. The traditional baselines include FixTime, Webster [3], and MaxPressure [4], which rely on handcrafted timing rules or pressure-based heuristics. For RL-based approaches, we consider IntelliLight [11], UniTSA [9], A-CATs [6], 3DQN-TSCC [7], and CCDA [10], which learn policies based on vectorized traffic state representations. Since VLMLight is the first framework that supports image-based input and leverages vision-language models to perform real-time traffic signal control, there are no existing VLM-based methods available for direct comparison. Therefore, to provide a more targeted comparison, we additionally include a baseline named Vanilla-VLM, which directly uses VLM-generated descriptions for decision-making without VLMLight's safety-critical meta-control and dual-branch reasoning architecture. A detailed description of these methods is provided in Appendix B.3.

4.2 Performance under Routine and Safety-Critical Traffic Scenarios

We evaluate VLMLight across three real-world intersection datasets and compare it with rule-based, RL-based, and VLM-based baselines. As shown in Table 1, VLMLight achieves strong performance in both routine and emergency scenarios, demonstrating a robust tradeoff between control efficiency and safety-aware decision-making.

Routine Traffic Efficiency VLMLight maintains near-optimal performance under standard traffic conditions. Compared to the best-performing RL-based methods, the increase in ATT is marginal—less than 1% across all datasets. For example, in the Songdo intersection, ATT increases slightly from 86.80 s (UniTSA) to 87.14 s (VLMLight). Similarly, average waiting time (AWT) increases from 39.53 s to 39.73 s. These minor differences reflect the effectiveness of VLMLight's meta-controller, which defers to the fast RL branch in routine settings, avoiding unnecessary LLM overhead. By contrast, Vanilla-VLM, which relies solely on vision-language reasoning, performs significantly worse due to the semantic burden of integrating perception and decision-making within a single monolithic pipeline.

Emergency Vehicle Prioritization In safety-critical scenarios, VLMLight activates its deliberative reasoning branch, leading to substantial improvements in emergency vehicle handling. Across all datasets, VLMLight reduces both AETT and AEWT by over 60% relative to RL-only baselines. For example, in the Songdo dataset, AEWT drops from 22.0 s to just 7.48 s. While Vanilla-VLM also benefits from explicit scene reasoning, its lack of structured phase mapping and decision modularity leads to frequent action errors, especially in complex intersections like Yau Ma Tei. This underscores the importance of decomposing the TSC task into modular reasoning steps. By assigning specialized roles to collaborating LLM agents' reasoning, VLMLight ensures correct, auditable actions and maintains real-time responsiveness even in safety-critical scenarios. Detailed analysis and examples of the intermediate reasoning steps generated by VLMLight are provided in Appendix D.

Table 1: Performance comparison on the three intersections. The top three results are marked with * (best), † (second), and ‡ (third).

Category	Method	South Korea, Songdo			
		ATT ↓	AWT ↓	AETT ↓	AEWT ↓
Rule-based	FixTime	111.73 ± 5.11	59.68 ± 1.95	108.68 ± 7.20	49.53 ± 2.08
	Webster [3]	102.89 ± 4.20	50.02 ± 2.75	82.77 ± 3.94	26.60 ± 1.62
	MaxPressure [4]	93.65 ± 3.41	43.71 ± 1.61	79.38 ± 2.53	35.38 ± 2.42
RL-based	IntelliLight [11]	**87.12 ± 5.10**†	**39.68 ± 1.98**†	**70.00 ± 2.36**‡	22.12 ± 1.27
	UniTSA [9]	**86.80 ± 4.89***	**39.53 ± 1.97***	69.74 ± 3.80	**22.04 ± 0.77**‡
	A-CATs [6]	88.23 ± 6.01	41.61 ± 1.80	70.08 ± 4.40	22.15 ± 0.95
	3DQN-TSCC [7]	99.09 ± 5.68	45.13 ± 2.20	79.62 ± 3.25	25.17 ± 1.54
	CCDA [10]	89.32 ± 6.21	40.68 ± 2.48	71.76 ± 4.82	22.68 ± 1.25
VLM-based	Vanilla-VLM	105.48 ± 17.28	48.09 ± 8.98	**60.38 ± 11.78**†	**11.05 ± 1.73**†
	VLMLight (Ours)	**87.14 ± 4.98**‡	**39.73 ± 1.71**‡	**49.88 ± 2.42***	**7.48 ± 0.45***
Category	**Method**	**Hongkong, Yau Ma Tei**			
		ATT ↓	AWT ↓	AETT ↓	AEWT ↓
Rule-based	FixTime	67.63 ± 4.57	40.00 ± 2.28	82.67 ± 3.23	53.17 ± 2.14
	Webster [3]	56.26 ± 3.39	28.62 ± 1.23	59.67 ± 4.11	30.83 ± 1.79
	MaxPressure [4]	41.36 ± 2.22	13.33 ± 0.40	36.17 ± 2.39	8.83 ± 0.27
RL-based	IntelliLight [11]	**38.07 ± 2.52***	**10.28 ± 0.54***	**33.17 ± 1.54**‡	**5.17 ± 0.18**‡
	UniTSA [9]	**38.10 ± 1.28**†	**10.29 ± 0.50***	33.19 ± 1.92	5.17 ± 0.35
	A-CATs [6]	42.64 ± 1.43	11.51 ± 0.71	37.14 ± 1.67	5.79 ± 0.34
	3DQN-TSCC [7]	46.20 ± 2.01	12.47 ± 0.54	40.25 ± 2.05	6.27 ± 0.32
	CCDA [10]	41.60 ± 2.02	11.23 ± 0.45	36.24 ± 1.45	5.65 ± 0.31
VLM-based	Vanilla-VLM	62.45 ± 8.76	17.62 ± 2.45	**27.79 ± 3.79**†	**5.86 ± 0.69**†
	VLMLight (Ours)	**39.80 ± 1.65**‡	**11.85 ± 0.60**‡	**13.50 ± 0.56***	**2.17 ± 0.12***
Category	**Method**	**France, Massy**			
		ATT ↓	AWT ↓	AETT ↓	AEWT ↓
Rule-based	FixTime	75.84 ± 4.46	28.19 ± 1.58	73.60 ± 4.62	27.80 ± 1.06
	Webster [3]	68.92 ± 2.15	20.89 ± 0.79	65.20 ± 3.04	19.80 ± 0.61
	MaxPressure [4]	64.82 ± 4.01	15.25 ± 0.86	72.40 ± 2.83	22.40 ± 0.92
RL-based	IntelliLight [11]	**57.73 ± 3.51***	**9.80 ± 0.67***	**54.80 ± 2.45**‡	**7.81 ± 0.39**‡
	UniTSA [9]	**57.84 ± 1.91**†	**9.82 ± 0.54**†	64.90 ± 2.28	9.81 ± 0.41
	A-CATs [6]	62.44 ± 3.24	10.60 ± 0.35	59.27 ± 2.72	7.35 ± 0.30
	3DQN-TSCC [7]	69.10 ± 2.86	13.73 ± 0.52	65.59 ± 2.80	8.14 ± 0.49
	CCDA [10]	62.83 ± 2.42	11.19 ± 0.38	58.80 ± 2.40	7.20 ± 0.49
VLM-based	Vanilla-VLM	81.18 ± 11.91	13.44 ± 1.91	**53.11 ± 8.03**†	**3.04 ± 0.61**†
	VLMLight (Ours)	**60.84 ± 2.63**‡	**11.49 ± 0.75**‡	**45.40 ± 2.88***	**2.60 ± 0.10***

4.3 Ablation Study

We conduct ablation studies to quantify the impact of each agent module in the Deliberative Reasoning branch. As shown in Table 2, disabling $\texttt{Agent}_{\text{Phase}}$ leads to a notable increase in AEWT from 48s to 58s, indicating its essential role in the reasoning pipeline. This agent plays a crucial role by abstracting low-level directional features into structured traffic phase representations, enabling downstream reasoning to operate over a discrete and legally grounded action space. Without this abstraction, the reasoning agent must handle raw directional information directly, which leads to error-prone or inapplicable decisions in intersections with complex phase-movement mappings.

In contrast, removing $\texttt{Agent}_{\text{Check}}$, which verifies the legality of the final decision—has limited effect on overall performance, as invalid actions are rare in most cases. These results highlight the importance of structured semantic grounding and explicit planning in traffic signal control. VLMLight's

Table 2: Ablation results on Songdo, impact of structured reasoning components and corresponding inference time.

