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:

$$\mathbf{m}_i^t = \left[F^{i,t}, O_{\text{max}}^{i,t}, O_{\text{mean}}^{i,t}, I_s^i, L_i, I_{\text{cg}}^{i,t}, I_{\text{mg}}^{i,t}\right]. \tag{1}$$

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 1. To capture temporal dynamics, the agent receives input from the current and previous four timesteps, forming a 5-frame observation window:

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

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:

$$\mathbf{h}_i^t = \mathbf{W}_e \mathbf{m}_i^t + \mathbf{b}_e, \quad i = 1, \ldots, 12, \tag{3}$$

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):

$$\mathbf{E}^t = \text{MLP}\left(\text{LN}\left(\mathbf{h}^t + \text{MSA}(\text{LN}(\mathbf{h}^t))\right)\right) \in \mathbb{R}^{12\times d}. \tag{4}$$

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

$$\mathbf{s}_t = \frac{1}{12}\sum_{i=1}^{12} \mathbf{E}_i^t \in \mathbb{R}^d. \tag{5}$$

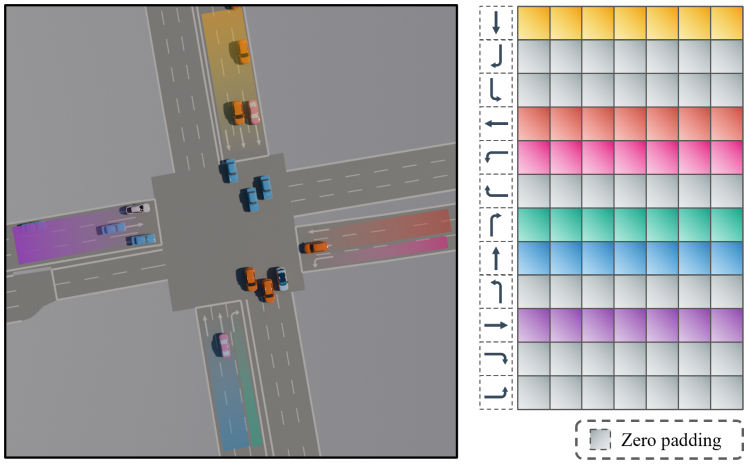


Figure 1: Example of zero-padding at the Yau Ma Tei intersection.

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:

$$\mathbf{Z}_t = \text{MLP}\left(\text{LN}\left(\mathbf{S}_t + \text{MSA}\left(\text{LN}(\mathbf{S}_t)\right)\right)\right) \in \mathbb{R}^{5\times d}. \tag{6}$$

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:

$$\mathbf{f}_t = \frac{1}{5}\sum_{k=1}^{5} \mathbf{Z}_{t,k} \in \mathbb{R}^d. \tag{7}$$

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:

$$\mathcal{L}^{\text{PPO}}(\theta) = \mathbb{E}_t\left[\min\left(r_t(\theta)\hat{A}_t,\ \text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t\right)\right], \tag{8}$$

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:

$$\mathcal{L}^{\text{value}}(\phi) = \mathbb{E}_t\left[\left(V(\mathbf{h}_t;\phi) - \hat{R}_t\right)^2\right], \tag{9}$$

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:

$$\mathcal{L}_{\text{total}}(\theta, \phi) = -\mathcal{L}_{\text{policy}}(\theta) + \lambda_v \mathcal{L}_{\text{value}}(\phi), \tag{10}$$

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 1 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 1: 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. 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 2. The other

four agents are based on LLMs: $\texttt{Agent}_{\text{ModeSelector}}$ determines the control mode (Figure 3), $\texttt{Agent}_{\text{Phase}}$ generates phase-level descriptions based on the scene context (Figure 4), $\texttt{Agent}_{\text{Plan}}$ selects the optimal phase (Figure 5), and $\texttt{Agent}_{\text{Check}}$ verifies rule compliance (Figure 6).

$Agent_{Scene}$

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 2: 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 3: 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 4: 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 5: Prompt template for $\texttt{Agent}_{\text{Plan}}$.

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

$Agent_{Check}$

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

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.

