Claude Code-Driving Scenario Mining for the Argoverse 2 Challenge

Wei Deng, Caoshengzhe Xue, Shuaikun Liu, Zhaohong Liu, Mengshi Qi*, Huadong Ma
State Key Laboratory of Networking and Switching Technology,
Beijing University of Posts and Telecommunications, China
{dw-dengwei,xuecsz,liushuaikun1,liuzhaoh,qms,mhd}@bupt.edu.cn

Abstract

We present our submission to the CVPR 2026 Argoverse 2 Scenario Mining Challenge. Our system uses a four-stage pipeline: (1) autonomous code generation via a `Claude Code` *agent powered by* `GLM 5.1` *[2], (2) iterative training set screening with Timestamp Balanced Accuracy threshold 0.8 to curate few-shot examples, (3) semantic code review by a separate* `Claude Code` *session, and (4)* `Qwen3-VL` *[3] scene-level verification to filter false positives. We report results on the Argoverse 2 test set.*

1. Introduction

The Argoverse 2 Scenario Mining Challenge is a CVPR 2026 competition tasking participants with localizing objects and temporal segments in autonomous driving logs given natural language descriptions. Built on the RefAV dataset [1], which comprises 10,000 planning-centric queries spanning 1,000 driving logs from the Argoverse 2 Sensor Dataset, the competition features two tracks: *Spatio-Temporal Localization* (metric: HOTA-Temporal) and *Temporal Localization* (metric: Timestamp Balanced Accuracy).

We present a four-stage scenario mining pipeline. First, we use `Claude Code` as an autonomous coding agent powered by `GLM 5.1` [2] to generate scenario code from natural language prompts. Second, we iteratively screen generated code on the training set, retaining only prompt-code pairs that achieve Timestamp Balanced Accuracy above 0.8 (up to 5 iterations) as few-shot examples. Third, a separate `Claude Code` session reviews generated code for semantic correctness and false positive risks. Fourth, we introduce a `Qwen3-VL` [3] verification stage that performs binary scene-level classification to filter false positives.

Our approach differs from existing work in two key aspects: (1) using an autonomous agent paradigm rather than single-pass LLM API calls, and (2) employing VLM-based verification as a post-execution filter that directly reduces false positive outputs.

*Corresponding author.

2. Method

We present a four-stage scenario mining pipeline built on the RefAV [1] atomic function framework. Our approach leverages an LLM-powered coding agent to autonomously generate scenario code, iteratively refines it using training set feedback, performs semantic code review, and validates outputs with a vision-language model to minimize false positives.

2.1. Background: Composable Atomic Functions

The RefAV framework provides a library of *atomic functions* that describe spatial relations, kinematics, and map attributes. Each function is decorated with `@composable` or `@composable_relational`, which handles parallelization over track candidates.

Spatial functions include `has_objects_in_relative_direction` (front, rear, left, right), `facing_toward`, `heading_toward`, `near_objects`, and `being_crossed_by`. Kinematic functions include `turning`, `changing_lanes`, `accelerating`, `has_velocity`, and `stationary`. Map functions include `at_stop_sign`, `at_pedestrian_crossing`, `near_intersection`, `on_road`, and `in_drivable_area`. Logical combinators `scenario_and`, `scenario_or`, and `scenario_not` allow Boolean composition.

2.2. LLM Agent Code Generation

Unlike prior work that uses LLM API calls to generate code in a single pass, we employ `Claude Code` as an autonomous coding agent powered by `GLM 5.1` [2]. The agent iteratively queries atomic function definitions using tool calls, analyzes the prompt requirements, and constructs the appropriate function composition. This agent-based approach allows the model to explore the function library, understand parameter semantics through tool-driven docu-

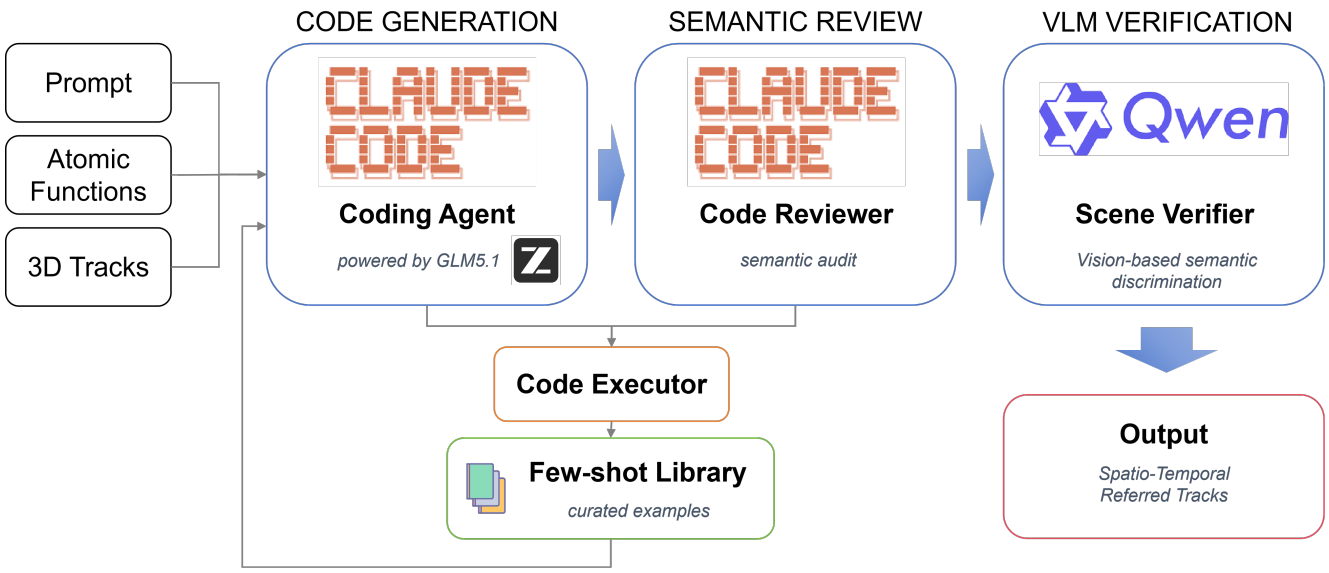

Figure 1. Overview of our framework.

mentation lookup, and verify its own output before finalizing the code.