Modules			Metrics		
$\texttt{Agent}_{\text{Phase}}$	$\texttt{Agent}_{\text{Plan}}$	$\texttt{Agent}_{\text{Check}}$	AETT ↓	AEWT ↓	Time (↓, s)
✗	✓	✗	58.93	9.03	7.10
✓	✓	✗	49.96	8.12	10.97
✓	✓	✓	**48.88**	**7.48**	11.48

modular agent design not only improves emergency response but also supports transparent, auditable decision-making in safety-critical environments.

We further evaluate the inference latency introduced by the structured reasoning branch under the Songdo intersection. As reported in the last column of Table 2, we compare different configurations of VLMLight modules to measure their computational overhead. Even with all modules enabled, the total decision latency remains 11.48 seconds. This latency is well within the acceptable range for real-time deployment, as it fits the typical signal phase buffer (10s green + 3s yellow). Detailed latency breakdowns and further analysis are provided in Appendix C.3.

5 Related Work

Traffic Signal Control TSC methods fall into three categories: rule-based, RL-based, and LLM-based. Traditional approaches like Webster's method [3], SOTL [5], and MaxPressure [4] use fixed heuristics and perform well under steady traffic, but struggle with real-time adaptability [2]. RL-based methods [6, 31, 9, 32, 33, 34, 13, 35] improve responsiveness by learning from interaction, but often overlook safety-critical events due to simplified states and static rewards [36, 37, 38, 39]. Recent LLM-based approaches [21, 23, 22, 25] introduce high-level reasoning for rare or long-tail cases, but rely on hand-crafted text inputs and lack true visual grounding. Moreover, they underperform in routine traffic control compared to specialized RL agents.

Simulators for TSC Most TSC simulators, such as SUMO [26], CityFlow [27], and LibSignal [40], provide only structured traffic states (e.g., vehicle counts, signal phases) without visual outputs, limiting their use in vision-grounded reasoning. High-fidelity driving simulators such as CARLA [41] and MetaDrive [42] offer photorealistic rendering, but they are primarily designed for autonomous driving tasks rather than intersection control, and they often lack native support for signal scheduling or require extensive customization. To address this gap, SynTraC [43] introduces an image-based dataset for TSC built upon CARLA, providing intersection images annotated with traffic states, signal phases, and reward information under diverse weather and lighting conditions. Although SynTraC represents a significant step toward visual perception in TSC, it remains limited to single-intersection scenarios and does not support user-defined or multi-view configurations, constraining its scalability and applicability to broader traffic management research.

6 Conclusion

We present **VLMLight**, a vision-language TSC framework that unifies fast policy execution with structured semantic reasoning. By dynamically selecting between a reinforcement learning policy for routine traffic and a deliberative reasoning branch for safety-critical cases, VLMLight adapts to diverse intersection scenarios with both efficiency and robustness. Empirical experiment results demonstrate that our method reduces emergency vehicle waiting time by up to **65%**, while maintaining comparable performance to RL-only baselines in standard conditions. Importantly, the full inference process completes within 11.5 seconds, which fits comfortably within the typical 13 s signal phase buffer used in urban deployments, indicating practical feasibility for real-time use. In addition, we release the first vision-based TSC simulator with support for multi-view image inputs and dynamic scene rendering. This simulator enables richer visual understanding of intersection conditions and provides a foundation for future research into perceptually grounded and interpretable traffic control strategies.

Limitation VLMLight has two primary limitations. First, our simulator currently lacks diverse weather and lighting conditions, limiting the evaluation of visual robustness under challenging environments. Second, all experiments are conducted on single intersections. Extending the framework to multi-intersection settings requires further study on the scalability and coordination of vision-language reasoning across a broader traffic network.

Broader Impacts VLMLight provides a vision-language framework for TSC that enhances safety and efficiency through real-time visual reasoning. The open-source simulator offers a valuable tool for future research on perception-based traffic systems.

Acknowledgments and Disclosure of Funding

This work was supported in part by the Guangdong Science and Technology Department under Grant 2025SF0001.

References

[1] Amudapuram Mohan Rao and Kalaga Ramachandra Rao. Measuring urban traffic congestion-a review. International Journal for Traffic & Transport Engineering, 2(4), 2012.

[2] Hua Wei, Guanjie Zheng, Vikash Gayah, and Zhenhui Li. A survey on traffic signal control methods. arXiv preprint arXiv:1904.08117, 2019.

[3] Fo Vo Webster. Traffic signal settings. Technical report, 1958.

[4] Pravin Varaiya. Max pressure control of a network of signalized intersections. Transportation Research Part C: Emerging Technologies, 36:177–195, 2013.

[5] Seung-Bae Cools, Carlos Gershenson, and Bart D'Hooghe. Self-organizing traffic lights: A realistic simulation. Advances in applied self-organizing systems, pages 41–50, 2008.

[6] Mohammad Aslani, Mohammad Saadi Mesgari, and Marco Wiering. Adaptive traffic signal control with actor-critic methods in a real-world traffic network with different traffic disruption events. Transportation Research Part C: Emerging Technologies, 85:732–752, 2017.

[7] Xiaoyuan Liang, Xunsheng Du, Guiling Wang, and Zhu Han. A deep reinforcement learning network for traffic light cycle control. IEEE Transactions on Vehicular Technology, 68(2):1243–1253, 2019.

[8] Afshin Oroojlooy, Mohammadreza Nazari, Davood Hajinezhad, and Jorge Silva. Attendlight: Universal attention-based reinforcement learning model for traffic signal control. Advances in Neural Information Processing Systems, 33:4079–4090, 2020.

[9] Maonan Wang, Xi Xiong, Yuheng Kan, Chengcheng Xu, and Man-On Pun. UniTSA: A universal reinforcement learning framework for v2x traffic signal control. IEEE Transactions on Vehicular Technology, 2024.

[10] Maonan Wang, Yirong Chen, Yuheng Kan, Chengcheng Xu, Michael Lepech, Man-On Pun, and Xi Xiong. Traffic signal cycle control with centralized critic and decentralized actors under varying intervention frequencies. IEEE Transactions on Intelligent Transportation Systems, 25(12):20085–20104, 2024.

[11] Hua Wei, Guanjie Zheng, Huaxiu Yao, and Zhenhui Li. IntelliLight: A reinforcement learning approach for intelligent traffic light control. In Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pages 2496–2505. ACM, 2018.

[12] Aoyu Pang, Maonan Wang, Yirong Chen, Man-On Pun, and Michael Lepech. Scalable reinforcement learning framework for traffic signal control under communication delays. IEEE Open Journal of Vehicular Technology, 2024.

[13] Hanyang Chen, Yang Jiang, Shengnan Guo, Xiaowei Mao, Youfang Lin, and Huaiyu Wan. Difflight: a partial rewards conditioned diffusion model for traffic signal control with missing data. Advances in Neural Information Processing Systems, 37:123353–123378, 2024.

[14] Hua Wei, Chacha Chen, Guanjie Zheng, Kan Wu, Vikash Gayah, Kai Xu, and Zhenhui Li. Presslight: Learning max pressure control to coordinate traffic signals in arterial network. In Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining, pages 1290–1298, 2019.

[15] Chacha Chen, Hua Wei, Nan Xu, Guanjie Zheng, Ming Yang, Yuanhao Xiong, Kai Xu, and Zhenhui Li. Toward a thousand lights: Decentralized deep reinforcement learning for large-scale traffic signal control. In Proceedings of the AAAI conference on artificial intelligence, volume 34, pages 3414–3421, 2020.

[16] Zian Ma, Chengcheng Xu, Yuheng Kan, Maonan Wang, and Wei Wu. Adaptive coordinated traffic control for arterial intersections based on reinforcement learning. In 2021 IEEE International Intelligent Transportation Systems Conference (ITSC), pages 2562–2567, 2021.

[17] Hao Mei, Junxian Li, Bin Shi, and Hua Wei. Reinforcement learning approaches for traffic signal control under missing data. In Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence, IJCAI '23, 2023.

[18] Haoran Su, Yaofeng Desmond Zhong, Biswadip Dey, and Amit Chakraborty. EMVLight: A decentralized reinforcement learning framework for efficient passage of emergency vehicles. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 36, pages 4593–4601, 2022.

[19] Miaomiao Cao, Victor OK Li, and Qiqi Shuai. A gain with no pain: Exploring intelligent traffic signal control for emergency vehicles. IEEE Transactions on Intelligent Transportation Systems, 23(10):17899–17909, 2022.

[20] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. arXiv preprint arXiv:2303.08774, 2023.

