ETM: EVENT TRIGGER MODULE BASED VLM FOR AUTONOMOUS DRIVING

Anonymous authors
Paper under double-blind review

ABSTRACT

Large Vision Language Models (VLMs) have recently demonstrated remarkable semantic reasoning capabilities for autonomous driving and are increasingly incorporated into fast-slow planning frameworks. However, existing approaches invoke the VLM at every planning step even though high-level semantic changes are typically sparse over time, leading to substantial redundant computation. This raises a fundamental question: when should the expensive VLM be invoked? To address this problem, we propose the Event Trigger Module (ETM), a lightweight event-driven policy that selectively activates the VLM only when refreshed semantic reasoning is expected to improve downstream planning. We formulate VLM invocation as an information-gain-driven decision that explicitly balances semantic utility against computational cost. Guided by this formulation, ETM integrates short-term visual dynamics, long-term temporal context, and compact motion-aware scalar cues to estimate the necessity of VLM invocation. To train the trigger policy effectively, we further introduce a hybrid supervision strategy that combines corner-case trigger labels with information gain based pseudo-labels, enabling ETM to capture both common driving situations and rare but safety-critical events. Extensive experiments on the nuScenes benchmark demonstrate that the proposed ETM method achieves comparable planning accuracy to always-triggered VLM systems while reducing the VLM invocation rate from 100% to 32%. Meanwhile, ETM decreases inference latency by nearly 50% and lowers computational cost by approximately 64%, without sacrificing planning performance. These results show that the proposed ETM method offers an effective paradigm for VLM-assisted autonomous driving.

1 INTRODUCTION

The rapid advancement of large vision language models (VLMs) has prompted growing interest in integrating them into autonomous driving systems Tian et al. (2024); Sima et al. (2024); Xu et al. (2024); Liu et al. (2023); Radford et al. (2021); Li et al. (2023); Wang et al. (2024). A prominent paradigm emerging from this trend is the dual system, or fast-slow framework, in which a VLM serves as the slow system while an end-to-end driving network acts as the fast system. In such architectures, the VLM provides high-level semantic decisions or meta-actions that guide trajectory optimization in the fast system Ma et al. (2025); Qian et al. (2024; 2025); Jiang et al. (2024).

Several methods have explored different integration strategies within this paradigm. One line of work designs the analysis process and the heuristic process to operate alternately in a dual system Ma et al. (2025). Another approach employs adaptive gating units to fuse meta-action features with downstream planning representations, enabling soft guidance over trajectory generation Chen et al. (2024c).

Despite their effectiveness, these methods share a common implicit assumption: the VLM should be invoked at every inference step, regardless of the complexity or novelty of the driving scenario Chen et al. (2024c); Tang et al. (2025); Qian et al. (2025). We argue that this assumption is both computationally wasteful and empirically unwarranted.

To motivate our approach, we conducted an empirical analysis of publicly available autonomous driving datasets and visualized the distribution of meta-action outputs from existing VLM based planners. Figure 1 reveals that the vast majority of driving scenarios involve routine behaviors, such

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

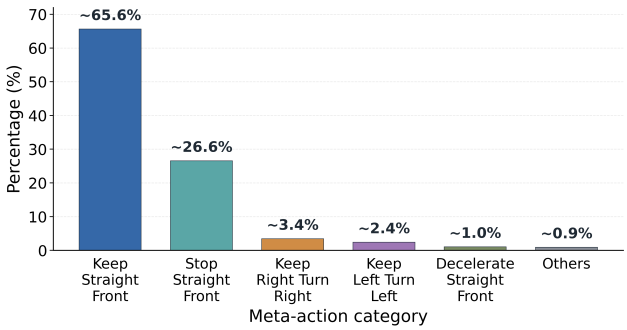


Figure 1: The Meta-action categories of the VLM output statistics in the nuScenes dataset

as lane keeping and speed following, safety-critical situations constituting only a small fraction of total frames. This imbalance suggests that invoking the VLM at every timestep incurs substantial computational overhead while providing little benefit for the majority of frames.

Inspired by event trigger control theory Tabuada (2007); Heemels et al. (2012), where a control action is executed only when a predefined triggering condition is met, we propose an Event Trigger Module (ETM) for VLM invocation in the fast-slow autonomous driving framework. Rather than invoke the VLM at a fixed-frequency, ETM dynamically determines whether the current scene warrants slow system intervention, thereby reducing unnecessary inference calls while reducing computational overhead.

Concretely, ETM operates on two complementary input streams. The first is a visual-trajectory feature stream encoding scene context from recent frames. The second consists of scalar auxiliary such as speed and yaw to capture safety relevant dynamics. These inputs are processed by a dual-path network comprising a fast path for real-time triggering decisions and a slow context path for temporally aggregated scene understanding. ETM is trained with a combination of hard labels based on corner-cases Chi et al. (2026) and pseudo-labels based on the information gain criterion Shannon (1948); MacKay (1992), encouraging calibrated triggering behavior across both common and rare events. The resulting module is designed as a plug-and-play component compatible with existing fast-slow driving architectures.

In summary, the contributions of this work are as follows:

1.We propose the ETM method. Based on the information gain criterion, we combine visual-trajectory features with scalar auxiliary signals, providing an informative representation for trigger condition estimation.

2.We design a dual-stream ETM structure to capture rapid triggering features as well as for long-term scene understanding.

3.We train ETM using a hybrid labeling scheme, processing hard labels from corner-case scenario dataset and pseudo-labels derived from information gain criterion, enabling the model to generalize across both safety-critical and routine driving conditions.

4.We evaluate ETM on the nuScenes benchmark through comprehensive ablation and comparative experiments, demonstrating that ETM substantially reduces VLM invocation frequency and overall system inference latency without degrading planning performance.

2 Related Work

2.1 End-to-End Autonomous Driving Planning

End-to-end autonomous driving has made substantial progress by unifying perception, prediction, and planning within a single network Chen et al. (2024a); Li et al. (2024); Liu et al. (2022); Renz et al. (2022). UniAD Hu et al. (2023) introduced a query centric architecture that aggregates multi task perception and prediction features into a unified planning objective. Building on this founda-

tion, VAD Jiang et al. (2023) and VADv2 Chen et al. (2024b) simplified the data pipeline through vectorized map representations, while SparseDrive Sun et al. (2025) achieved efficient parallel computation of perception and planning via sparse instance representations. Despite strong performance on standard benchmarks, these purely discriminative models struggle to generalize to long tail scenarios, where semantic reasoning beyond pattern matching is required. This limitation motivates our use of a fast end-to-end planner as the system backbone, and highlights the necessity of incorporating VLM based reasoning for handling complex edge cases Zhou et al. (2024).

