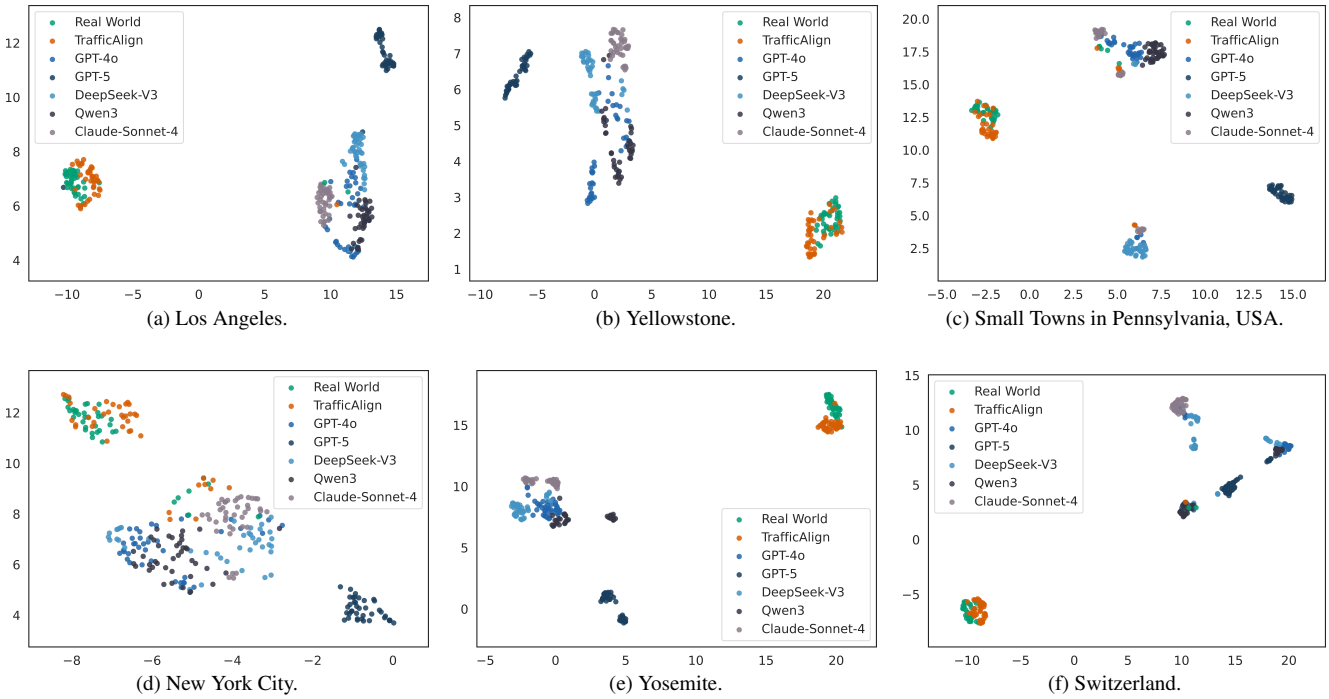
# TrafficAlign: Aligning Large Language Models for Traffic Scenario Generation

Supplementary Material

## A. Qualitative Study on Traffic Scenario Distribution Alignment

In this section, we present qualitative results on all six geographically diverse regions considered in our study, complementing the subset of three regions reported in Section 4.4 due to page limit. For each region, we visualize the embedding-space distributions of traffic scenarios generated by TRAFFICALIGN and the five unaligned LLM baselines, alongside real-world traffic scenarios.




(a) Los Angeles. (b) Yellowstone. (c) Small Towns in Pennsylvania, USA.

(d) New York City. (e) Yosemite. (f) Switzerland.

Figure 4. UMAP visualization of embeddings of traffic scenarios generated by TRAFFICALIGN, five unaligned LLM baselines, and real-world traffic scenarios.