[21] Yiqing Tang, Xingyuan Dai, Chen Zhao, Qi Cheng, and Yisheng Lv. Large language model-driven urban traffic signal control. In 2024 Australian & New Zealand Control Conference (ANZCC), pages 67–71, 2024.

[22] Siqi Lai, Zhao Xu, Weijia Zhang, Hao Liu, and Hui Xiong. LLMLight: Large language models as traffic signal control agents. In Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V.1, KDD '25, page 2335–2346, New York, NY, USA, 2025. Association for Computing Machinery.

[23] Maonan Wang, Aoyu Pang, Yuheng Kan, Man-On Pun, Chung Shue Chen, and Bo Huang. LLM-Assisted Light: Leveraging large language model capabilities for human-mimetic traffic signal control in complex urban environments. arXiv preprint arXiv:2403.08337, 2024.

[24] Aoyu Pang, Maonan Wang, Man-On Pun, Chung Shue Chen, and Xi Xiong. iLLM-TSC: Integration reinforcement learning and large language model for traffic signal control policy improvement. arXiv preprint arXiv:2407.06025, 2024.

[25] Xingchen Zou, Yuhao Yang, Zheng Chen, Xixuan Hao, Yiqi Chen, Chao Huang, and Yuxuan Liang. Traffic-R1: Reinforced llms bring human-like reasoning to traffic signal control systems. arXiv preprint arXiv:2508.02344, 2025.

[26] Pablo Alvarez Lopez, Michael Behrisch, Laura Bieker-Walz, Jakob Erdmann, Yun-Pang Flötteröd, Robert Hilbrich, Leonhard Lücken, Johannes Rummel, Peter Wagner, and Evamarie Wießner. Microscopic traffic simulation using sumo. In The 21st IEEE International Conference on Intelligent Transportation Systems. IEEE, 2018.

[27] Zheng Tang, Milind Naphade, Ming-Yu Liu, Xiaodong Yang, Stan Birchfield, Shuo Wang, Ratnesh Kumar, David Anastasiu, and Jenq-Neng Hwang. Cityflow: A city-scale benchmark for multi-target multi-camera vehicle tracking and re-identification. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pages 8797–8806, 2019.

[28] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. arXiv preprint arXiv:1707.06347, 2017.

[29] Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, Humen Zhong, Yuanzhi Zhu, Mingkun Yang, Zhaohai Li, Jianqiang Wan, Pengfei Wang, Wei Ding, Zheren Fu, Yiheng Xu, Jiabo Ye, Xi Zhang, Tianbao Xie, Zesen Cheng, Hang Zhang, Zhibo Yang, Haiyang Xu, and Junyang Lin. Qwen2.5-VL technical report. arXiv preprint arXiv:2502.13923, 2025.

[30] An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 technical report. arXiv preprint arXiv:2412.15115, 2024.

[31] Tianshu Chu, Jie Wang, Lara Codecà, and Zhaojian Li. Multi-agent deep reinforcement learning for large-scale traffic signal control. IEEE transactions on intelligent transportation systems, 21(3):1086–1095, 2019.

[32] Minshuo Chen, Yu Bai, H Vincent Poor, and Mengdi Wang. Efficient rl with impaired observability: Learning to act with delayed and missing state observations. Advances in Neural Information Processing Systems, 36:46390–46418, 2023.

[33] Wenlu Du, Junyi Ye, Jingyi Gu, Jing Li, Hua Wei, and Guiling Wang. Safelight: A reinforcement learning method toward collision-free traffic signal control. In Proceedings of the AAAI conference on artificial intelligence, volume 37, pages 14801–14810, 2023.

[34] Yin Gu, Kai Zhang, Qi Liu, Weibo Gao, Longfei Li, and Jun Zhou. π-light: Programmatic interpretable reinforcement learning for resource-limited traffic signal control. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 38, pages 21107–21115, 2024.

[35] Qinchen Yang, Zejun Xie, Hua Wei, Desheng Zhang, and Yu Yang. MalLight: Influence-aware coordinated traffic signal control for traffic signal malfunctions. In Proceedings of the 33rd ACM International Conference on Information and Knowledge Management, pages 2879–2889, 2024.

[36] Kok-Lim Alvin Yau, Junaid Qadir, Hooi Ling Khoo, Mee Hong Ling, and Peter Komisarczuk. A survey on reinforcement learning models and algorithms for traffic signal control. ACM Computing Surveys (CSUR), 50(3):1–38, 2017.

[37] Syed Shah Sultan Mohiuddin Qadri, Mahmut Ali Gökçe, and Erdinç Öner. State-of-art review of traffic signal control methods: challenges and opportunities. European transport research review, 12:1–23, 2020.

[38] Hua Wei, Guanjie Zheng, Vikash Gayah, and Zhenhui Li. Recent advances in reinforcement learning for traffic signal control: A survey of models and evaluation. ACM SIGKDD explorations newsletter, 22(2):12–18, 2021.

[39] Haiyan Zhao, Chengcheng Dong, Jian Cao, and Qingkui Chen. A survey on deep reinforcement learning approaches for traffic signal control. Engineering Applications of Artificial Intelligence, 133:108100, 2024.

[40] Hao Mei, Xiaoliang Lei, Longchao Da, Bin Shi, and Hua Wei. Libsignal: an open library for traffic signal control. Machine Learning, 113(8):5235–5271, 2024.

[41] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. CARLA: An open urban driving simulator. In Proceedings of the 1st Annual Conference on Robot Learning, pages 1–16, 2017.

[42] Quanyi Li, Zhenghao Peng, Lan Feng, Qihang Zhang, Zhenghai Xue, and Bolei Zhou. Metadrive: Composing diverse driving scenarios for generalizable reinforcement learning. IEEE transactions on pattern analysis and machine intelligence, 45(3):3461–3475, 2022.

[43] Tiejin Chen, Prithvi Shirke, Bharatesh Chakravarthi, Arpitsinh Vaghela, Longchao Da, Duo Lu, Yezhou Yang, and Hua Wei. SynTraC: A synthetic dataset for traffic signal control from traffic monitoring cameras. In 2024 IEEE 27th International Conference on Intelligent Transportation Systems (ITSC), pages 2386–2391. IEEE, 2024.

[44] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, Advances in Neural Information Processing Systems, volume 36, pages 34892–34916. Curran Associates, Inc., 2023.

[45] Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. arXiv preprint arXiv:2407.21783, 2024.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: See Abstract and Section 1

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: See Section 6, where two primary limitations are discussed.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [NA]

Justification: We do not have theory assumptions and proofs.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: See Section 4.1.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: See the supplemental material.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: See Section 4 and Appendix B.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: See Section 4, where standard deviation is reported for the experiment results.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.

- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: See Section 4.1 and Appendix. B.1

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: This research follows the NeurIPS Code of Ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: See Section 6.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to

generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.

- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: The paper does not suffer from this risk.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example, by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [NA]

    Justification: The paper does not use existing assets.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [NA]

Justification: The paper does not introduce any new assets.

Guidelines:

- The answer NA means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [NA]

Justification: This paper does not involve research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [NA]

Justification: This paper does not involve research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

Answer: [Yes]

Justification: See Section 3. The backbone used for the experiment is Qwen as disclosed in Section 4.1.

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.

Appendix

In this appendix, we provide supplementary material to further elaborate on VLMLight:

- Additional method details, including the routine control policy, full algorithm, and prompt templates in Section A.
- Full experimental setup, including datasets and compared baselines in Section B.
- Extended results, including RL convergence curves, LLM model comparisons, and inference times in Section C.
- Three case studies illustrating the decision-making process of VLMLight in Section D.
- In-depth discussion on limitations and broader impact in Section E.