2.2 VLM Augmented fast-slow System Frameworks

To address the semantic reasoning limitations of end-to-end models, recent work has explored integrating VLMs into autonomous driving systems Sima et al. (2024); Xu et al. (2024); Yang et al. (2025); Huang et al. (2024); Wang et al. (2025b). DriveVLM Tian et al. (2024) leverages a VLM for scene description, situation analysis, and hierarchical planning. Its extended variant, DriveVLM-Dual, runs a VLM in parallel with a conventional 3D perception-planning pipeline, forming a dual system architecture reminiscent of human fast-slow thinking. However, in DriveVLM-Dual, both systems operate continuously in parallel, meaning the VLM is invoked at every timestep, incurring significant computational overhead. LeapVAD Ma et al. (2025) draws on dual process theory from cognitive science Kahneman (2011), designing an alternating mechanism between an analytic process (System II) and a heuristic process (System I), further augmented by a memory module enabling closed loop continual learning. While these works demonstrate the value of slow, deliberative reasoning in autonomous driving, they do not address the fundamental question of when the slow system should be activated—instead treating VLM invocation as a fixed or always-on policy Huang et al. (2026); Fang et al. (2025); Zhou et al. (2026); Chen et al. (2025); Wang et al. (2025a); Han et al. (2025); Jiang et al. (2025). Our work focuses precisely on this underexplored question.

2.3 Adaptive Triggering and Frequency Control

Reducing VLM invocation frequency to lower inference cost has attracted increasing attention Zhang et al. (2025); Huang et al. (2025); Gopalkrishnan et al. (2024). AsyncDriver Chen et al. (2024c) proposed decoupling LLM inference from real-time planning by invoking the LLM at fixed asynchronous intervals. While effective at reducing overhead, fixed interval triggering is inherently inflexible: it may respond too slowly in safety-critical situations and wastefully in routine ones. FASIONAD Qian et al. (2024) and FASIONAD++ Qian et al. (2025) introduced an uncertainty estimation module that activates the slow system when the fast planner's uncertainty exceeds a predefined threshold, representing an early attempt to couple adaptive triggering with dual system planning. However, its triggering signal is derived from a single source. VLMPlanner Tang et al. (2025) proposed the CAI-Gate mechanism, which uses a lightweight network to estimate scene complexity and dynamically adjusts VLM invocation frequency. Its triggering decision relies solely on scene complexity, without accounting for information gain, or the cost of VLM invocation.

In contrast, the proposed Event Trigger Module (ETM) treats VLM invocation as an adaptive decision to balance information gain against cost, rather than a fixed process. Motivated by an information gain criterion, ETM estimates whether refreshing the cached meta-action can provide sufficient new semantic guidance to justify the computational cost of VLM inference. To this end, ETM jointly models short-term visual changes, long-term temporal context, and cache scalar cues. The estimated information gain score is then converted into a differentiable trigger probability through learnable threshold and parameters. During inference, ETM cooperates with a semantic cache to reuse valid meta-actions in routine scenes, while selectively activating the VLM when the cached semantic representation is likely to become stale or insufficient. Compared with prior fixed-frequency or single signal triggering strategies, ETM provides a more comprehensive mechanism for balancing semantic reasoning utility and computational efficiency.

3 Method

3.1 Overview

Figure 2 illustrates the overall framework of the proposed ETM for VLM-assisted end-to-end autonomous driving. Given the navigation command, map information, historical trajectory, and

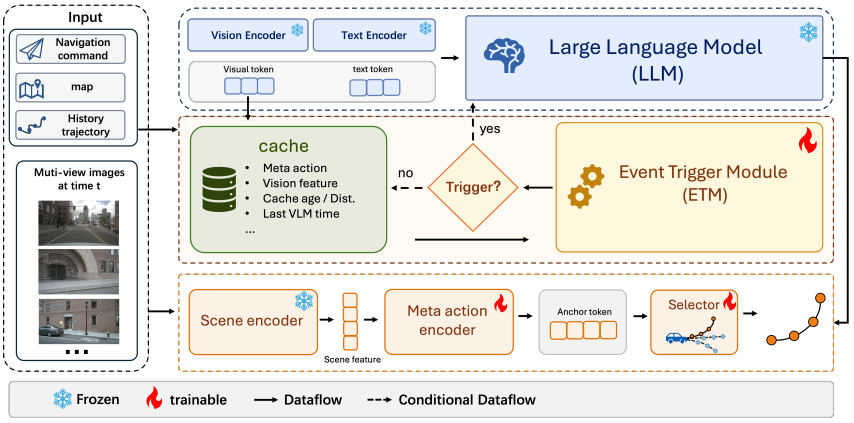


Figure 2: Overall framework of the proposed ETM. The frozen vision language model is invoked only when triggered by ETM to refresh the cached meta-action, while the planner continuously predicts trajectories using cached or newly generated semantic guidance.

multi-view images at time step t, our framework combines a frozen VLM Liu et al. (2023) with a lightweight event triggering policy and a meta-action conditioned planner to efficiently generate future driving trajectories.

The frozen visual and text encoders first extract visual and textual tokens, which are fed into the frozen large language model (LLM) to generate a high-level language based meta-action. Since such semantic guidance usually changes much more slowly than raw visual observations, repeatedly invoking the LLM at every frame leads to substantial redundant computation. To address this issue, we introduce an Event Trigger Module together with a semantic cache. ETM predicts a trigger probability indicating whether the LLM should be invoked and the cached meta-action should be refreshed. If triggered, the LLM generates a new meta-action and updates the cache; otherwise, the planner directly reuses the cached semantic representation.

The planning network operates continuously regardless of whether the LLM is activated. A trainable scene encoder extracts scene features from the current observations, which are fused with the cached or newly updated meta-action through a meta-action encoder. Finally, a selector predicts the future driving trajectory conditioned on both visual features and high-level semantic guidance. The following subsections present the meta-action conditioned planner, the proposed ETM, the trigger supervision, and the cache-based inference strategy in detail.

3.2 Meta-action Conditioned Planning

We represent the VLM output as a structured meta-action Jiang et al. (2024); Tian et al. (2024)

$$m_t = (s_t, d_t, a_t), \tag{1}$$

where $s_t \in \{1, \ldots, 4\}$ denotes the speed intent, including keeping speed, accelerating, decelerating, and stopping. $d_t \in \{1, \ldots, 5\}$ denotes the path level maneuver, and $a_t \in \{1, \ldots, 4\}$ denotes the attended spatial region. The three categorical variables are converted into meta-action tokens.

For vocabulary-based planning Chen et al. (2024b), we construct a trajectory anchor set $\mathcal{A} = \{A_k\}_{k=1}^{K}$ from the training trajectories using clustering, where each anchor $A_k \in \mathbb{R}^{T\times 2}$ represents a future ego trajectory. K represents the number of anchors, and T represents the time dimension. The planner encodes the meta-action categories into tokens and fuses them with the ego scene feature e_t to obtain:

$$H_t = [E_s(s_t), E_d(d_t), E_a(a_t), \phi(e_t)], \tag{2}$$

where E_s, E_d, E_a, ϕ represent the encoder composed of Multi Layer Perceptrons (MLP). Each trajectory anchor is projected into an anchor token and attends to H_t via multi head attention. The resulting score $q_{t,k}$ for anchor k is

$$q_{t,k} = f_{\text{score}}\left(\text{Attn}(\mathbf{Q} = \psi(A_k), \mathbf{K} = H_t, \mathbf{V} = H_t) + \psi(A_k)\right), \tag{3}$$

where ψ represents the projection head composed of MLPs, and f_{score} denotes the scoring head. The selected ego plan $\hat{Y}_t$ is the k_t-th anchor with the highest score:

$$k_t = \arg\max_k q_{t,k}, \qquad \hat{Y}_t = A_{k_t}. \tag{4}$$

To avoid over penalizing near equivalent trajectories, we train the anchor classifier with distance based soft labels. Let Y_t be the ground truth future trajectory and let $\text{ADE}(A_k, Y_t)$ be the average displacement error over valid future steps.τ_a is a temperature hyperparameter. The soft target is

$$\pi_{t,k} = \frac{\exp(-\text{ADE}(A_k, Y_t)/\tau_a)}{\sum_j \exp(-\text{ADE}(A_j, Y_t)/\tau_a)}, \tag{5}$$

and the anchor classification loss is

$$\mathcal{L}_{\text{anchor}} = -\sum_{k=1}^{K} \pi_{t,k} \log \text{softmax}(q_t)_k. \tag{6}$$

3.3 Event Trigger Module

ETM determines whether the current frame requires a fresh VLM meta-action Tabuada (2007); Heemels et al. (2012). The trigger decision is motivated by information gain criterion, the VLM should be invoked only when refreshing the cached semantic guidance provides sufficient new information for planning Shannon (1948); MacKay (1992).