Across all six regions, we observe the same qualitative trend as in the Section 4.4 Figure 3): TRAFFICALIGN-generated scenarios consistently co-locate with and overlap the real-world cluster, indicating close alignment with real-world traffic distributions. In contrast, GPT-4o, Claude-Sonnet-4, DeepSeek-V3, Qwen3, and GPT-5 form clusters that are clearly displaced from the real-world region, with limited or no overlap. This consistent pattern across all datasets further supports that TRAFFICALIGN captures the dominant modes of the real-world traffic distribution and mitigates distribution shift relative to unaligned LLM baselines.

## B. Ablation Study on Time Window Selection

TRAFFICALIGN uniformly samples frames from videos every 15 seconds (FPS=1/15), since the videos we collected from YouTube mainly depict everyday driving scenarios and rarely include short-horizon edge cases, making 15 seconds sufficient. However, in more intense driving videos, shorter time windows may better capture short-horizon dynamics. To examine this, we curate 10 intense driving videos from YouTube and then compare 5s vs. 15s sampling rate under the same settings as Section 4.2 and 4.3.

Table 4 shows the effectiveness of the scenarios in testing and improving autonomous driving model performances. The results show that a shorter time window can indeed catch more challenging driving scenarios and enhance the effectiveness of scenario-based testing. An interesting future direction is to adaptively detect short-horizon interactions using keyframe sampling methods instead of suing s fixed sampling interval.

Table 4. **Scenario effectiveness across time windows.** This table presents the average test results on three distinct autonomous driving models using scenarios generated from ten intense videos by TRAFFICALIGN using frame sampling rates of 5 seconds and 15 seconds. CR: collision rate, RR: frequency of running red lights, SS: frequency of running stop signs, OR: average distance driven out of road, RF: route following stability, Comp: average percentage of route completion, TS: average time spent to complete the route, ACC: average acceleration, YV: average yaw velocity, LI: frequency of lane invasion, OS: overall score. ↑/↓: higher/lower values indicate more effective in testing the driving models. **Values in bold** indicate the best performance;

(a) Testing driving models.

| **Method** | **Safety Level** | | | | **Functionality Level** | | | **Etiquette Level** | | | **OS** ↓ |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | CR ↑ | RR ↑ | SS ↑ | OR ↑ | RF ↓ | Comp ↓ | TS ↑ | ACC ↑ | YV ↑ | LI ↑ | |
| TRAFFICALIGN (5s) | **0.907** | **0.469** | 0.183 | **0.207** | **0.741** | 0.582 | **0.477** | 0.756 | **0.660** | 0.218 | **0.431** |
| TRAFFICALIGN (15s) | 0.899 | 0.456 | **0.221** | 0.204 | 0.785 | **0.438** | 0.298 | **0.809** | 0.476 | **0.355** | 0.441 |

(b) Fine-tuning driving models.

| **Method** | **Safety Level** | | | | **Functionality Level** | | | **Etiquette Level** | | | **OS** ↑ |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | CR ↓ | RR ↓ | SS ↓ | OR ↓ | RF ↑ | Comp ↑ | TS ↓ | ACC ↓ | YV ↓ | LI ↓ | |
| TRAFFICALIGN (5s) | **0.766** | **0.471** | 0.007 | 0.005 | **0.826** | **0.528** | 0.311 | 0.631 | 0.533 | **0.513** | **0.396** |
| TRAFFICALIGN (15s) | 0.768 | 0.476 | **0.000** | **0.003** | 0.813 | 0.496 | **0.298** | **0.629** | **0.519** | 0.524 | 0.391 |

## C. Additional Details on Data Synthesis

### C.1. Prompt for Traffic Scenario Synthesis

The prompt used to synthesize traffic scenarios from real-world driving videos with GPT 4.1 nano (Section 3.1) is shown in the following box.

**Prompt for Traffic Scenario Synthesis from Real-World Driving Video Frames**

**SYSTEM**

You are a traffic domain expert. Analyze the given image and extract structured information about the traffic scenario.

**TASK**

Analyze the given image and extract information about the traffic scenario following the steps below. Approach this task step-by-step, take your time, and don't skip steps.

Extract the basic information of the traffic scenario:

Step 1. `weather`: one of [*clear, cloudy, foggy, rainy, snowy*], or describe in your own words.