A Method

A.1 Details of Routine Control Policy

In standard traffic scenarios, VLMLight adopts an Reinforcement Learning (RL) policy trained within a Markov Decision Process (MDP) framework. At each timestep t, the intersection state is encoded as $J_t = [\mathbf{m}_1^t, \ldots, \mathbf{m}_{12}^t]$, where each $\mathbf{m}_i^t \in \mathbb{R}^7$ represents the status of the i-th movement at the intersection. The vector $\mathbf{m}_i^t$ includes a combination of traffic flow, movement, and signal-related features. The traffic characteristics are captured through the average vehicle flow $F^{i,t}$, maximum occupancy $O_{\text{max}}^{i,t}$, and mean occupancy $O_{\text{mean}}^{i,t}$ since the last control action. The movement-specific features consist of an indicator $I_s^i \in \{0, 1, 2\}$ that reflects whether the movement is straight, left, or right, and the lane count L_i. Signal status is described by two binary indicators: $I_{\text{cg}}^{i,t}$ for whether the current phase is green, and $I_{\text{mg}}^{i,t}$ to indicate whether the minimum green duration requirement has been satisfied. The complete feature vector is written as:

$$\mathbf{m}_i^t = \left[F^{i,t}, O_{\text{max}}^{i,t}, O_{\text{mean}}^{i,t}, I_s^i, L_i, I_{\text{cg}}^{i,t}, I_{\text{mg}}^{i,t}\right]. \tag{2}$$

To ensure a consistent input size, intersections with fewer than 12 movements are zero-padded. For example, at the Yau Ma Tei intersection, the absence of a left-turn movement from north to south is represented by a zero vector. An illustration of the zero-padding scheme is shown in Figure 5. To capture temporal dynamics, the agent receives input from the current and previous four timesteps, forming a 5-frame observation window:

$$S_t = [J_{t-4}, J_{t-3}, J_{t-2}, J_{t-1}, J_t] \in \mathbb{R}^{5\times 12\times 7}. \tag{3}$$

At each timestep, the agent selects an action $a_t \in \mathcal{P}$, where $\mathcal{P}$ denotes the set of available traffic signal phases. The reward r_t is defined as the negative average queue length, encouraging smoother traffic flow.

To extract expressive representations from S_t, we use a Transformer-based encoder that processes both spatial and temporal dimensions. In the spatial encoding stage, the individual movement vectors $\mathbf{m}_i^t \in \mathbb{R}^7$ for each frame $J_t \in \mathbb{R}^{12\times 7}$ are projected to d-dimensional embeddings $\mathbf{h}_i^t$ via a shared linear layer:

$$\mathbf{h}_i^t = \mathbf{W}_e \mathbf{m}_i^t + \mathbf{b}_e, \quad i = 1, \ldots, 12, \tag{4}$$

producing a token matrix $\mathbf{h}^t = [\mathbf{h}_1^t, \ldots, \mathbf{h}_{12}^t] \in \mathbb{R}^{12\times d}$. This matrix is processed by a Transformer block composed of layer normalization (LN), multihead self-attention (MSA), and a feed-forward network (MLP):

$$\mathbf{E}^t = \text{MLP}\left(\text{LN}\left(\mathbf{h}^t + \text{MSA}(\text{LN}(\mathbf{h}^t))\right)\right) \in \mathbb{R}^{12\times d}. \tag{5}$$

We then apply mean pooling across the 12 movement embeddings to obtain a compact spatial summary $\mathbf{s}_t$:

$$\mathbf{s}_t = \frac{1}{12}\sum_{i=1}^{12} \mathbf{E}_i^t \in \mathbb{R}^d. \tag{6}$$

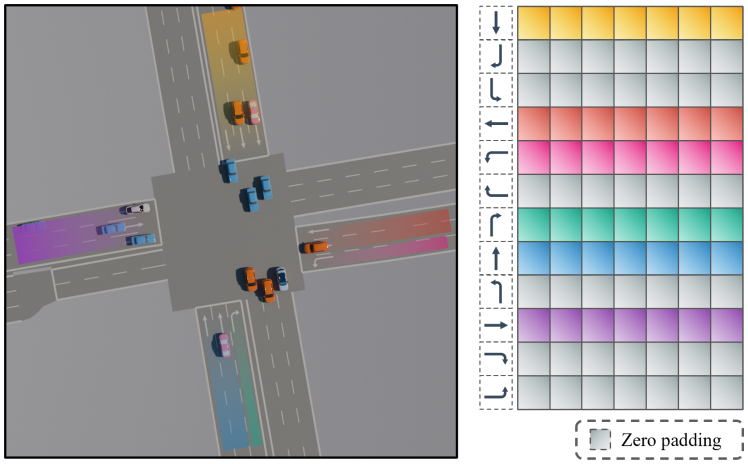


Figure 5: Example of zero-padding at the Yau Ma Tei intersection.

In the second stage, we process the five temporal embeddings $\mathbf{S}_t = [\mathbf{s}_{t-4}, \ldots, \mathbf{s}_t] \in \mathbb{R}^{5\times d}$ using another Transformer encoder:

$$\mathbf{Z}_t = \text{MLP}\left(\text{LN}\left(\mathbf{S}_t + \text{MSA}\left(\text{LN}(\mathbf{S}_t)\right)\right)\right) \in \mathbb{R}^{5\times d}. \tag{7}$$

where each row $\mathbf{Z}_{t,k} \in \mathbb{R}^d$ corresponds to the enriched representation of time step $t-k$. To obtain a fixed-length state embedding for decision-making, we apply average pooling over the temporal axis:

$$\mathbf{f}_t = \frac{1}{5}\sum_{k=1}^{5} \mathbf{Z}_{t,k} \in \mathbb{R}^d. \tag{8}$$

After obtaining the spatio-temporal representation $\mathbf{h}_t$, we pass it into both the policy and value networks. The policy head outputs a probability distribution $\pi(a_t \mid \mathbf{h}_t; \theta)$ over actions, and the value head estimates the expected return $V(\mathbf{h}_t; \phi)$. We optimize the policy using the Proximal Policy Optimization (PPO) algorithm, which maximizes the following surrogate objective:

$$\mathcal{L}^{\text{PPO}}(\theta) = \mathbb{E}_t\left[\min\left(r_t(\theta)\hat{A}_t,\ \text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t\right)\right], \tag{9}$$

where $r_t(\theta) = \frac{\pi_\theta(a_t|\mathbf{h}_t)}{\pi_{\theta_{\text{old}}}(a_t|\mathbf{h}_t)}$ is the policy ratio and $\hat{A}_t$ is the advantage estimate. The clipping parameter ϵ is used to bound policy updates and improve training stability. The value network is trained by minimizing the squared error between predicted values and empirical returns:

$$\mathcal{L}^{\text{value}}(\phi) = \mathbb{E}_t\left[\left(V(\mathbf{h}_t;\phi) - \hat{R}_t\right)^2\right], \tag{10}$$

where $\hat{R}_t$ denotes the bootstrap return. This hierarchical design enables the routine control policy to leverage both short-term dynamics and long-term patterns for efficient traffic management. The full training objective combines the above terms:

$$\mathcal{L}_{\text{total}}(\theta, \phi) = -\mathcal{L}_{\text{policy}}(\theta) + \lambda_v \mathcal{L}_{\text{value}}(\phi), \tag{11}$$

where λ_v is a hyperparameter that balances value learning. This Transformer-based hierarchical design allows the fast RL policy to effectively reason over fine-grained spatio-temporal signals for efficient traffic control in routine scenarios.

A.2 Algorithm for VLMLight

VLMLight employs a modular set of collaborative agents that together enable perception-aware, safety-critical traffic control. Table 3 summarizes the responsibilities of each agent. The architecture is designed to interleave fast decision-making (via RL) with high-level reasoning (via LLM agents)

Table 3: Summary of agent roles in VLMLight.

Agent Name	Function
$\texttt{Agent}_{\text{Scene}}$	Converts multi-view images I_i into directional text descriptions T_i
$\texttt{Agent}_{\text{ModeSelector}}$	Selects control mode: fast RL policy or structured LLM reasoning
$\texttt{Agent}_{\text{Phase}}$	Aggregates T_i into phase-level descriptions P_i
$\texttt{Agent}_{\text{Plan}}$	Selects optimal action a_t^{LLM} and explains the rationale
$\texttt{Agent}_{\text{Check}}$	Validates action feasibility against current legal phase set $\mathcal{A}$

under a unified meta-control mechanism. Each agent operates on structured inputs—either visual, textual, or phase-level representations—and outputs either a decision or an intermediate semantic representation used by downstream agents.

Algorithm 1 Algorithm for VLMLight

Require: Maximum simulation time $T_{\max}$, legal phase set $\mathcal{A}$, phase-to-lane mapping $\mathcal{M}_{\text{ph-lane}}$, maximum check attempts N_{check}, control interval Δt.