Since the fresh VLM output is unavailable before invoking the VLM. ETM therefore uses the pre-invocation VLM input, together with visual dynamics and cache state, as observable cues to predict the potential utility of semantic refresh. Specifically, ETM learns a differentiable information-gain-inspired refresh-utility score

$$\hat{ig}_t = f_\theta\left(I_t, \Delta I_t, S_{\text{cache}}\right), \tag{8}$$

where I_t denotes the current VLM input, ΔI_t captures temporal visual dynamics, and S_{cache} represents the cached semantic state. The learned score serves as a practical surrogate for the ideal information gain rather than a direct measurement of $IG_{\text{ideal}}(t)$.

The final trigger decision is given by

$$u_t = \mathbb{I}\left[\hat{ig}_t > \tau\right], \tag{9}$$

where $\mathbb{I}[\cdot]$ denotes the indicator function and τ is a tunable threshold controlling the trade-off between semantic refresh frequency and downstream planning performance. The detailed architecture of ETM is illustrated in Figure 3.

Specifically, ETM uses a dual-stream temporal architecture. The fast stream observes the most recent $W_f = 4$ visual features to capture abrupt scene changes, while the slow stream observes $W_s = 8$ down sampled features from a longer temporal context with stride 2. Let $v_t \in \mathbb{R}^D$ be the visual feature of the current frame. The fast and slow inputs are

$$F_t = [v_{t-W_f+1}, \ldots, v_t], \quad S_t = [v_{t-2(W_s-1)}, \ldots, v_t]. \tag{7}$$

Both streams are first projected by ρ_f and ρ_s and then encoded by GRUs:

$$h_t^f = \text{GRU}_f(\rho_f(F_t)), \quad H_t^s = \text{GRU}_s(\rho_s(S_t)). \tag{8}$$

The fast hidden state queries the slow temporal states through cross attention:

$$c_t = \text{Attn}(\mathbf{Q} = h_t^f, \mathbf{K} = H_t^s, \mathbf{V} = H_t^s). \tag{9}$$

In addition to visual dynamics, ETM uses an 8-dimensional scalar state vector:

$$x_t = [\Delta_{\cos}, H(m), \bar{v}, |\bar{\omega}|, \bar{n}, \bar{l}, \bar{o}, \mathbb{I}(n \geq N_{\max})], \tag{10}$$

where $\Delta_{\cos}$ is the cosine distance between the current visual feature and the cached feature, $H(m)$ is the cached meta-action entropy, $\bar{v}$ is the normalized ego speed, $|\bar{\omega}|$ is the normalized yaw rate, $\bar{n}$

is the normalized number of frames since the last trigger, $\bar{l}$ is the normalized traveled distance since the last trigger, $\bar{o}$ is the normalized object count, and $\mathbb{I}(n \geq N_{\max})$ indicates whether the cache has reached the maximum allowed age.

The fused ETM representation is

$$z_t = [c_t, H_t^s, \eta(x_t)], \tag{11}$$

where $\eta(\cdot)$ is a lightweight MLP for scalar state encoding. ETM then estimates the information gain score as

$$\hat{g}_t = \sigma(f_{\text{ig}}(z_t)), \tag{12}$$

where $f_{\text{ig}}(\cdot)$ denotes a lightweight prediction head that maps the fused representation to an information gain logit, and $\sigma(\cdot)$ is the sigmoid function. The differentiable trigger probability is computed by

$$p_t = \sigma\left(\frac{\hat{g}_t - \theta}{T}\right), \tag{13}$$

where θ and T are learnable threshold and temperature parameters. At inference time, the VLM is invoked and cache is refreshed when $p_t > \delta$ or when the cache exceeds the maximum age or traveled distance.

3.4 Trigger Supervision

We generate ETM supervision offline using an automatic pseudo-labeling pipeline. A frame is labeled as trigger if any of the following events occur, it belongs to a corner-case set Chi et al. (2026), the visual feature drift from the cached frame exceeds a threshold, the ego motion changes abruptly, the cached meta-action times out, or the previous VLM description contains safety-critical keywords. Each positive frame is also assigned a risk score $r_t \in [0, 1]$, which reflects the severity of the refresh signal. The exact pseudo-label generation procedure and risk score construction are provided in Appendix.

ETM is trained with a safety oriented asymmetric focal loss Lin et al. (2017); Ridnik et al. (2021). For binary label $y_t \in \{0, 1\}$ and predicted trigger probability p_t, we define

$$\mathcal{L}_{\text{trig}} = \alpha_t (1 - p_t^*)^\gamma \text{BCE}(p_t, y_t), \tag{14}$$

where $p_t^* = p_t$ if $y_t = 1$ and $p_t^* = 1 - p_t$ otherwise, and γ is the focal focusing parameter that emphasizes hard trigger examples while down weighting easy ones. Missed triggers are penalized more heavily:

$$\alpha_t = \begin{cases} \lambda_{\text{miss}}(1 + r_t), & y_t = 1 \text{ and } p_t < 0.5, \\ 1, & \text{otherwise}, \end{cases} \tag{15}$$

where λ_{miss} is a penalty coefficient that increases the cost of false negative trigger predictions. We additionally regularize the information gain prediction:

$$\mathcal{L}_{\text{ig}} = \|\hat{g}_t - r_t\|_2^2 . \tag{16}$$

The final ETM objective is

$$\mathcal{L}_{\text{ETM}} = \mathcal{L}_{\text{trig}} + \lambda_{\text{ig}} \mathcal{L}_{\text{ig}}, \tag{17}$$

where λ_{ig} balances trigger classification and information gain regression.

3.5 Cache-based Inference

During online inference, ETM maintains a scene cache Bang (2023); Kwon et al. (2023) containing the last VLM text, categorical meta-action probabilities, encoded meta-action feature, visual feature, cache age, and traveled distance. For each frame, the visual encoder is always executed to obtain v_t, whereas the VLM is invoked only on triggered frames. The planner consumes a semantic meta-action m_t, which is either generated by a fresh VLM query or retrieved from the scene cache:

$$m_t = \begin{cases} \text{VLM}(I_t), & p_t > \delta \text{ or } \text{Expired}(C_t), \\ C_t.m, & \text{otherwise}, \end{cases} \tag{18}$$

where I_t denotes the input image at time step t, $\text{VLM}(I_t)$ denotes the semantic meta-action generated by the VLM, C_t denotes the current scene cache, and $C_t.m$ is the cached semantic meta-action. Here, δ is the trigger threshold. The cache is forcibly refreshed when its age or accumulated ego travel distance exceeds predefined limits, which prevents unbounded semantic staleness and improves safety in long stable segments.