Step 2. `time`: one of [*daytime, nighttime*], or describe in your own words.

Step 3. `road_type`: one of [*straight road, intersection, t-intersection, roundabout*].

Step 4. `one-way_or_two-way`: Determine carefully from visual evidence. A road is one-way if: (a) all visible moving vehicles travel in the same direction, (b) lane markings show no center divider separating opposing flows, (c) one-way signs are visible, or (d) there is no opposing traffic lane. A road is two-way *only* if opposing traffic lanes or oncoming vehicles are clearly visible. Do NOT default to two-way; state one-way whenever opposing traffic is absent. You MUST end this step with an explicit conclusion: state exactly *"This is a one-way road."* or *"This is a two-way road."* with no other phrasing.

Step 5. `special_lane`: note any special lanes (e.g., bike lane, tram track, bus lane), or state `none`.

Step 6. `lane_number`: Count only the lanes visible in the ego vehicle's direction of travel. Use a precise integer. Estimate if unknown.

Step 7. `traffic_sign`: one of [*stop sign, speed limit sign, yield sign, yield to pedestrian sign, school zone*], or state `none`.

Step 8. `traffic_light`: one of [*green, red, yellow, broken*], or state `none`.

Step 9. `road_context`: one of [*urban, residential, highway, country road, dirt road, hill road*], or describe in your own words.

Step 10. Observe additional and detailed information of the traffic scenario:
- Whether the traffic is heavy or light. Is there a traffic jam?
- Whether there are any traffic accidents or other road hazards.
- Whether there are any road works or lane closures.
- Whether there are any emergency vehicles.
- Describe the roadside environment, including buildings, trees, and other objects.

Step 11. Observe any other information that you think is important.

Step 12. `actors`: The input image is captured by a dashboard (forward-facing) camera mounted on the ego vehicle. This means ONLY actors in FRONT of or BESIDE the ego vehicle are visible. Do NOT describe any actor as being behind, left behind, or right behind the ego vehicle.

Before listing actors, scan the full image systematically: near field (close to the ego vehicle), mid field, and far field; left side, center, and right side. Include ALL actors that are present in the image, even if they are partially visible or at the edge of the frame. Do NOT omit any actor that can be seen. At the same time, do NOT invent or assume actors that have no visual evidence in the image.

For each detected actor (e.g., sedan, bus, pedestrian, bicycle, etc.), record the following attributes. Remember to also describe the ego vehicle.
- `type`: The classification of the actor (e.g., sedan, SUV, truck, pedestrian, bicycle, etc.).
- `current_behavior`: The actor's present action or intent (e.g., move forward, turn left, stopped,

yield).

- `speed`: The current speed of the actor as an integer in km/h. You MUST provide a numeric estimate — never use vague descriptions such as "slow", "fast", "moderate", or "unknown". If the exact speed cannot be determined, estimate it based on context:
  - stopped vehicle ⇒ `0 km/h`
  - city vehicle moving normally ⇒ `30--50 km/h`
  - highway vehicle ⇒ `80--120 km/h`
- `position_target` and `position_relation`: Relative position to the ego vehicle. The valid values and their precise definitions are:
  - `front`: the actor is in the *same lane* as the ego vehicle and is ahead.
  - `left front`: the actor is in a lane to the *left* of the ego vehicle AND is ahead longitudinally.
  - `right front`: the actor is in a lane to the *right* of the ego vehicle AND is ahead longitudinally.
  - `left`: the actor is in a lane to the *left* at roughly the same longitudinal position (side-by-side).
  - `right`: the actor is in a lane to the *right* at roughly the same longitudinal position (side-by-side).

  The relations `behind`, `left behind`, and `right behind` must NOT be used. Left and right are always from the ego vehicle driver's perspective (same as left/right in the image). To assign the correct relation, first determine the actor's lane relative to the ego vehicle (same / left / right), then determine whether the actor is ahead or side-by-side.
- `lane_index`: An integer for the lane within the road network. The leftmost lane in the ego vehicle's direction is index `1`; the leftmost lane in the opposite direction is `-1`.
  - E.g., "parked in the rightmost lane" with `lane_number: 3` ⇒ `lane_index: 3`.
  - E.g., "moving forward in the leftmost lane" with `lane_number: 2` ⇒ `lane_index: 1`.