2.3. Training Set Iterative Screening

We perform iterative screening on the RefAV training set to build a high-quality few-shot example library. For each prompt, the system:

1. Generates candidate code via the LLM agent.
2. Executes the code on the training split and computes Timestamp Balanced Accuracy.
3. If the score exceeds 0.8, the prompt-code pair is retained as a few-shot example.
4. If not, the code is regenerated for up to 5 iterations before being discarded.

This process produces a curated set of prompt-code pairs that serve as context examples in subsequent inference, improving code quality for unseen prompts.

2.4. Semantic Code Review

Before execution-based verification, a separate `Claude Code` session reviews each generated code snippet. The reviewer checks for semantic mismatches between the prompt and the code logic, identifies potential false positive sources (e.g., incorrect relationship direction, missing spatial constraints), and suggests or applies corrections. This two-tier agent design—generation followed by review—ensures both syntactic validity and semantic fidelity.

2.5. VLM Scene-Level Verification

Generated scenario outputs may still contain false positives when the LLM hallucinates object interactions. To address this, we introduce a verification stage using `Qwen3-VL` [3], a vision-language model. For each prompt-log pair, Qwen3-VL performs binary classification on the driving frames to determine whether the described event actually occurs.

If the VLM confirms the event, the code execution result is retained. Otherwise, the trajectory output is discarded and an empty prediction is produced. This effectively filters out false positive scenarios that pass the code execution stage but do not correspond to real events in the sensor data.

3. Experiments

3.1. Experimental Setup

Dataset. We evaluate on the Argoverse 2 Sensor Dataset test split using the RefAV [1] scenario mining annotations. The evaluation covers four metrics: HOTA-Temporal, HOTA-Track, Timestamp Balanced Accuracy (Timestamp BA), and Log Balanced Accuracy (Log BA).

Code generation agent. We use `Claude Code` as the coding agent with `GLM 5.1` [2] as the underlying language model. The agent performs tool-driven queries to explore the atomic function library before generating code.

VLM verification. `Qwen3-VL` [3] performs binary scene classification on driving frames to determine whether the described event occurs.

Trackers. We obtain trajectories from the generated code using the Le3DE2E [4] tracking model.

3.2. Training Set Iterative Screening

On the RefAV training set, we apply iterative code generation with up to 5 rounds per prompt. Code achieving Timestamp BA above 0.8 is retained as a few-shot example. This

Table 1. EvalAI test set leaderboard.

Rank	Team	HOTA-Temp	HOTA-Trk	TS BA	Log BA
1	bigbang	31.05	41.80	71.47	72.03
2	blackTea	30.87	41.98	72.39	75.23
3	lazydogs	30.43	41.76	71.23	73.32
4	Lilly	30.03	38.09	72.33	75.52
5	**MICTeam (Ours)**	**27.91**	**37.88**	**69.65**	**69.32**
6	RefProg	26.27	36.18	68.07	70.46
7	SM-Agent	23.25	31.15	66.95	67.66
8	bem	18.77	25.90	63.69	65.62

yields a small curated set of high-quality prompt-code pairs that serve as context in subsequent inference.

3.3. Code Review Quality

A separate `Claude Code` session reviews all generated code for semantic mismatches. The review identifies common error patterns including: incorrect relationship direction in relational functions, missing implicit map constraints, and unrealistic parameter thresholds. Corrected code replaces the original when issues are detected.

3.4. Main Results

Table 1 shows the EvalAI test set leaderboard with all participating teams. Our team (MICTeam) ranks 5th with a HOTA-Temporal of 27.91, Timestamp BA of 69.65, and Log BA of 69.32. The top-4 teams achieve competitive scores within a narrow margin on HOTA-based metrics, while our approach outperforms the official baselines (RefProg and SM-Agent) by a clear margin.

4. Conclusion

We presented a Claude-Code-driven scenario mining system for the CVPR 2026 Argoverse 2 Scenario Mining Challenge. Our approach combines composable atomic functions with automated code generation, validated few-shot examples, scene-level vision-based verification, and iterative refinement. We achieved 27.91 scores in the HOTA-Temp metric.

References

[1] Cainan Davidson, Deva Ramanan, and Neehar Peri. Refav: Towards planning-centric scenario mining. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2026. 1, 2

[2] GLM-5-Team. Glm-5: from vibe coding to agentic engineering, 2026. 1, 2

[3] Qwen-Team. Qwen3-vl technical report, 2025. 1, 2

[4] Zhepeng Wang, Feng Chen, Kanokphan Lertniphonphan, Siwei Chen, Jinyao Bao, Pengfei Zheng, Jinbao Zhang, Kaer Huang, and Tao Zhang. Technical report for argoverse challenges on unified sensor-based detection, tracking, and forecasting, 2023. 2