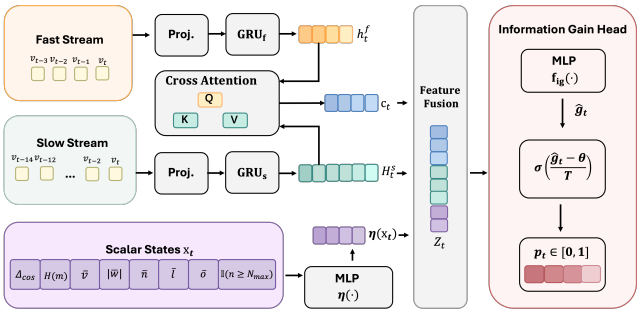
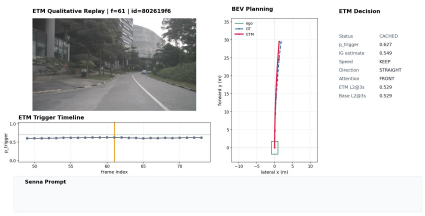


Figure 3: Architecture of the proposed ETM method. A dual-stream temporal encoder models short and long-term scene dynamics, which are fused with cache scalar states to estimate the information gain and predict the trigger probability.

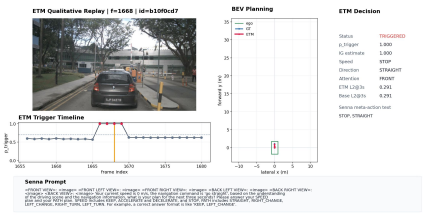
Table 1: Comparison with state-of-the-art methods on the nuScenes dataset. L2 (m) and Col (%) denote L2 displacement error and collision rate at the given horizon. Trigger indicates the VLM invocation rate.

Method	L2 (m) ↓			Col (%) ↓			Trigger
	1s	2s	3s	1s	2s	3s	
IL	0.44	1.15	2.47	0.08	0.27	1.95	—
UniAD	0.48	0.96	1.65	0.05	0.17	0.71	—
VAD-Base	0.41	0.70	1.05	0.07	0.17	0.41	—
FASIONAD	0.19	0.62	1.25	0.02	0.09	0.44	—
Senna	0.37	0.54	0.86	0.09	0.12	0.33	100%
ETM-full	0.38	0.64	0.96	0.03	0.14	0.29	100%
ETM-trigger (Ours)	**0.38**	**0.64**	**0.96**	**0.03**	**0.13**	**0.27**	**32%**

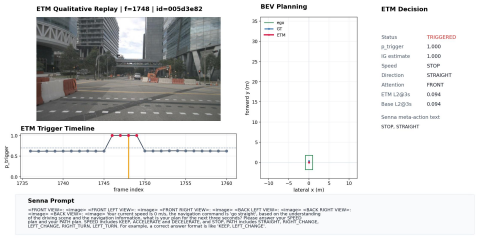
4 EXPERIMENTS



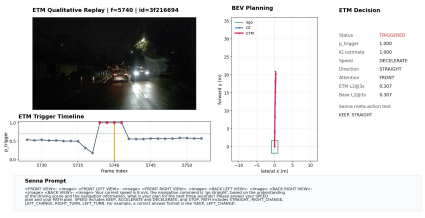
(a) Easy driving scene.



(b) Occluded scene.



(c) Signalized intersection scene.



(d) Multi-agent interaction scene.

Figure 4: Qualitative analysis of the proposed ETM under representative driving scenarios. ETM suppresses unnecessary VLM invocation in simple scenes while activating semantic reasoning only when additional information is expected to improve planning.

Table 2: Efficiency comparison under different VLM trigger strategies. Avg. latency and GFLOP/frame reflect the per-frame inference cost. GPU memory (GB) is measured during evaluation.

Strategy	Trigger Rate	Avg. Latency (ms)	GFLOP/frame	GPU Mem. (GB)	L2 3s ↓	Col 3s ↓
Always	1.00	574.84	16632.75	14.34	0.96	0.29
Fixed (k=5)	0.21	250.08	4129.35	14.34	1.10	0.34
Fixed (k=3)	0.35	305.63	6338.75	14.34	1.07	0.33
Random (k=3.4)	0.36	310.27	6503.48	14.34	1.08	0.36
ETM (δ=0.5)	0.88	518.31	14739.72	14.34	0.99	0.30
ETM (δ=0.6)	0.65	427.32	11149.77	14.34	0.98	0.30
ETM (δ=0.7)	**0.32**	**296.84**	**5946.55**	14.34	**0.96**	**0.27**

Table 3: Ablation study on ETM trigger module components. Trigger F1, Recall, and Miss Rate evaluate the trigger classifier, while Trigger Rate, Plan L2 3s, and Col 3s evaluate planning quality.

Variant	Trigger F1 ↑	Recall ↑	Miss Rate ↓	Trigger Rate ↓	Plan L2 3s ↓	Col 3s ↓
Full ETM	**0.904**	**0.975**	**0.025**	**0.321**	**0.964**	**0.267**
w/o Fast	0.896	0.947	0.053	0.342	0.987	0.296
w/o Slow	0.904	0.961	0.039	0.350	0.970	0.267
w/o Scalars	0.696	0.910	0.090	0.788	0.965	0.293
Scalar-only	0.861	0.943	0.056	0.617	0.985	0.296
w/o IG loss	0.901	0.905	0.095	0.491	0.987	0.296

4.1 Experimental Setup

Dataset. Experiments are conducted on the nuScenes benchmark Caesar et al. (2020) using the official train/validation split following prior end-to-end planning work Jiang et al. (2023).

Implementation Details. ETM is integrated into a VLM-assisted planning framework with the visual encoder and VLM frozen during training. Only ETM and the planner are optimized using the hybrid supervision described in Section 3. During inference, the cached meta-action is refreshed when the predicted trigger probability exceeds the threshold δ or the cache expires. The framework is implemented in PyTorch and trained on two NVIDIA GeForce RTX 4090 GPUs.

Evaluation Metrics. Planning performance is evaluated using the standard planning L2 displacement error (L2) and collision rate (Col) at prediction horizons of 1 s, 2 s, and 3 s Jiang et al. (2023). Trigger efficiency is evaluated by the VLM trigger rate, defined as the percentage of inference frames that invoke the VLM, together with average inference latency, GFLOPs per frame, and GPU memory consumption. For trigger prediction, we additionally report Trigger F1, Recall, and Miss Rate.

4.2 Comparison with State-of-the-Art

Table 1 compares ETM with representative end-to-end and VLM-assisted planning methods. Compared with conventional end-to-end planners such as IL, UniAD, and VAD-Base, the proposed ETM achieves substantially lower planning errors and collision rates across all prediction horizons. In particular, ETM reduces the 3 s collision rate from 0.41% (VAD-Base) to 0.27%, indicating that event trigger semantic reasoning provides more reliable guidance in safety-critical situations.

Compared with Senna, which invokes the VLM at every frame, ETM-trigger achieves nearly identical planning accuracy while reducing the collision rate from 0.33% to 0.27%. More importantly, ETM-trigger reduces the VLM invocation frequency from 100% to only 32%, demonstrating that semantic reasoning is required for only a small subset of driving frames.