Step 13. Generate a concise, technical description of the traffic scenario that:

- Uses precise traffic engineering terminology, active voice, and present tense.
- Uses assertive language (*"is"/"are"* instead of *"appears"/"seems"*).
- Avoids image-related terms (e.g., "shows", "displays", "depicts").
- Uses precise numbers (e.g., "10 sedans" instead of "many").
- Describes road network first (road context, road type, one-way/two-way, special lanes, lane number, traffic sign, traffic light), then environment (weather, time of day), then additional details from Steps 10–11.
- Ends with an **Actors** section (bold title on a new line) listing each actor type with its precise count.

## OUTPUT FORMAT

<ANALYSIS>
[Your chain of thoughts following the step-by-step instructions provided in step 1 to step 12]
</ANALYSIS>

<TEXT>
[Insert paragraph description here]
**Actors**
[Insert paragraph describing the actors here]
</TEXT>

## IN CONTEXT LEARNING EXAMPLE

**Example input image:**

**Example output:**
<ANALYSIS>
Step 1. weather: clear
Step 2. time: daytime
Step 3. road type: straight road
Step 4. one-way or two-way: two-way
Step 5. special lane: None
Step 6. lane number: 6
Step 7. traffic sign: no traffic sign is observed
Step 8. traffic light: The traffic light is current green
Step 9. road context: urban, residential and commercial mixed-use area
Step 10. the traffic is light; there are no traffic accidents; there are no road works or lane closures; there are no special vehicles such as emergency vehicles
Step 11. No special components are worth noticing
Step 12. actors: Several types of actors are observed. The ego vehicle travels forward at about 38 km/h. A bus is in front of the ego vehicle moving at around 35 km/h. Approximately 10 sedans are parked along the street in the ego direction, and 7 sedans are parked on the rightmost lane in the opposite direction. A truck is driving toward the ego vehicle in the central oncoming lane at about 40 km/h.
</ANALYSIS>

<TEXT>
The traffic scenario occurs on an urban, two-way road with a standard lane configuration, three lanes per direction. The road is marked with one solid yellow centerline, indicating no overtaking, and has no speed limit sign present. A traffic light is visible and currently displays green, enabling vehicle passage in the direction of travel. The environment is during daytime with clear weather and bright conditions. The road is bordered by residential and commercial buildings, with trees lining both sides of the street, and sidewalks are visible along the curb. No traffic jams or accidents are evident, and there are no ongoing road works or lane closures. No emergency vehicles are present. The roadside vegetation, including trees and shrubbery, appears well-maintained, and the overall environment is calm. Traffic flow is smooth with no signs of congestion or hazards, and pedestrians are not observed in the scene.
****Actors****
The ego vehicle is traveling forward at about 38 km/h. There is a bus driving in front of the ego vehicle at around 35 km/h. There are approximately 10 sedans parked along the sides of the street in the ego vehicle's direction and 7 parked sedans on the rightmost lane in the opposite direction. There is a truck driving in the opposite direction in the most central lane at about 40 km/h.
</TEXT>

**Input image:**
[Insert the input image here.]

### C.2. Natural Language Description Examples

This section provides two example natural language scenario descriptions (Section 3.1).

**Natural Language Scenario Description Example 1**

The scenario unfolds on a straight, two-lane urban roadway with a single direction of travel and no special-purpose lanes. Conditions are daytime with clear skies, providing good visibility. The surroundings consist of commercial buildings, sidewalks, and typical city infrastructure, including several tall structures. Traffic signals are green, permitting continuous vehicle flow, and there are no signs of congestion, incidents, or construction work. The road surface is unobstructed, contributing to smooth travel conditions. No emergency vehicles or hazardous elements are detected, and lane availability is unrestricted, indicating all lanes remain open for normal use.
****Actors****
The ego vehicle occupies the right lane, proceeding forward at 30 km/h. In the left lane, a bus travels straight at roughly 25 km/h. Two additional vehicles are present: a car ahead of the ego vehicle and an SUV positioned directly ahead of the bus, each moving at about 30 km/h.