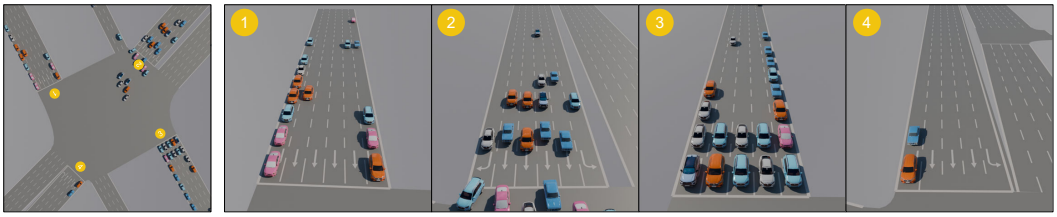
To ensure high-performance traffic signal control, we deploy both VLMs and LLMs locally on a high-performance computing system. The system is equipped with an Intel Xeon 6738P CPU, 256 GB of RAM, and five A100 GPUs, all running Ubuntu 20.04 LTS. This setup allows us to run multiple VLMs in parallel, 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 [1] 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.

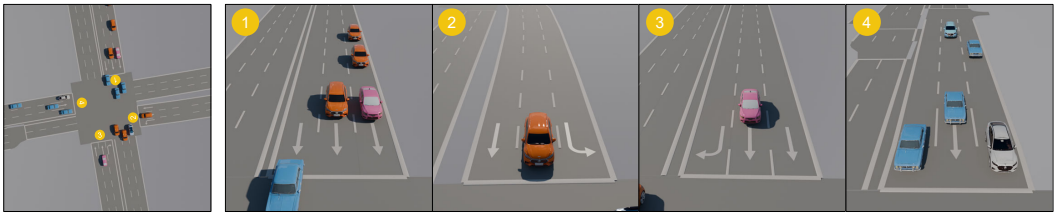
B.2 Dataset Details

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.

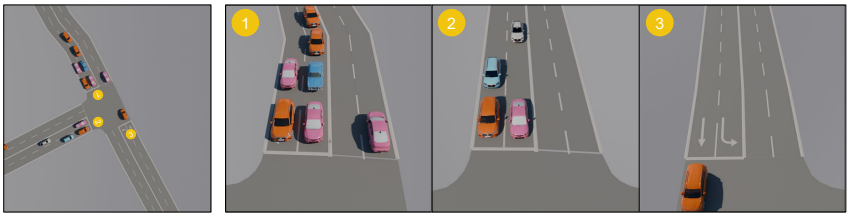
For each intersection, multi-directional cameras were deployed to capture 30 minutes of continuous traffic footage, with all approaches covered. As shown in Figure 7, the first column in each subfigure presents the top-down intersection layout, while subsequent images show direction-specific views from each inbound lane. These direction-wise visual inputs are fed into the VLMLight perception module for downstream reasoning. Table 2 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



(a)



(b)



(c)

Figure 7: 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.

sites, ensuring meaningful evaluation of safety-critical reasoning. This comprehensive setup enables reproducible and diverse benchmarking for vision-language-based traffic signal control.

Table 2: 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

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 [2]: 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 [3]: 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.

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

- IntelliLight [4]: 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 [5]: 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 [8]: 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.

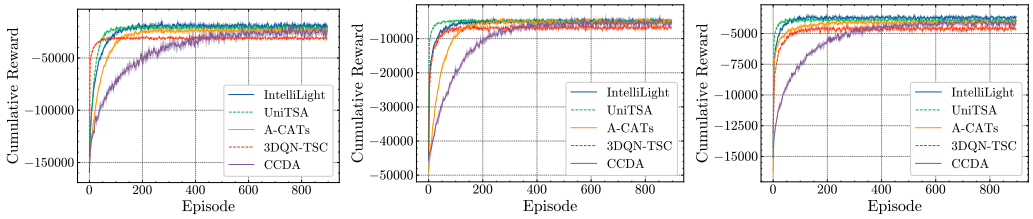


Figure 8: Training reward curves of five RL-based TSC methods across three real-world intersections: (a) Songdo (South Korea), (b) Yau Ma Tei (Hong Kong), and (c) Massy (France).

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 8 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 3: 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 4: 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 5: 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 [9], LLaVA-7B, LLaVA-13B [10], and GPT-4o [11] for the scene understanding agent ($\texttt{Agent}_{\text{Scene}}$), and Qwen2.5-7B, Qwen2.5-32B [12], Llama3.1-70B [13], and GPT-4o [11] for the $\texttt{Agent}_{\text{ModeSelector}}$ and reasoning policy agents. The results are presented in Tables 3, 4, and 5, 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 3, 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 4 and Table 5, 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 6, 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 6, 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 6: 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.