Initialize $t \leftarrow 0$
while $t < T_{\max}$ **do**
  Obtain multi-view images $\{I_1, I_2, \ldots, I_D\}$ from simulator
  $\{T_1, T_2, \ldots, T_D\} \leftarrow \textsc{Agent}_{\textsc{Scene}}(\{I_i\})$
  $m \leftarrow \textsc{Agent}_{\textsc{ModeSelector}}(\{T_i\})$
  if $m = \texttt{RL}$ **then**
    $a_t \leftarrow \textsc{Agent}_{\text{RL}}(s_t)$
  else
    $\{P_1, P_2, \ldots, P_K\} \leftarrow \textsc{Agent}_{\textsc{Phase}}(\{T_i\}, \mathcal{M}_{\text{ph-lane}})$
    $a_t^{\text{LLM}} \leftarrow \textsc{Agent}_{\textsc{Plan}}(\{P_k\})$
    for $n = 1$ **to** N_{check} **do**
      $a_t \leftarrow \textsc{Agent}_{\textsc{Check}}(a_t^{\text{LLM}}, \mathcal{A}_t)$
      if $a_t \in \mathcal{A}_t$ **then**
        break
      end if
    end for
    if $a_t \notin \mathcal{A}_t$ **then**
      $a_t \leftarrow \textsc{Agent}_{\text{RL}}(s_t)$
    end if
  end if
  Execute a_t in simulator
  $t \leftarrow t + \Delta t$
end while

Algorithm 1 outlines the inference procedure of VLMLight. At each decision interval, the system receives multi-view images from the simulator and invokes the $\texttt{Agent}_{\text{Scene}}$ to generate directional scene descriptions. A safety-prioritized meta-controller $\texttt{Agent}_{\text{ModeSelector}}$ then determines whether to proceed with the fast RL policy or activate the structured reasoning branch. In routine conditions, a lightweight RL agent issues a control action based on the current traffic state. In contrast, for safety-critical scenarios, three LLM agents collaborate sequentially: the $\texttt{Agent}_{\text{Phase}}$ module transforms scene descriptions into phase-level summaries using a predefined phase-to-lane mapping $\mathcal{M}$ph-lane; the $\texttt{Agent}_{\text{Plan}}$ agent proposes a candidate action aligned with system objectives; and the $\texttt{Agent}_{\text{Check}}$ agent verifies the action's legality against the current feasible phase set $\mathcal{A}_t$. If the verification fails after N_{check} attempts, the system falls back to the action proposed by $\texttt{Agent}_{\text{RL}}$ to ensure continued operation. This strict validation pipeline, coupled with the fallback mechanism, safeguards against potential noise or reasoning errors, ensuring that only feasible and safety-compliant actions are executed. The selected action is then executed in the simulator, and the simulation clock advances by a fixed interval Δt. This loop continues until the maximum simulation time $T\max$ is reached.

A.3 Prompt Templates

This section presents the prompt templates used by the five agents in VLMLight. $\texttt{Agent}_{\text{Scene}}$ uses a VLM to convert intersection images into textual descriptions, as shown in Figure 6. The other four agents are based on LLMs: $\texttt{Agent}_{\text{ModeSelector}}$ determines the control mode (Figure 7), $\texttt{Agent}_{\text{Phase}}$ generates phase-level descriptions based on the scene context (Figure 8), $\texttt{Agent}_{\text{Plan}}$ selects the optimal phase (Figure 9), and $\texttt{Agent}_{\text{Check}}$ verifies rule compliance (Figure 10).

$Agent_{Scene}$

You are TrafficVision, an AI traffic analyst. Based on the intersection image below from a fixed surveillance camera, provide an accurate and concise description of the scene:
- Assess traffic congestion level.
- Identify special vehicles (e.g., ambulances, police cars, fire trucks) only if clearly visible.
- Avoid speculation — report only what is verifiable.

[{Image}]

Figure 6: Prompt template for $\texttt{Agent}_{\text{Scene}}$.

$Agent_{ModeSelector}$

You are in a role play game. The following roles are available:
- Routine Control Agent: Handle normal traffic using RL-based decisions to optimize flow and ensure safety.
- Reasoning Agent: Take over when special vehicles appear or unusual conditions arise, ensuring their priority while keeping traffic orderly. Please read the dialogue history and choose the next suitable role to speak.

When the user indicates to stop chatting or when the topic should be terminated, please return '[STOP]'.
Only return the role name from [{agent_names}] or '[STOP]'. Do not reply any other content.

Figure 7: Prompt template for $\texttt{Agent}_{\text{ModeSelector}}$.

$Agent_{Phase}$

You are a traffic phase analyst. The intersection has [{direction_number}] directional descriptions, each representing a different view. The following is the phase-to-lane mapping for this junction:
[{phase-to-lane}]
Please summarize each traffic phase by extracting:
1) Congestion level
2) Confirmed special vehicles (ambulance, police car, fire truck — only if clearly visible)
3) Any notable traffic events
Use the scene descriptions provided for each direction:
[{junction_description-direction}], …

Figure 8: Prompt template for $\texttt{Agent}_{\text{Phase}}$.

$Agent_{Plan}$

You are roleplaying as a traffic police officer managing a real-time intersection. You have received the description for each traffic phase: [{phase_description}].
Please make decisions by:
- Prioritizing confirmed emergency vehicles (ambulances, police cars, or fire trucks)
- Otherwise, adjusting signal timings to minimize congestion and maximize overall traffic efficiency

Note: You must choose only from the following available actions: [{available_actions}]

Figure 9: Prompt template for $\texttt{Agent}_{\text{Plan}}$.

$Agent_{Check}$

You are an evaluator responsible for verifying whether a traffic control decision is valid.
A valid decision must select an action strictly from the following set: [{self.available_actions}].
If the decision is compliant, return a JSON object with exactly two keys:
- "decision": the selected Traffic Phase ID
- "explanation": the rationale for the decision based on the traffic phase description
No other keys are allowed in the output. Example output:

```
{
  "decision": "Phase-2",
  "explanation": "The image depicts a basic road intersection scenario with no special vehicle markings; …
}
```

Figure 10: Prompt template for $\texttt{Agent}_{\text{Check}}$.

B Experiments Setup

B.1 Experiment Settings

In this section, we describe the experimental setup used to evaluate the performance of VLMLight, our proposed traffic signal control framework. The experiment involves the integration of a self-developed traffic simulator, the deployment of large models for vision-language understanding, and the training of an RL policy to assess VLMLight's adaptability in dynamic traffic scenarios, especially in safety-critical situations.

The experiments are conducted using a self-developed traffic simulation environment built on top of the SUMO (Version 1.22) framework. The simulated roads feature a maximum speed limit of 13.9 m/s (approximately 50 km/h) to model typical urban traffic conditions. The simulator includes three specialized types of vehicles—police cars, ambulances, and fire trucks—to evaluate the system's performance in emergency response scenarios. The vehicle speed distribution is modeled as a Gaussian distribution with a mean of 10 m/s and a variance of 3, reflecting typical urban traffic flow patterns.

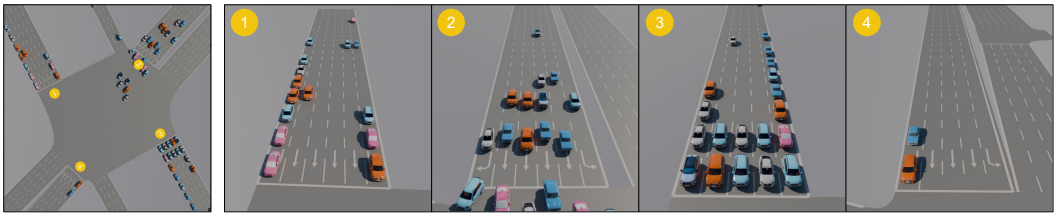
To ensure high-performance traffic signal control, we deploy both VLMs and LLMs locally on a high-performance computing system. The system is equipped with an Intel Xeon 6738P CPU, 256 GB of RAM, and five A100 GPUs, all running Ubuntu 20.04 LTS. This setup allows us to run multiple VLMs in parallel, significantly improving the speed and efficiency of scene understanding. The use of multiple VLMs is essential for rapidly processing the visual inputs from various intersection views and generating natural language descriptions that capture the complex dynamics of the scene. This configuration ensures that VLMLight can make real-time decisions based on a thorough understanding of the current traffic situation.

For the RL-based components of VLMLight, we use the PPO [28] algorithm, implemented via the Stable Baselines3 library. To speed up training and improve the exploration of different traffic conditions, we deploy 30 parallel processes, each interacting with a separate instance of the simulation. The total number of environment steps is set to $3e5$ and the batch size is configured to 64. The learning rate follows a linear schedule, starting at $1e-3$ and gradually decreasing as the number of training steps increases. Additionally, the trajectory memory size is set to 3000.