**Natural Language Scenario Description Example 2**

The scene depicts a four-lane, straight urban roadway with single-direction traffic during bright daylight with clear weather beneath an overcast sky. Trees and high-rise buildings line both sides of the street. No traffic lights regulate movement. At the approaching intersection, a "Stop" sign controls vehicle behavior. The roadside environment consists of sidewalks, trees, and dense urban structures. Traffic is partially free-flowing with minimal congestion or hazards, and several vehicles are already in motion.
****Actors****
The ego vehicle is positioned in the second lane from the right, traveling forward at approximately 35 km/h. Seven pedestrians are crossing at the marked crosswalk. Two taxis are moving in the outer-right lane at roughly 30 km/h, while around ten stationary vehicles are parked or stopped along the right roadside. An unoccupied bus is positioned along the right-side curb, remaining stationary throughout the scene.

# D. Additional Details on Data Validation

## D.1. Details of the Domain-Specific Language

For efficient data validation (Section 3.2), we adopt the existing domain-specific language (DSL) design proposed by TARGET [1]. Building on the original grammar, we extend the DSL with an `actor_group` element that groups actors located in close proximity and a `lane_index` attribute that specifies lane-level positions. These extensions enable a concise and precise representation of scenes with many participants. As defined in Figure 5, the DSL follows a context-free grammar, comprising three sections: *Environment*, *Road Network*, and *Actors*.

- ***Environment*** encodes time of day and weather conditions.
- ***Road network*** encodes road type, lane configurations, and traffic signal settings.
- ***Actors*** encode the number, type, position, lane occupancy, and behaviors (e.g., go straight, turn left, etc.) of vehicles and pedestrians grouped by positions.

| | |
|---|---|
| ***Scenario*** | ::= *Environment*; *Road_network*; *Actors* |
| ***Environment*** | ::= *weather*; *time* |
| *weather* | ::= `rainy` \| `foggy` \| `snowy` \| `wet` \| `...` |
| *time* | ::= `daytime` \| `nighttime` |
| ***Road_network*** | ::= *road_type*; *traffic_signals*; *lane_number* |
| *road_type* | ::= `intersection` \| `roundabout` \| `...` |
| *traffic_signals* | ::= *traffic_signs*, *traffic_light* |
| *traffic_signs* | ::= $\epsilon$ \| *traffic_sign*; *traffic_signs* |
| *traffic_sign* | ::= `stop_sign` \| `speed_limit_sign` \| `...` |
| *traffic_light* | ::= $\epsilon$ \|`red_light` \| `green_light` |
| *lane_number* | ::= `0` \| `1` \| `2` \| `3` \| `...` |
| ***Actors*** | ::= *ego_vehicle*; *npc_actors* |
| *ego_vehicle* | ::= *behavior*; *position* |
| *npc_actors* | ::= $\epsilon$ \| *actor_group*; *npc_actors* |
| *actor_group* | ::= *actor_number*; *actor_type*; *behavior*; *position* |
| *actor_type* | ::= `car` \| `truck` \| `pedestrian` \| `...` |
| *behavior* | ::= `go_forward` \| `turn_left` \| `...` |
| *position* | ::= *reference_point*; *relative_position*; *lane_index* |
| *reference_point* | ::= `ego_vehicle` \| *road_type* \| *traffic_signals* |
| *relative_position* | ::= `front` \| `behind` \| `left` \| `on` \| `...` |
| *lane_index* | ::= `0` \| `1` \| `2` \| `3` \| `...` |

Figure 5. The grammar of the traffic scenario DSL [1].

### D.2. Prompt for Natural Language to DSL Translation

The prompt used to translate TRAFFICALIGN-synthesized traffic scenario natural language descriptions to the driving videos with GPT 5 (Section 3.1) is shown in the following box.