A particularly important comparison is between ETM-full and ETM-trigger. Both methods share exactly the same planning architecture and VLM backbone, differing only in whether ETM selectively triggers the VLM. Remarkably, ETM-trigger maintains identical planning accuracy (0.96 m at 3 s) while slightly improving the collision rate (0.27% vs. 0.29%), despite invoking the VLM less than one-third as often. This result validates the central hypothesis of this work: continuously invoking a VLM is unnecessary for most driving scenarios, and event trigger semantic updates are sufficient to preserve planning quality.

4.3 Efficiency Analysis

Table 2 compares different VLM triggering strategies in terms of computational efficiency and planning performance. Naive triggering strategies reduce computational cost but degrade planning quality. For example, Fixed ($k = 5$) decreases the trigger rate to 21% but increases the 3 s planning error from 0.96 m to 1.10 m and the collision rate from 0.29% to 0.34%. Random triggering exhibits a similar trend, indicating that simply reducing VLM invocations is insufficient.

In contrast, ETM achieves a substantially better efficiency–performance trade-off by selectively triggering the VLM. Increasing the trigger threshold from 0.5 to 0.7 reduces the trigger rate from 88% to 32%, decreasing latency from 518.31 ms to 296.84 ms while reducing computational cost by nearly 64%.

Notably, ETM with $\delta = 0.7$ matches the planning accuracy of the always-trigger baseline while approximately halving latency and GFLOPs, and further reduces the collision rate from 0.29% to 0.27%. These results demonstrate that ETM effectively identifies informative frames, enabling efficient semantic reasoning without sacrificing planning performance.

4.4 Ablation Study

Table 3 evaluates the contribution of each component in ETM. Removing either the fast or slow temporal branch degrades trigger prediction, confirming that both short- and long-term temporal cues are important. Specifically, removing the fast branch reduces Recall from 0.975 to 0.947, while removing the slow branch increases the trigger rate from 32.1% to 35.0%.

Scalar auxiliary features are also essential. Without them, the Trigger F1 score drops from 0.904 to 0.696 and the trigger rate rises to 78.8%, indicating that visual features alone are insufficient. Using only scalar features likewise degrades planning performance, demonstrating the benefit of combining visual and scalar information.

Finally, removing information gain supervision reduces Recall (0.975 $\rightarrow$ 0.905), increases the trigger rate (32.1% $\rightarrow$ 49.1%), and degrades planning performance (L2: 0.964 m $\rightarrow$ 0.987 m; Col: 0.267% $\rightarrow$ 0.296%). These results validate the effectiveness of the proposed supervision for accurate trigger prediction.

4.5 Qualitative Analysis

Figure 4 presents four representative scenarios illustrating ETM's context-aware triggering behavior. ETM suppresses VLM invocation and reuses cached semantics in routine road-following scenes, while activating the VLM in occluded regions, complex intersections, and dense multi-agent interactions where additional semantic cues are beneficial for planning. These examples demonstrate that ETM selectively refreshes semantic guidance in ambiguous or semantically complex situations while avoiding unnecessary VLM inference during routine driving.

5 Conclusion

We presented ETM, an event-triggered semantic reasoning framework that invokes VLMs only when sufficient information gain is expected. On nuScenes, ETM reduces VLM calls to about one-third of frames while maintaining comparable planning accuracy and safety. Future work will explore adaptive triggering with larger multimodal models in closed-loop driving.

REFERENCES

Fu Bang. Gptcache: An open-source semantic cache for llm applications enabling faster answers and cost savings. In *Proceedings of the 3rd Workshop for Natural Language Processing Open Source Software (NLP-OSS 2023)*, pp. 212–218, 2023.

Holger Caesar, Varun Bankiti, Alex H Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuscenes: A multimodal dataset for autonomous driving. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 11621–11631, 2020.

Li Chen, Penghao Wu, Kashyap Chitta, Bernhard Jaeger, Andreas Geiger, and Hongyang Li. End-to-end autonomous driving: Challenges and frontiers. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(12):10164–10183, 2024a.

Shaoyu Chen, Bo Jiang, Hao Gao, Bencheng Liao, Qing Xu, Qian Zhang, Chang Huang, Wenyu Liu, and Xinggang Wang. Vadv2: End-to-end vectorized autonomous driving via probabilistic planning. *arXiv preprint arXiv:2402.13243*, 2024b.

Xuesong Chen, Linjiang Huang, Tao Ma, Rongyao Fang, Shaoshuai Shi, and Hongsheng Li. Solve: Synergy of language-vision and end-to-end networks for autonomous driving. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 12068–12077, 2025.

Yuan Chen, Zi-han Ding, Ziqin Wang, Yan Wang, Lijun Zhang, and Si Liu. Asynchronous large language model enhanced planner for autonomous driving. In *European Conference on Computer Vision*, pp. 22–38. Springer, 2024c.

Haohan Chi, Huan-ang Gao, Ziming Liu, Jianing Liu, Chenyu Liu, Jinwei Li, Kaisen Yang, Yangcheng Yu, Zeda Wang, Wenyi Li, et al. Impromptu vla: Open weights and open data for driving vision-language-action models. *Advances in Neural Information Processing Systems*, 38, 2026.

Shiyu Fang, Yiming Cui, Haoyang Liang, Chen Lv, Peng Hang, and Jian Sun. Corevla: A dual-stage end-to-end autonomous driving framework for long-tail scenarios via collect-and-refine. *arXiv preprint arXiv:2509.15968*, 2025.

Akshay Gopalkrishnan, Ross Greer, and Mohan Trivedi. Multi-frame, lightweight & efficient vision-language models for question answering in autonomous driving. *arXiv preprint arXiv:2403.19838*, 2024.

Wencheng Han, Dongqian Guo, Cheng-Zhong Xu, and Jianbing Shen. Dme-driver: Integrating human decision logic and 3d scene perception in autonomous driving. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 3347–3355, 2025.

Wilhelmus PMH Heemels, Karl Henrik Johansson, and Paulo Tabuada. An introduction to event-triggered and self-triggered control. In *2012 ieee 51st ieee conference on decision and control (cdc)*, pp. 3270–3285. IEEE, 2012.

Yihan Hu, Jiazhi Yang, Li Chen, Keyu Li, Chonghao Sima, Xizhou Zhu, Siqi Chai, Senyao Du, Tianwei Lin, Wenhai Wang, et al. Planning-oriented autonomous driving. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 17853–17862, 2023.

Jin Huang, Yuchao Jin, Le An, and Josh Park. Litevlm: A low-latency vision-language model inference pipeline for resource-constrained environments. *arXiv preprint arXiv:2506.07416*, 2025.

Zhijian Huang, Chengjian Feng, Feng Yan, Baihui Xiao, Zequn Jie, Yujie Zhong, Xiaodan Liang, and Lin Ma. Drivemm: All-in-one large multimodal model for autonomous driving. *arXiv preprint arXiv:2412.07689*, 2(3):8, 2024.

Zilin Huang, Zihao Sheng, Zhengyang Wan, Yansong Qu, Junwei You, Sicong Jiang, and Sikai Chen. Drivevlm-rl: Neuroscience-inspired reinforcement learning with vision-language models for safe and deployable autonomous driving. *arXiv preprint arXiv:2603.18315*, 2026.