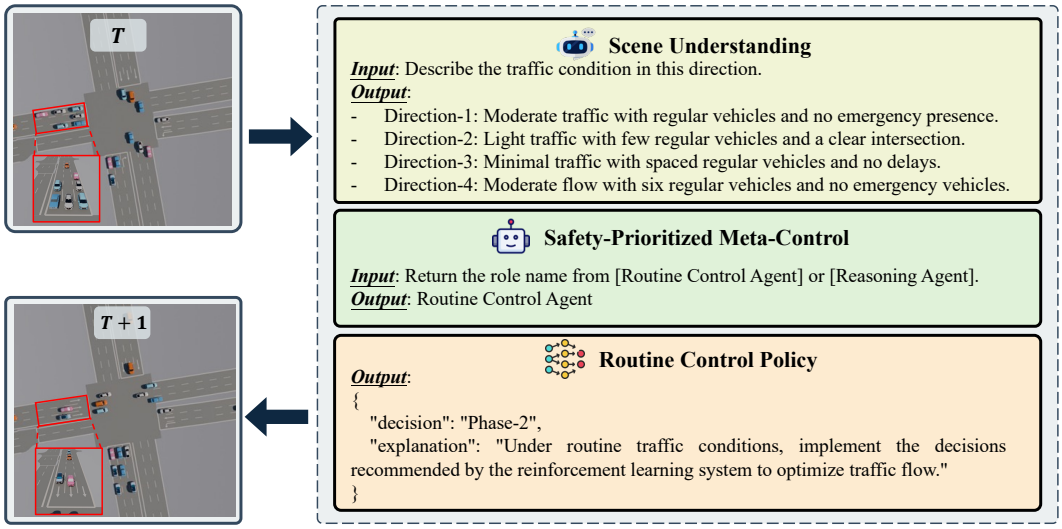


Figure 9: Routine Control in Yau Ma Tei.