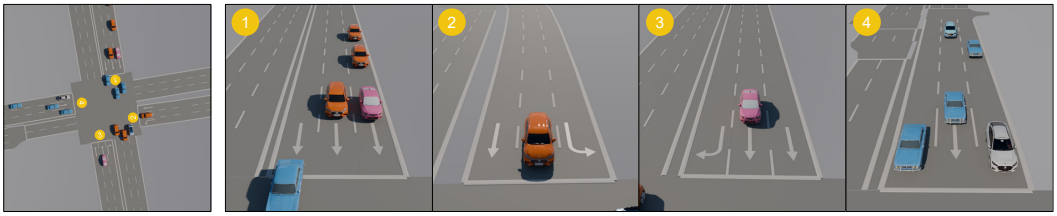
B.2 Dataset Details

To evaluate the generalizability of VLMLight, we construct an evaluation suite based on three real-world intersections with varying topologies and traffic conditions, located in Songdo (South Korea), Yau Ma Tei (Hong Kong), and Massy (France). Each site was selected to represent distinct urban forms: Songdo features large-scale grid intersections with high traffic throughput; Yau Ma Tei is situated in a densely populated downtown with constrained geometry and restricted turning rules; and Massy contains a suburban T-junction with lighter traffic and fewer lanes. These variations allow us to systematically test VLMLight across a broad range of physical layouts and flow intensities.

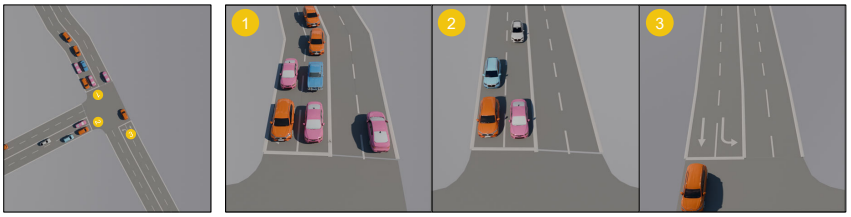
For each intersection, multi-directional cameras were deployed to capture 30 minutes of continuous traffic footage, with all approaches covered. As shown in Figure 11, the first column in each subfigure



(a)



(b)



(c)

Figure 11: Multi-view camera observations of three real-world intersections. Top-down layouts are shown on the left; directional inbound views follow. (a) Songdo, (b) Yau Ma Tei, and (c) Massy.

presents the top-down intersection layout, while subsequent images show direction-specific views from each inbound lane. These direction-wise visual inputs are fed into the VLMLight perception module for downstream reasoning. Table 4 summarizes the directional traffic flow statistics. Vehicle counts and arrival rates vary considerably across sites: Songdo shows the heaviest traffic load, with arrival rates exceeding 2 vehicles/s in certain directions, while Massy represents the lightest scenario with sub-1 vehicle/s flow. Emergency vehicles were sparsely but consistently present across all sites, ensuring meaningful evaluation of safety-critical reasoning. This comprehensive setup enables reproducible and diverse benchmarking for vision-language-based traffic signal control.

B.3 Compared Methods

In this section, we introduce the methods compared to our VLMLight framework, which includes three traditional baselines, five RL-based approaches, and one VLM-based method. These methods are evaluated to highlight the advantages of VLMLight in terms of traffic efficiency and safety.

Traditional Methods. We adopt three traditional approaches in experiments as follows:

- FixTime: Fixed-time control assigns predetermined cycle and phase durations, which are most effective in steady traffic conditions. We consider the FixTime-30 variant, where each phase duration is fixed at 30 seconds.
- Webster [3]: The Webster method adjusts cycle lengths and phase splits based on traffic volumes, optimizing travel time in uniform traffic. In this study, we use it for adjusting traffic lights based on real-time traffic flow.
- MaxPressure [4]: The MaxPressure method prioritizes phases with the highest traffic demand, optimizing the flow by minimizing congestion. This approach is known for its simplicity and effectiveness in maximizing intersection throughput.

Table 4: Traffic flow statistics for each approach direction at the three intersections. #Veh: total vehicles; #Emerg: emergency vehicles.

Network	Dir	#Veh	#Emerg	Arrival Rate (vehicles/s)			
				Mean	Std	Min	Max
Songdo	①	3780	9	2.10	0.31	1.60	2.67
	②	3740	4	2.08	0.32	1.58	2.78
	③	2993	4	1.66	0.23	1.20	2.20
	④	2932	5	1.63	0.21	1.35	2.03
Yau Ma Tei	①	2556	7	1.42	0.20	1.05	1.70
	②	1916	4	1.06	0.17	0.78	1.37
	③	1927	4	1.07	0.17	0.75	1.35
	④	2346	2	1.30	0.20	1.00	1.65
Massy	①	1216	3	0.68	0.09	0.45	0.85
	②	626	2	0.35	0.06	0.25	0.47
	③	1079	2	0.60	0.10	0.42	0.78

RL-Based Methods. We examine five RL-based methods, each offering distinct strategies for TSC:

- IntelliLight [11]: IntelliLight uses a DQN-based approach to select the best traffic phase from available options, addressing data imbalance by maintaining a balanced data buffer for each phase. In this study, decisions are made every 5 s from all available phases.
- UniTSA [9]: UniTSA introduces junction matrices, which enable it to adapt to different intersection layouts. The method also leverages state augmentation, ensuring the agent encounters diverse intersection types and traffic volume during training.
- A-CATs [6]: A-CATs employs an actor-critic approach to train TSC agents, where the output phase duration is adjusted within a range of 10 to 40 seconds. This method provides continuous learning for phase duration optimization based on traffic conditions.
- 3DQN-TSCC [7]: 3DQN-TSCC applies DQN to adjust phase durations in small increments, focusing on stabilizing the signal light transitions. In this method, the phase duration is modified by a fixed set of values $\{-5, 0, 5\}$ s.
- CCDA [10]: CCDA introduces a centralized critic and decentralized actor framework, ensuring stability in phase duration changes. The method adjusts all phase durations in smaller steps $\{-6, -3, 0, 3, 6\}$ s and decisions are made every 10 s to ensure stability.

VLM-Based Method. We also consider a VLM-based approach, Vanilla-VLM, which directly utilizes a VLM for scene understanding and generates a textual description of the traffic situation, which is then used by an LLM to make decisions without the involvement of RL policies in regular scenarios.

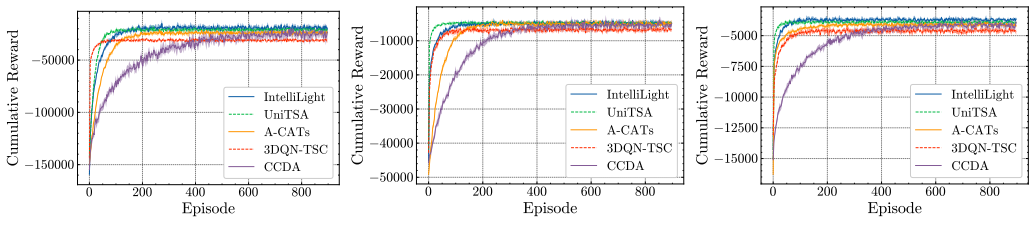


Figure 12: Training reward curves of five RL-based TSC methods across three real-world intersections: (a) Songdo (South Korea), (b) Yau Ma Tei (Hong Kong), and (c) Massy (France).

C Additional Experiments Results

C.1 Additional Performance Analysis of RL Methods for TSC

In this section, we describe five RL-based TSC methods: IntelliLight, UniTSA, A-CATs, CCDA, and 3DQN-TSC. Figure 12 shows the reward curves for each method in three scenarios, where the x-axis represents training episodes and the y-axis represents cumulative rewards. Among the methods, IntelliLight and UniTSA achieve the highest cumulative rewards and exhibit rapid convergence. Their superior performance stems from their design: both adopt direct phase-switching actions, allowing for timely and responsive adjustments to dynamic traffic conditions.

A-CATs and CCDA exhibit competitive, though slightly inferior performance. A-CATs decomposes the multi-phase scheduling task into sequential single-phase adjustments, which increase the learning difficulty but eventually result in near-optimal control once convergence is reached. CCDA extends this idea by enabling simultaneous updates of all phases with a larger action space, leading to slower convergence but similar final performance.

Finally, 3DQN-TSC performs the worst across all scenarios. Its restrictive action design—limited to modifying a single phase per cycle—constrains its ability to optimize phase allocations holistically. As a result, it accumulates fewer rewards and struggles to match the performance of other methods.

Table 5: Performance comparison of different VLMs for scene understanding. The top two results are marked with * (best), † (second).