Anqing Jiang, Yu Gao, Zhigang Sun, Yiru Wang, Jijun Wang, Jinghao Chai, Qian Cao, Yuweng Heng, Hao Jiang, Yunda Dong, et al. Diffvla: Vision-language guided diffusion planning for autonomous driving. *arXiv preprint arXiv:2505.19381*, 2025.

Bo Jiang, Shaoyu Chen, Qing Xu, Bencheng Liao, Jiajie Chen, Helong Zhou, Qian Zhang, Wenyu Liu, Chang Huang, and Xinggang Wang. Vad: Vectorized scene representation for efficient autonomous driving. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 8340–8350, 2023.

Bo Jiang, Shaoyu Chen, Bencheng Liao, Xingyu Zhang, Wei Yin, Qian Zhang, Chang Huang, Wenyu Liu, and Xinggang Wang. Senna: Bridging large vision-language models and end-to-end autonomous driving. *arXiv preprint arXiv:2410.22313*, 2024.

Daniel Kahneman. Fast and slow thinking. *Allen Lane and Penguin Books, New York*, 2, 2011.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the 29th symposium on operating systems principles*, pp. 611–626, 2023.

Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. In *International conference on machine learning*, pp. 19730–19742. PMLR, 2023.

Zhiqi Li, Wenhai Wang, Hongyang Li, Enze Xie, Chonghao Sima, Tong Lu, Qiao Yu, and Jifeng Dai. Bevformer: learning bird's-eye-view representation from lidar-camera via spatiotemporal transformers. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 47(3):2020–2036, 2024.

Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pp. 2980–2988, 2017.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. *Advances in neural information processing systems*, 36:34892–34916, 2023.

Yingfei Liu, Tiancai Wang, Xiangyu Zhang, and Jian Sun. Petr: Position embedding transformation for multi-view 3d object detection. In *European conference on computer vision*, pp. 531–548. Springer, 2022.

Yukai Ma, Tiantian Wei, Naiting Zhong, Jianbiao Mei, Tao Hu, Licheng Wen, Xuemeng Yang, Botian Shi, and Yong Liu. Leapvad: A leap in autonomous driving via cognitive perception and dual-process thinking. *IEEE Transactions on Neural Networks and Learning Systems*, 2025.

David JC MacKay. Information-based objective functions for active data selection. *Neural computation*, 4(4):590–604, 1992.

Kangan Qian, Zhikun Ma, Yangfan He, Ziang Luo, Tianyu Shi, Tianze Zhu, Jiayin Li, Jianhui Wang, Ziyu Chen, Xiao He, et al. Fasionad: Fast and slow fusion thinking systems for human-like autonomous driving with adaptive feedback. *arXiv preprint arXiv:2411.18013*, 2024.

Kangan Qian, Ziang Luo, Sicong Jiang, Zilin Huang, Jinyu Miao, Zhikun Ma, Tianze Zhu, Jiayin Li, Yangfan He, Zheng Fu, et al. Fasionad++: Integrating high-level instruction and information bottleneck in fat-slow fusion systems for enhanced safety in autonomous driving with adaptive feedback. *arXiv preprint arXiv:2503.08162*, 2025.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pp. 8748–8763. PmLR, 2021.

Katrin Renz, Kashyap Chitta, Otniel-Bogdan Mercea, A Koepke, Zeynep Akata, and Andreas Geiger. Plant: Explainable planning transformers via object-level representations. *arXiv preprint arXiv:2210.14222*, 2022.

Tal Ridnik, Emanuel Ben-Baruch, Nadav Zamir, Asaf Noy, Itamar Friedman, Matan Protter, and Lihi Zelnik-Manor. Asymmetric loss for multi-label classification. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 82–91, 2021.

Claude Elwood Shannon. A mathematical theory of communication. *The Bell system technical journal*, 27(3):379–423, 1948.

Chonghao Sima, Katrin Renz, Kashyap Chitta, Li Chen, Hanxue Zhang, Chengen Xie, Jens Beißwenger, Ping Luo, Andreas Geiger, and Hongyang Li. Drivelm: Driving with graph visual question answering. In *European conference on computer vision*, pp. 256–274. Springer, 2024.

Wenchao Sun, Xuewu Lin, Yining Shi, Chuang Zhang, Haoran Wu, and Sifa Zheng. Sparsedrive: End-to-end autonomous driving via sparse scene representation. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 8795–8801. IEEE, 2025.

Paulo Tabuada. Event-triggered real-time scheduling of stabilizing control tasks. *IEEE Transactions on Automatic control*, 52(9):1680–1685, 2007.

Zhipeng Tang, Sha Zhang, Jiajun Deng, Chenjie Wang, Guoliang You, Yuting Huang, Xinrui Lin, and Yanyong Zhang. Vlmplanner: Integrating visual language models with motion planning. In *Proceedings of the 33rd ACM International Conference on Multimedia*, pp. 5040–5049, 2025.

Xiaoyu Tian, Junru Gu, Bailin Li, Yicheng Liu, Yang Wang, Zhiyong Zhao, Kun Zhan, Peng Jia, Xianpeng Lang, and Hang Zhao. Drivevlm: The convergence of autonomous driving and large vision-language models. *arXiv preprint arXiv:2402.12289*, 2024.

Dingrui Wang, Marc Kaufeld, and Johannes Betz. Dualad: Dual-layer planning for reasoning in autonomous driving. In *2025 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 12057–12063. IEEE, 2025a.

Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, et al. Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution. *arXiv preprint arXiv:2409.12191*, 2024.

Shihao Wang, Zhiding Yu, Xiaohui Jiang, Shiyi Lan, Min Shi, Nadine Chang, Jan Kautz, Ying Li, and Jose M Alvarez. Omnidrive: A holistic vision-language dataset for autonomous driving with counterfactual reasoning. In *Proceedings of the computer vision and pattern recognition conference*, pp. 22442–22452, 2025b.

Zhenhua Xu, Yujia Zhang, Enze Xie, Zhen Zhao, Yong Guo, Kwan-Yee K Wong, Zhenguo Li, and Hengshuang Zhao. Drivegpt4: Interpretable end-to-end autonomous driving via large language model. *IEEE Robotics and Automation Letters*, 9(10):8186–8193, 2024.

Xuemeng Yang, Licheng Wen, Tiantian Wei, Yukai Ma, Jianbiao Mei, Xin Li, Wenjie Lei, Daocheng Fu, Pinlong Cai, Min Dou, et al. Drivearena: A closed-loop generative simulation platform for autonomous driving. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 26933–26943, 2025.

Ruifei Zhang, Junlin Xie, Wei Zhang, Weikai Chen, Xiao Tan, Xiang Wan, and Guanbin Li. Adadrive: Self-adaptive slow-fast system for language-grounded autonomous driving. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 5112–5121, 2025.

Xingcheng Zhou, Mingyu Liu, Ekim Yurtsever, Bare Luka Zagar, Walter Zimmer, Hu Cao, and Alois C Knoll. Vision language models in autonomous driving: A survey and outlook. *IEEE Transactions on Intelligent Vehicles*, 2024.

Xingcheng Zhou, Xuyuan Han, Feng Yang, Yunpu Ma, Volker Tresp, and Alois Knoll. Opendrivevla: Towards end-to-end autonomous driving with large vision language action model. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 13782–13790, 2026.