**Prompt for Translating Natural Language Description into DSL Representation**

**SYSTEM**

You are a traffic domain expert familiar with domain-specific languages (DSL).

**TASK**

Your task is to extract the information about a traffic scenario from a textual description and represent it as a DSL in YAML format. Take your time to solve this task step by step.

**Steps 1–8:** Extract the basic information of the traffic scenario:

Step 1. `weather`: one of [*clear, cloudy, foggy, rainy, snowy*]. Don't infer. If the natural language description does not clearly state this, use `unspecified`.

Step 2. `time`: one of [*daytime, nighttime*]. Don't infer. If the natural language description does not clearly state this, use `unspecified`.

Step 3. `road_type`: one of [*straight road, curved road, intersection, t-intersection, roundabout*]; include one-way/two-way and any special lanes (e.g., bike lane, tram track). Don't infer. If the natural language description does not clearly state this, use `unspecified`.

Step 4. `lane_number`: The number of lanes *per direction* for a two-way road. Use a precise integer. Don't infer. If not clearly stated, use `unspecified`.
- E.g., a two-way road with two lanes in each direction ⇒ `lane_number: 2`.
- E.g., a two-way road with two lanes total ⇒ `lane_number: 1`.

Step 5. `one-way_or_two-way`: one of [*one-way, two-way*]. Don't infer. If not clearly stated, use `unspecified`.

Step 6. `traffic_sign`: one of [*stop sign, speed limit sign, yield sign, yield to pedestrian sign, school zone*], or `none` if no sign is present. Don't infer. If not clearly stated, use `unspecified`.

Step 7. `traffic_light`: one of [*green, red, yellow, broken*], or `none` if no traffic light is present. Don't infer. If not clearly stated, use `unspecified`.

Step 8. `road_context`: one of [*urban, residential, highway, country road, dirt road, hill road*]. Don't infer. If not clearly stated, use `unspecified`.

Step 9. `actors`: For each actor group (e.g., car, bus, pedestrian, bicycle), extract the following attributes:
- `type`: The classification of the actor (e.g., sedan, SUV, truck, pedestrian, cyclist). Don't infer. If not clearly stated, use `unspecified`.
- `number`: The number of actors in this group. If the description states an approximate count using hedging words (e.g., "around 10", "approximately 5", "about 3", "roughly 8"), extract the integer value — do *not* treat hedging language as `unspecified`. Only use `unspecified` when no numeric information is given at all (e.g., "several", "a few", "many").
- `behavior`: The actor's present action or intent (e.g., move forward, turn left, stop, yield). Don't infer. If the natural language description does not clearly state this, use "unspecified". Use "parked" when the actor is described as parked (e.g., "parked along the roadside", "parked by the curb") — do not use "stopped" for parked actors. Use "stopped" only for actors that have temporarily halted while in traffic (e.g., waiting at a red light). Treat "stationary or moving slowly" as `stopped`.
- `speed`: The current speed in km/h. If an approximate speed is given using hedging words (e.g., "around 30 km/h", "approximately 50 km/h", "roughly 60 km/h"), extract the integer value. Use `unspecified` when: (a) no numeric speed information is given, or (b) speed is described only with a qualitative adjective (e.g., "slow", "fast", "moderate", "crawling") with no accompanying number.
- `position_target` and `position_relation`: Relative position to the ego vehicle. If impossible to infer, use `unspecified`.

  - E.g., "directly ahead of the ego vehicle in the same lane" ⇒ `position_target: ego_vehicle; position_relation: front`.
  - E.g., "to the left rear of the ego vehicle in an adjacent lane" ⇒ `position_target: ego_vehicle; position_relation: left behind`.
  - E.g., "in the opposite direction driving towards the ego vehicle" ⇒ `position_target: ego_vehicle; position_relation: front`.
  - E.g., ego vehicle is in the leftmost lane and another actor is parked by the curbside ⇒ `position_target: ego_vehicle; position_relation: right`.
- `lane_index`: An integer for the lane within the road network. The leftmost lane in the ego vehicle's direction is index `1`; the leftmost lane in the opposite direction is `-1`. Use `unspecified` if not determinable.
  - E.g., "parked in the rightmost lane" with `lane_number: 3` ⇒ `lane_index: 3`.
  - E.g., "moving forward in the leftmost lane" with `lane_number: 2` ⇒ `lane_index: 1`.
  - E.g., "parked by the curbside in the opposite direction" with `lane_number: 2` ⇒ `lane_index: -2`.