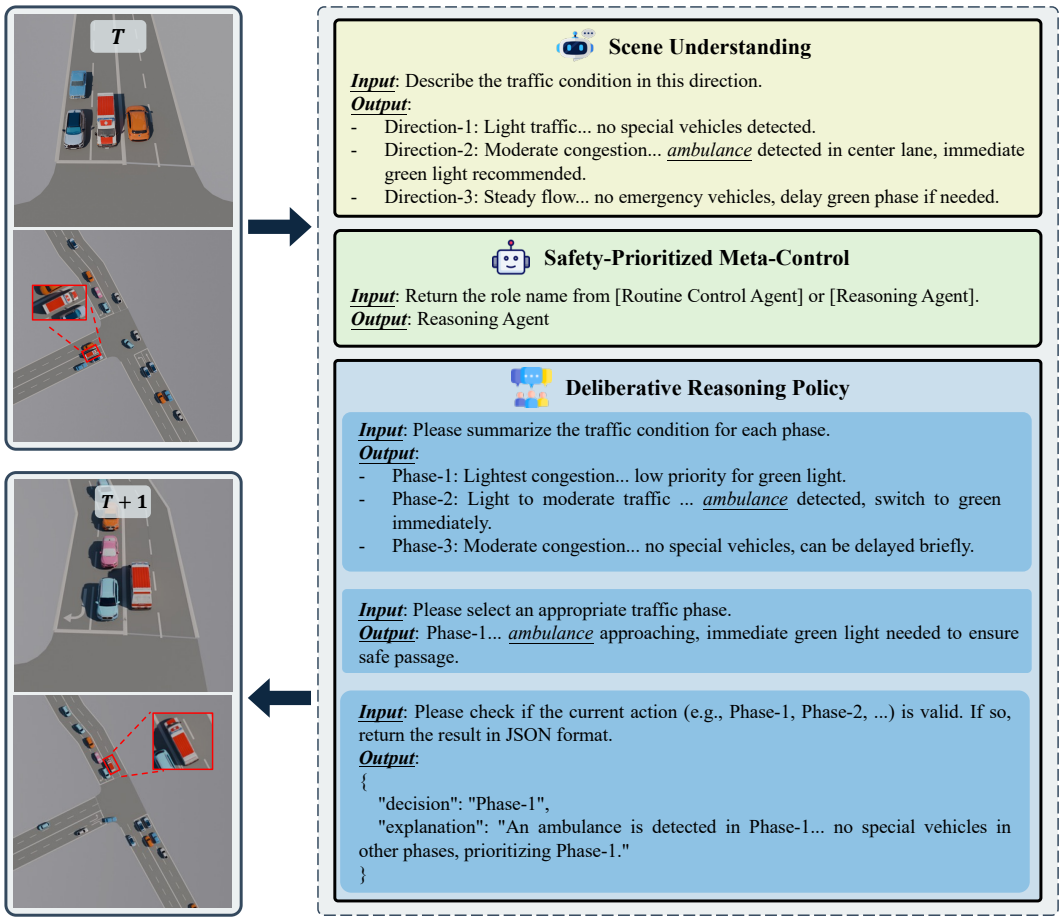


Figure 10: Deliberative Reasoning policy for complex traffic in Massy.

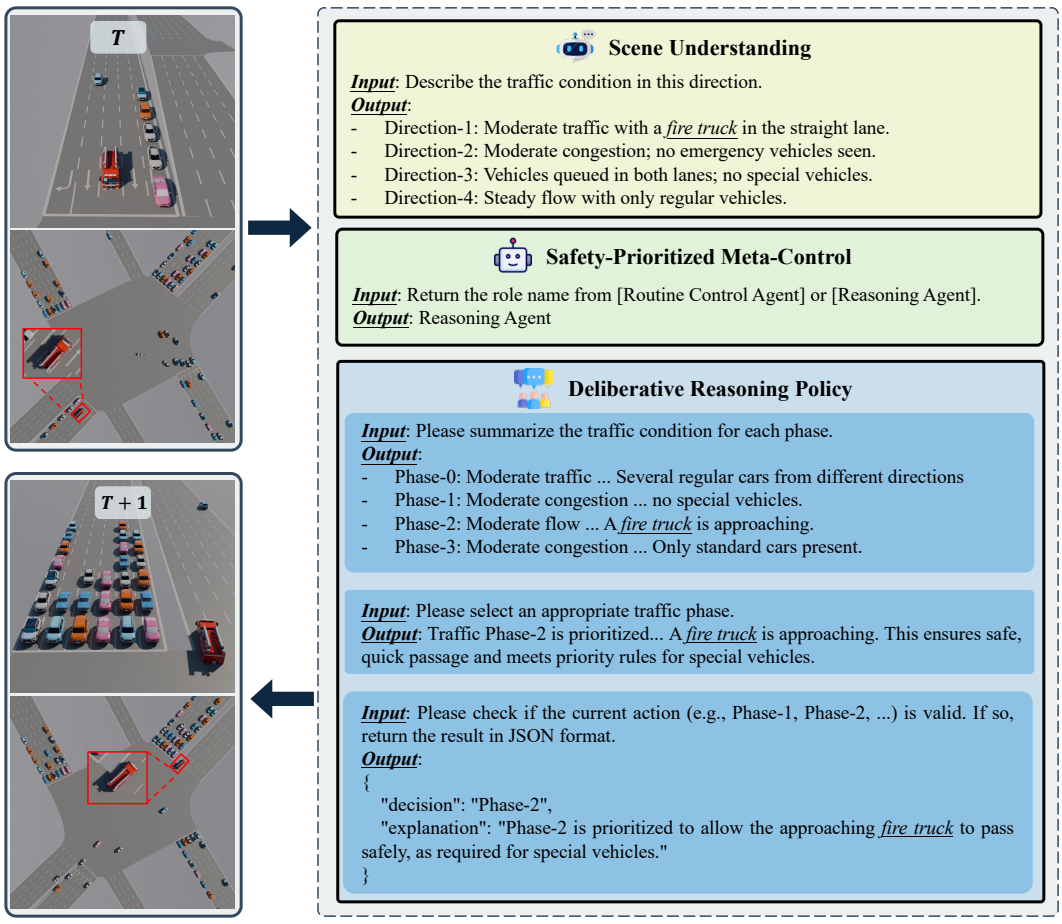


Figure 11: Deliberative Reasoning policy for complex traffic in Songdo.

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

Figure 9 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 10 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 11 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.

References

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

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

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

[4] 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.

[5] 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.

[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] 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.

[9] 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.

[10] 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.

[11] 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.

[12] 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.

[13] 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.