A APPENDIX

A.1 DETAILS OF EVENT TRIGGER SUPERVISION

This appendix provides the exact construction of the offline supervision used to train the Event Trigger Module (ETM). We emphasize that the trigger labels are not manually annotated human labels. Instead, they are automatically generated pseudo-labels derived from safety-critical events, visual state changes, ego-motion changes, and cache expiration rules. The goal of this supervision is to teach ETM when the cached VLM-derived meta-action is likely to become stale and should be refreshed.

A.1.1 NOTATION

For each frame t, let I_t denote the multi-view camera observation and let $v_t \in \mathbb{R}^D$ denote the pre-extracted ViT feature of the frame. Let $p_t^{ego} \in \mathbb{R}^2$ denote the ego-vehicle position in the global coordinate system. We generate a binary trigger label $y_t \in \{0, 1\}$ and a continuous risk score $r_t \in [0, 1]$ for ETM training.

The label $y_t = 1$ means that the current frame is considered an event frame and the cached meta-action should be refreshed. The label $y_t = 0$ means that the previous meta-action can be reused.

A.1.2 FEATURE DRIFT EVENT

We first detect abrupt visual changes using the cosine distance between consecutive ViT features:

$$d_t^{feat} = 1 - \frac{v_t^\top v_{t-1}}{\|v_t\|_2 \|v_{t-1}\|_2}. \tag{19}$$

A feature drift event is triggered when

$$e_t^{feat} = \mathbb{I}\left[d_t^{feat} > \tau_{feat}\right], \tag{20}$$

where we use

$$\tau_{feat} = 0.25. \tag{21}$$

This rule captures sudden changes in the visual scene, such as newly visible objects, occlusions, or rapid viewpoint changes.

A.1.3 EGO-MOTION CHANGE EVENT

We also detect large ego-motion changes using the displacement between consecutive ego positions:

$$d_t^{ego} = \left\|p_t^{ego} - p_{t-1}^{ego}\right\|_2. \tag{22}$$

The ego-motion event is defined as

$$e_t^{ego} = \mathbb{I}\left[d_t^{ego} > \tau_{ego}\right], \tag{23}$$

where

$$\tau_{ego} = 6.0 \text{ meters}. \tag{24}$$

This rule identifies frames where the vehicle state changes substantially, which may invalidate the previously cached high-level intention.

A.1.4 TIMEOUT EVENT

Even when the scene appears stable, the cached meta-action should not be reused indefinitely. We therefore define a timeout event based on the number of frames since the last positive trigger:

$$e_t^{time} = \mathbb{I}\left[\Delta t_t \geq \tau_{time}\right], \tag{25}$$

where Δt_t is the number of frames since the last trigger. We use

$$\tau_{time} = 4 \text{ frames}. \tag{26}$$

For nuScenes-style frame intervals of approximately 0.5 seconds, this corresponds to about 2 seconds. This rule prevents unbounded semantic staleness of the cached VLM meta-action.

A.1.5 CORNER-CASE EVENT

We further incorporate external corner-case samples from the Impromptu VLA annotation/index file. Let $\mathcal{C}$ denote the set of sample tokens identified as corner cases. The corner-case event is defined as

$$e_t^{corner} = \mathbb{I}\left[\text{token}(t) \in \mathcal{C}\right]. \tag{27}$$

When this rule is activated, the frame is treated as high-risk because it corresponds to a rare or safety-critical driving situation.

A.1.6 KEYWORD-BASED SEMANTIC EVENT

If a cached VLM text description is available, we additionally inspect it for safety-critical keywords. Let q_{t-1} denote the previous cached VLM text. We use the keyword set

$$\begin{aligned}\mathcal{K} = \{&\texttt{unclear boundary}, \texttt{temporary traffic}, \texttt{unusual obstacle},\\ &\texttt{adverse}, \texttt{construction}, \texttt{emergency}, \texttt{sudden},\\ &\texttt{pedestrian crossing unexpectedly}, \texttt{poor visibility}\}.\end{aligned} \tag{28}$$

The keyword event is

$$e_t^{key} = \mathbb{I}\left[\exists k \in \mathcal{K},\ k \subseteq \text{lower}(q_{t-1})\right]. \tag{29}$$

This rule encourages ETM to refresh meta-actions when the previously generated language already indicates a potentially unstable or safety-sensitive scene.

A.1.7 FINAL TRIGGER LABEL

The final trigger label is the logical union of all event sources:

$$y_t = e_t^{corner} \vee e_t^{feat} \vee e_t^{ego} \vee e_t^{time} \vee e_t^{key}. \tag{30}$$

Equivalently,

$$y_t = \mathbb{I}\left[e_t^{corner} + e_t^{feat} + e_t^{ego} + e_t^{time} + e_t^{key} > 0\right]. \tag{31}$$

The first frame of each scene is initialized with a large cache age, which forces a trigger and initializes the meta-action cache.

A.1.8 RISK SCORE CONSTRUCTION

In addition to the binary trigger label, we assign a continuous risk score $r_t \in [0, 1]$ to each frame. Negative frames are assigned zero risk:

$$r_t = 0, \qquad \text{if } y_t = 0. \tag{32}$$

For each positive frame, the risk score is initialized as

$$r_t = 0.2, \qquad \text{if } y_t = 1, \tag{33}$$

and is subsequently updated according to the activated event types:

$$r_t = \begin{cases} 1.0, & e_t^{corner} = 1, \\ \max(r_t, d_t^{feat}), & e_t^{feat} = 1, \\ \max(r_t, 0.8), & e_t^{key} = 1, \\ r_t, & \text{otherwise.} \end{cases} \tag{34}$$

In our implementation, ego-motion and timeout events affect the binary trigger label but do not further increase the risk score unless they co-occur with feature drift, corner-case, or keyword events. This design makes the binary label responsible for cache validity, while the risk score emphasizes high-severity semantic or visual events.

A.1.9 TRAINING USAGE

The generated pair (y_t, r_t) is used to supervise ETM. The binary label trains the trigger probability p_t, while the risk score supervises the information-gain estimate $\hat{g}_t$:

$$\mathcal{L}_{ig} = \|\hat{g}_t - \mathrm{clip}(r_t, 0, 1)\|_2^2 . \tag{35}$$

The trigger classifier is trained with an asymmetric focal loss that assigns larger weights to missed positive triggers:

$$\mathcal{L}_{trig} = w_t(1 - p_t^*)^\gamma \mathrm{BCE}(p_t, y_t), \tag{36}$$

where

$$p_t^* = \begin{cases} p_t, & y_t = 1, \\ 1 - p_t, & y_t = 0, \end{cases} \tag{37}$$

and

$$w_t = \begin{cases} \lambda_{miss}(1 + r_t), & y_t = 1 \text{ and } p_t < 0.5, \\ 1, & \text{otherwise.} \end{cases} \tag{38}$$

We use $\gamma = 2.0$ and $\lambda_{miss} = 5.0$. The final ETM loss is

$$\mathcal{L}_{ETM} = \mathcal{L}_{trig} + \lambda_{ig}\mathcal{L}_{ig}, \tag{39}$$

with $\lambda_{ig} = 0.3$.

A.1.10 DISCUSSION

The generated labels should be interpreted as event-trigger pseudo-labels rather than human-annotated ground truth. This is intentional: the purpose of ETM is not to recognize a fixed semantic taxonomy, but to learn a practical refresh policy for VLM-derived meta-actions. The supervision combines visual change, ego-motion change, cache expiration, and safety-critical semantic cues, which together approximate whether the cached meta-action is still reliable.