Step 10. Based on the information extracted in Steps 1–9, generate a structured YAML block. Wrap this with `<YAML>...</YAML>` as shown below.

```
<YAML>
environment:
    weather: clear  # one of: clear, cloudy, foggy, rainy, snowy, unspecified
    time: daytime   # one of: daytime, nighttime, unspecified
road_network:
    road_type: straight road  # one of: straight road, curved road,
                              #   intersection, t-intersection, roundabout,
                              #   unspecified
    road_context: urban  # one of: urban, residential, highway, country road,
                         #   dirt road, hill road, unspecified
    traffic_sign: none   # one of: stop sign, speed limit sign, yield sign,
                         #   yield to pedestrian sign, school zone, none,
                         #   unspecified
    traffic_light: none  # one of: green, red, yellow, broken, none, unspecified
    lane_number: 1       # integer or unspecified
    one-way_or_two-way: two-way  # one of: one-way, two-way, unspecified
actors:
    ego_vehicle:
        type: none           # always none for ego_vehicle
        behavior: move forward  # e.g.: move forward, stopped, turn left,
                                #   turn right, merge left, merge right,
                                #   unspecified
        speed: 30 km/h       # integer in km/h, or unspecified
        position_target: unspecified
        position_relation: unspecified  # one of: front, left, right, behind,
                                        #   left front, left behind, right
                                        ↪  front,
                                        #   right behind, unspecified
        lane_index: 1        # integer or unspecified
    other_actor_1:
        type: sedan          # e.g.: sedan, truck, tram, van, car, SUV, bus,
                             #   pedestrian, bicycle
        number: 1            # integer or unspecified
        behavior: move forward
        speed: 30 km/h       # integer in km/h, or unspecified
        position_target: ego_vehicle  # NOTE: always ego_vehicle for all
                                      #   actors other than ego_vehicle
        position_relation: front
        lane_index: 1
    ... (other actors if any) ...
</YAML>
```

### IN CONTEXT LEARNING EXAMPLE

**Example input 1:**

The traffic scenario takes place on a two-lane, two-way urban street with no special lanes and no visible traffic signs or signals. The environment is during daytime with cloudy weather. The roadway is lined with commercial buildings and sidewalks populated by pedestrians and cyclists. Traffic is moderate, with vehicles moving smoothly and no apparent congestion, accidents, or hazards. No roadworks or lane closures are present. No emergency vehicles are observed.

**Actors**

The ego vehicle travels at about 28 km/h. Ahead of it, a yellow sedan moves forward at approximately 30 km/h, followed by a black SUV traveling at around 32 km/h. In the opposite lane, around four cars drive toward the ego at roughly 35 km/h.

**Example output 1:**

```
<YAML>
environment:
    time: daytime
    weather: cloudy
road_network:
    lane_number: 1
    one-way_or_two-way: two-way
    road_type: straight road
    traffic_light: none
    traffic_sign: none
    road_context: urban
actors:
    ego_vehicle:
        type: none
        behavior: move forward
        lane_index: 1
        position_relation: unspecified
        position_target: unspecified
        speed: 28 km/h
    vehicle_in_front_1:
        type: sedan
        number: 1
        behavior: move forward
        lane_index: 1
        position_relation: front
        position_target: ego_vehicle
        speed: 30 km/h
    vehicle_in_front_2:
        type: SUV
        number: 1
        behavior: move forward
        lane_index: 1
        position_relation: front
        position_target: ego_vehicle
        speed: 32 km/h
    vehicles_opposite_direction:
        type: car
        number: 4
        behavior: move forward
        lane_index: -1
        position_relation: front
        position_target: ego_vehicle
        speed: 35 km/h
</YAML>
```