Scene	**Metrics**	VLMLight	Qwen2.5-VL-7B	LLava-7B	LLava-13B	GPT-4o
Songdo	ATT ↓	**87.14***	92.68	95.25	91.96	**87.46**†
Yau Ma Tei		**39.80***	41.79	48.54	41.27	**41.12**†
Massy		**60.84***	65.62	70.54	66.44	**63.72**†
Songdo	AETT ↓	**49.88**†	76.13	80.45	60.86	**48.06***
Yau Ma Tei		**13.50**†	30.34	27.55	19.71	**13.04***
Massy		**45.40**†	62.63	65.97	55.78	**45.17***

Table 6: Performance comparison of different LLMs for mode selection.

Scene	**Metrics**	VLMLight	Qwen2.5-7B	Qwen2.5-32B	Llama3-70B	GPT-4o
Songdo	ATT ↓	**87.14**†	87.18	**86.28***	89.53	88.57
Yau Ma Tei		**39.80**†	41.01	41.14	**38.89***	40.19
Massy		**60.84**†	62.46	62.73	62.67	**60.79***
Songdo	AETT ↓	49.88	**49.81**†	50.67	51.25	**49.70***
Yau Ma Tei		13.50	13.36	13.91	**13.19**†	**12.96***
Massy		**45.40***	46.81	48.61	**45.44**†	46.76

Table 7: Performance comparison of different LLMs on reasoning policy.

Scene	**Metrics**	VLMLight	Qwen2.5-7B	Qwen2.5-32B	Llama3-70B	GPT-4o
Songdo	ATT ↓	**87.14***	90.59	88.59	**87.40**†	87.99
Yau Ma Tei		39.80	**38.50**†	40.57	**38.23***	39.93
Massy		**60.84**†	63.71	62.56	**59.11***	61.11
Songdo	AETT ↓	49.88	51.85	49.71	**49.45**†	**49.37***
Yau Ma Tei		**13.50**†	13.34	13.97	13.76	**13.14***
Massy		**45.40**†	53.68	46.68	50.54	**44.61***

C.2 Additional Ablation Study on different LLMs

In this section, we analyze the impact of different LLM models on the performance of VLMLight across various modules, including scene understanding, mode selection, and reasoning policy. We evaluated multiple models, including Qwen2.5-VL-7B [29], LLaVA-7B, LLaVA-13B [44], and GPT-4o [20] for the scene understanding agent ($\texttt{Agent}_{\text{Scene}}$), and Qwen2.5-7B, Qwen2.5-32B [30], Llama3.1-70B [45], and GPT-4o [20] for the $\texttt{Agent}_{\text{ModeSelector}}$ and reasoning policy agents. The results are presented in Tables 5, 6, and 7, showcasing the effects of model selection on the respective modules.

The results indicate that the scene understanding module ($\texttt{Agent}_{\text{Scene}}$) is the most sensitive to model changes. As shown in Table 5, the performance of the model significantly affects both the Average Travel Time (ATT) and Average Emergency Travel Time (AETT), especially in identifying special vehicles. For instance, switching from Qwen2.5-VL-32B to Qwen2.5-VL-7B results in notable increases in the waiting time and travel time of special vehicles, likely due to missed recognition of emergency vehicles. Moreover, the performance drop in model accuracy also leads to longer travel times for regular vehicles, as normal vehicles may be mistakenly treated as special vehicles, causing unnecessary green lights to be given. This highlights the importance of a reliable and accurate scene understanding for the overall system performance, as inaccurate scene descriptions can propagate errors in the subsequent decision-making stages.

In contrast, the $\texttt{Agent}_{\text{ModeSelector}}$ and reasoning policy agents, which involve simpler textual processing tasks, show more resilience to model changes. As indicated in Table 6 and Table 7, even smaller models like Qwen2.5-7B maintain similar performance to larger models, with only marginal differences in ATT for regular vehicles. However, special vehicles may still experience slight delays due to inaccurate lane-to-phase mappings in the reasoning process. Overall, the most critical takeaway is that the scene understanding module has the greatest impact on VLMLight's performance, particularly in ensuring timely prioritization of special vehicles. Thus, using a high-performance model for scene understanding is essential for maintaining the system's ability to handle complex, safety-critical scenarios effectively.

C.3 Additional Ablation Study on Inference Time

In this section, we analyze the inference time of VLMLight across three distinct environments. The results, shown in Table 8, demonstrate that VLMLight achieves inference times well below 13 s in all three environments, falling within an acceptable range for real-world deployment. This is particularly notable considering the minimum green light duration of 10 s and the additional 3 s for yellow lights.

As illustrated in Table 8, the majority of the inference time is spent on scene understanding, while mode selection and deliberative reasoning stages require considerably less time. Overall, the results indicate that VLMLight is suitable for deployment in practical settings, with its architecture optimized for both speed and safety.

Table 8: Inference time for each stage of VLMLight across three environments.

Stage	Songdo	Yau Ma Tei	Massy
$\texttt{Agent}_{\text{Scene}}$	5.12	5.15	4.79
$\texttt{Agent}_{\text{ModeSelector}}$	0.75	0.95	0.77
$\texttt{Agent}_{\text{Phase}}$	3.87	1.95	2.24
$\texttt{Agent}_{\text{Plan}}$	1.23	0.86	1.21
$\texttt{Agent}_{\text{Check}}$	0.51	0.45	0.34
Total	**11.48**	**9.36**	**9.35**

D Case Study

To showcase **VLMLight** in action, we present three representative case studies. Each example covers a complete TSC cycle from time step T to $T+1$, demonstrating how different agents collaborate under both routine and safety-critical scenarios. For each case, we describe the visual inputs, the decision

made by each agent, and the resulting traffic outcome. This offers insight into how VLMLight dynamically selects between the fast RL branch and the deliberative LLM branch as circumstances demand.

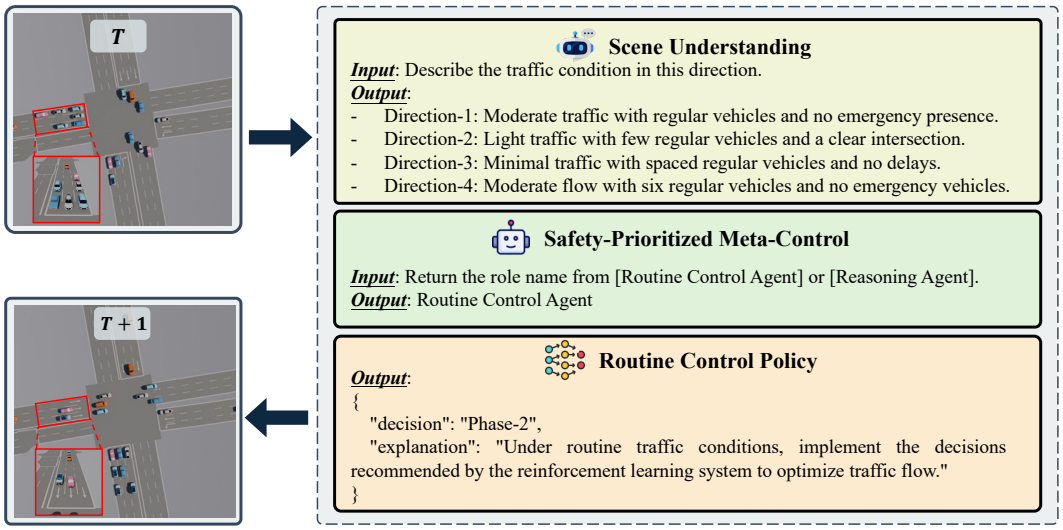


Figure 13: Routine Control in Yau Ma Tei.