A.2 IMPLEMENTATION DETAILS

A.2.1 COMPUTING INFRASTRUCTURE

All experiments were conducted on two NVIDIA GeForce RTX 4090 GPUs with 24 GB memory. The machine uses an Intel Core i9-14900K CPU and 125 GiB system memory. The operating system is Ubuntu 20.04 with Linux kernel 5.15.0.

A.2.2 SOFTWARE ENVIRONMENT

The VAD-based planner was trained with Python 3.8, PyTorch 1.9.1+cu111, CUDA 11.1, cuDNN 8.0.5, TorchVision 0.10.1, MMCV 1.4.0, MMDetection 2.14.0, MMSegmentation 0.14.1, and MMDetection3D 0.17.1. The Senna/LLaVA meta-action generation and feature extraction components were run with Python 3.10, PyTorch 2.1.2+cu121, CUDA 12.1, cuDNN 8.9.2, TorchVision 0.16.2, Transformers 4.37.2, Accelerate 1.13.0, Tokenizers 0.15.2, SentencePiece 0.2.1, and NumPy 1.26.4.

A.2.3 SENNA META-ACTION MODEL

Senna uses Vicuna-7B-v1.5 as the language backbone and CLIP ViT-L/14-336 as the vision encoder. Six surround-view camera images are used as visual input, with 128 visual tokens per image. During meta-action generation, we use greedy decoding with temperature 0, top-p disabled, one beam, and a maximum generation length of 64 tokens. The generated driving intent is parsed into three discrete meta-action factors: speed action, path action, and attention direction.

For Senna instruction tuning, the full-parameter setting uses batch size 8 per device, gradient accumulation 3, learning rate 2×10^{-5}, weight decay 0, cosine learning-rate scheduling, warmup ratio 0.03, one epoch, bf16 mixed precision, and gradient checkpointing. The projector pretraining stage uses batch size 32, learning rate 10^{-3}, weight decay 0, cosine scheduling, warmup ratio 0.03, one epoch, and bf16 mixed precision.

A.2.4 VAD PLANNER TRAINING

The planner is based on VAD with a ResNet-50 image backbone and BEVFormer-style spatiotemporal encoding. The BEV grid size is 200×200, the hidden dimension is 256, the feed-forward dimension is 512, and the temporal queue length is 4. The model uses 300 object queries and 100 map-vector queries. The planning horizon is 6 future steps.

We initialize from the released VAD base checkpoint and train a stage-1 meta-action planning model. In this stage, the perception, map, and motion components are frozen, while only the meta-action embeddings and ego-planning-related modules are trainable. This gives approximately 2.20M trainable parameters out of 58.96M total parameters, i.e., 3.73%. VAD stage-1 training uses AdamW with learning rate 1×10^{-4}, weight decay 0.01, batch size 3, 20 epochs, cosine annealing, linear warmup for 500 iterations with warmup ratio $1/3$, and gradient clipping with maximum norm 35. Mixed precision is not enabled for VAD training. The random seed is 0, with deterministic CUDA behavior disabled.

The original VAD losses are retained. The main loss weights are: detection classification 2.0, detection box L1 0.25, trajectory L1 0.2, trajectory classification 0.2, map classification 2.0, map point regression 1.0, map direction 0.005, planning regression 1.0, planning boundary 1.0, planning collision 1.0, and planning direction 0.5.

A.2.5 META-ACTION FUSION

Meta-action information is injected at the planning stage. Speed, path, and attention actions are embedded as discrete tokens with hidden dimension 256. A learned meta-action type embedding distinguishes the three token types. The ego planning feature is projected into the same space and appended as a scene token. We use 4096 learned trajectory anchors, each representing a 6-step future trajectory. Anchor tokens attend to the three meta-action tokens and the scene token through multi-head attention with 8 heads and dropout 0.1. The attended anchor tokens are scored by an MLP, and the highest-scoring anchor is selected as the planning prior. Anchor selection is additionally supervised with a soft-label anchor classification loss, where the soft label is computed from the anchor ADE to the ground-truth ego trajectory using temperature 0.5. The anchor classification loss weight is 1.0.

A.2.6 ETM TRIGGER MODEL

ETM decides whether to execute Senna meta-action generation or reuse the cached meta-action. Each frame is represented by a 1024-dimensional visual feature extracted from the Senna vision encoder. ETM uses a dual-stream temporal architecture: a fast stream over the latest 4 frames and a slow stream over 8 frames with stride 2, covering a 16-frame temporal range. The fast stream projects features to 64 dimensions and uses a one-layer GRU. The slow stream projects features to 128 dimensions and uses a two-layer GRU with dropout 0.1. In the full model, the fast hidden state attends to the slow-stream outputs through 4-head cross-attention.

ETM also uses an 8-dimensional scalar feature vector, including visual feature drift, action entropy, ego speed, yaw rate, time since last refresh, ego distance since last refresh, object count, and a forced-refresh flag. The scalar feature is encoded by a two-layer MLP into a 32-dimensional vector and fused with the temporal features before predicting information gain. ETM predicts an information-gain value $\hat{g}$, and the trigger probability is computed as

$$p_{\text{trigger}} = \sigma\left(\frac{\hat{g} - \tau}{T}\right), \tag{40}$$

where the threshold τ and temperature T are learnable parameters initialized to 0.5 and 0.1. At inference time, the external trigger threshold is swept over $\{0.5, 0.6, 0.7, 0.8, 0.9\}$. The cache is forcibly refreshed when its age exceeds 4 frames or the accumulated ego displacement exceeds 20 meters.

ETM is trained with AdamW using batch size 64, learning rate 3×10^{-4}, weight decay 1×10^{-4}, 20 epochs, cosine annealing, and gradient clipping with maximum norm 1.0. Mixed precision is not used. The data split seed is 42, with 10% of the training labels held out for validation. The ETM loss is

$$\mathcal{L} = \mathcal{L}_{cls} + \alpha_{info}\mathcal{L}_{info}, \tag{41}$$

where $\alpha_{info} = 0.3$. $\mathcal{L}_{cls}$ is an asymmetric focal loss with $\gamma = 2.0$. False negatives are penalized using a miss penalty of 5.0, scaled by the frame risk score. $\mathcal{L}_{info}$ regresses $\hat{g}$ to the normalized risk score during ETM training. In the ablation without information-gain supervision, α_{info} is set to 0.

A.2.7 EXPERIMENTAL PROTOCOL

Unless otherwise stated, each setting is run once with the fixed seeds specified above. Random-refresh baselines use seed 42. The ETM checkpoint is selected on the validation split by the best F1 score subject to recall exceeding 0.90. The VAD planning experiments use the stage-1 checkpoint evaluated in the experiment scripts, with epoch 14 used for the main efficiency evaluation. We compare always-refresh, fixed-interval refresh, random refresh, and ETM-triggered refresh. The fixed intervals are $k \in \{3, 5, 8, 10\}$, the random refresh probabilities are $p \in \{0.34, 0.20, 0.10\}$, and ETM thresholds are swept over $\{0.5, 0.6, 0.7, 0.8, 0.9\}$. The main selection criterion is the planning accuracy-efficiency trade-off on the nuScenes validation split, using planning L2 error, trigger rate, latency, and GPU memory.