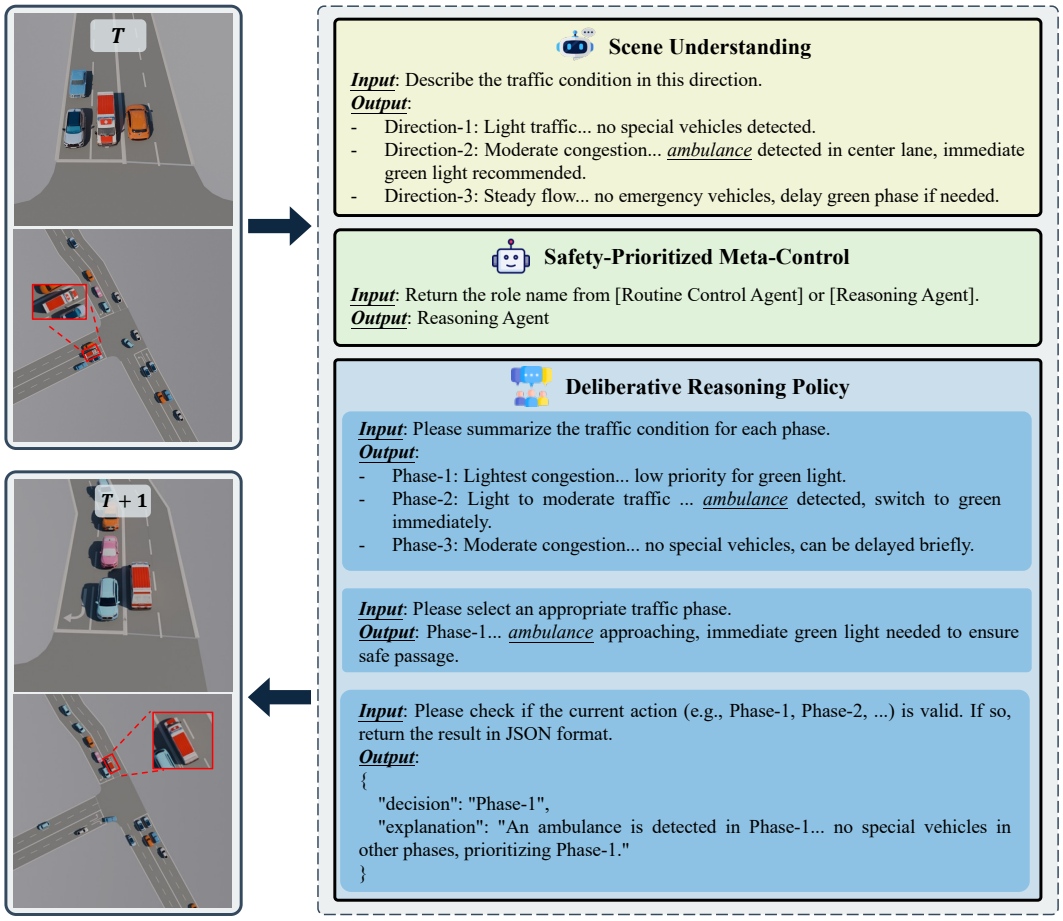


Figure 14: Deliberative Reasoning policy for complex traffic in Massy.

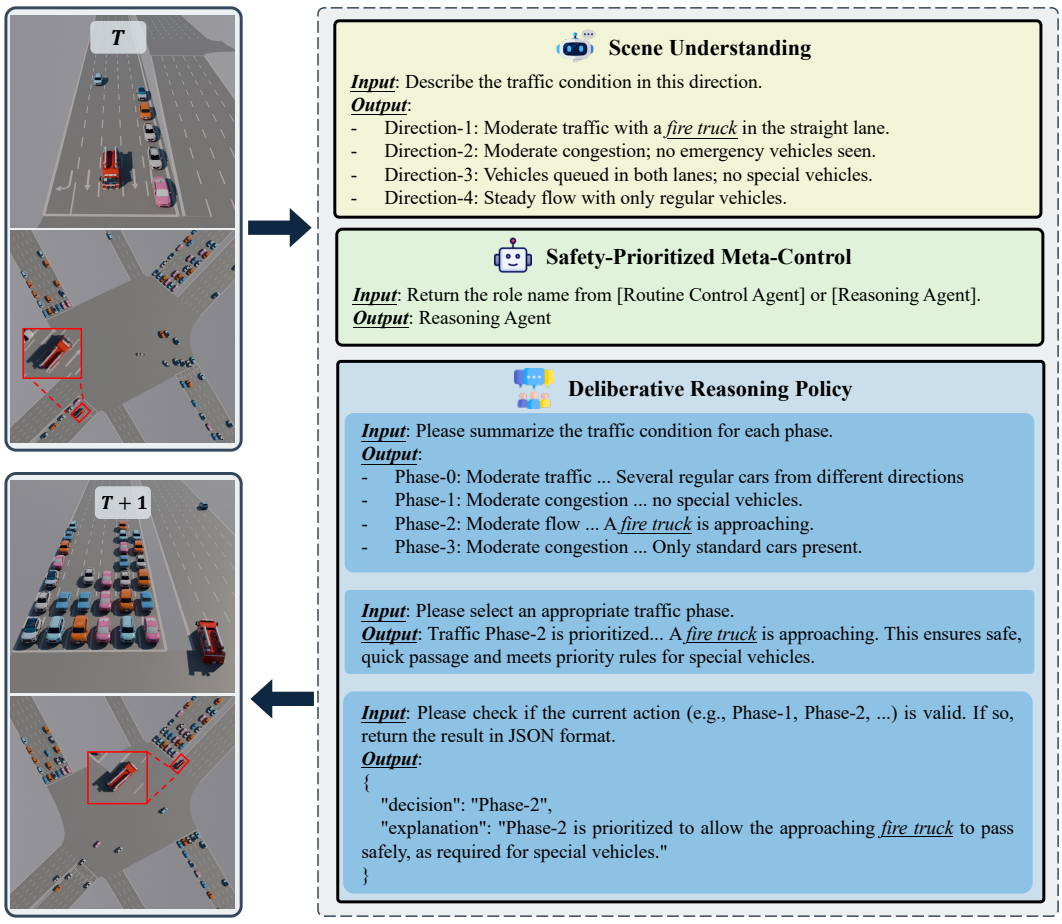


Figure 15: Deliberative Reasoning policy for complex traffic in Songdo.

D.1 Example 1: Routine Control in Yau Ma Tei

Figure 13 presents a routine scenario at the Yau Ma Tei intersection. The $\texttt{Agent}_{\text{Scene}}$ first transforms multi-view traffic images into structured language descriptions. Finding no anomalies or priority vehicles, the $\texttt{Agent}_{\text{ModeSelector}}$ routes the control to the RL branch. Based on the traffic density, the $\texttt{Agent}_{\text{RL}}$ selects Phase-2 (westbound) for the green signal. The transition from T to $T+1$ confirms that the westbound queue clears once Phase 2 is activated.

D.2 Example 2: Complex Scenario in Massy

Figure 14 showcases a special case at the Massy intersection, where an ambulance is detected on the west approach. The $\texttt{Agent}_{\text{Scene}}$ detects the emergency vehicle from the image inputs and generates descriptive observations. Recognizing a priority event, the $\texttt{Agent}_{\text{ModeSelector}}$ subsequently triggers the Deliberative Reasoning branch. The $\texttt{Agent}_{\text{Phase}}$ agent maps the scene to candidate signal phases, $\texttt{Agent}_{\text{Plan}}$ recommends Phase-1 for a green signal, and $\texttt{Agent}_{\text{Check}}$ verifies compliance with emergency-priority rules. By the time $T+1$, the ambulance has cleared the intersection through the northbound turn.

D.3 Example 3: Complex Scenario in Songdo

Figure 15 presents a complex case at the Songdo intersection. Similar to the previous Massy case, the $\texttt{Agent}_{\text{Scene}}$ identifies key cues (a fire truck approaching), and $\texttt{Agent}_{\text{ModeSelector}}$ activates the Deliberative Reasoning branch. The sequence of $\texttt{Agent}_{\text{Phase}}$, $\texttt{Agent}_{\text{Plan}}$, and $\texttt{Agent}_{\text{Check}}$ ensures a

compliant and safe control action. By the time $T + 1$, the fire truck moves through the intersection without interruption.

E Disscussion

VLMLight introduces a novel vision-language framework for TSC, combining real-time visual reasoning with safety and efficiency improvements. As the first open-source vision-based simulator in the TSC domain, VLMLight is compatible with RL-based TSC algorithms, offering a valuable resource for future research on perception-driven traffic systems. This framework enables enhanced scene understanding through multi-view visual perception and structured reasoning, ensuring both fast decision-making for routine traffic and reliable handling of critical scenarios like emergency vehicles.

However, VLMLight has several limitations. Firstly, the current simulator lacks diverse weather and lighting conditions, limiting its evaluation of visual robustness in challenging environments. Secondly, the absence of pedestrians, bicycles, and other real-world elements makes the simulated environment less realistic; incorporating more diverse models is necessary for future iterations. Third, all experiments have been conducted on single intersections, and extending the framework to multi-intersection scenarios requires further research on scalability and coordination in broader traffic networks. Lastly, VLMLight's performance is closely tied to VLM capabilities, with optimal results requiring models with numerous parameters. Future work will focus on fine-tuning smaller models to improve both traffic scene recognition accuracy and inference speed.