**Example input 2:**
The traffic scenario takes place on a rural, two-way curved road within mountainous terrain. The road has one lane in each direction, two lanes in total. No traffic signs or lights are visible in the scene. The environment features mostly clear daytime weather, providing good visibility of the surrounding rugged landscape. Road work is in progress, as shown by a worker standing ahead in the ego vehicle's direction. Traffic is light, with no congestion, accidents, or emergency vehicles present. The roadside is characterized by rocky slopes and sparse trees, with wide open areas extending along both sides of the roadway.
**Actors**

The ego vehicle is moving forward at about 45 km/h. In the opposite lane, two construction trucks are traveling extremely slowly at around 3 km/h. Ahead of the ego vehicle, a road worker is standing still in the lane, remaining stationary at 0 km/h.

**Example output 2:**

```
<YAML>
environment:
    time: daytime
    weather: clear
road_network:
    lane_number: 1
    one-way_or_two-way: two-way
    road_type: curved road
    traffic_light: none
    traffic_sign: none
    road_context: country road
actors:
    ego_vehicle:
        type: none
        behavior: move forward
        lane_index: 1
        position_relation: unspecified
        position_target: unspecified
        speed: 45 km/h
    construction_vehicles:
        type: construction vehicle
        number: 2
        behavior: move forward
        lane_index: -1
        position_relation: front
        position_target: ego_vehicle
        speed: 3 km/h
    traffic_worker:
        type: construction worker
        number: 1
        behavior: stopped
        lane_index: 1
        position_relation: front
        position_target: ego_vehicle
        speed: 0 km/h
</YAML>
```

**Input natural language description:**

[Insert the input natural language description here.]

## D.3. Effectiveness of DSL Validation

We conduct a preliminary comparison to compare driving model testing and fine-tuning performance on scenarios synthesized with and without the DSL component.

As shown in Table 5, incorporating DSL validation is associated with improves scenario quality across both evaluation settings. For autonomous driving model testing experiments, scenarios generated with DSL validation achieve a lower overall score (0.319 vs. 0.408) and a higher collision rate (0.925 vs. 0.873). A similar trend is also observed when using the generated scenarios in fine-tuning autonomous models, where DSL validation improves the overall score from 0.369 to 0.402 and lowers the collision rate from 0.745 to 0.691.

Table 5. **Driving model testing & fine-tuning experiments.** This table presents the average test results on three distinct autonomous driving models using scenarios generated with and without the DSL component. CR: collision rate, RR: frequency of running red lights, SS: frequency of running stop signs, OR: average distance driven out of road, RF: route following stability, Comp: average percentage of route completion, TS: average time spent to complete the route, ACC: average acceleration, YV: average yaw velocity, LI: frequency of lane invasion, OS: overall score. ↑/↓: higher/lower values indicate more effective in testing the driving models. **Values in bold** indicate the best performance;

(a) Testing driving models.

| **Method** | **Safety Level** | | | | **Functionality Level** | | | **Etiquette Level** | | | **OS** ↓ |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | CR ↑ | RR ↑ | SS ↑ | OR ↑ | RF ↓ | Comp ↓ | TS ↑ | ACC ↑ | YV ↑ | LI ↑ | |
| w/o DSL Validation | 0.873 | 0.286 | 0.219 | 0.038 | 0.718 | **0.496** | 0.381 | 0.659 | 0.488 | 0.219 | 0.408 |
| with DSL Validation | **0.925** | **0.461** | **0.273** | **0.041** | **0.699** | 0.527 | **0.391** | **0.793** | **0.541** | **0.357** | **0.319** |

(b) Fine-tuning driving models.

| **Method** | **Safety Level** | | | | **Functionality Level** | | | **Etiquette Level** | | | **OS** ↑ |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | CR ↓ | RR ↓ | SS ↓ | OR ↓ | RF ↑ | Comp ↑ | TS ↓ | ACC ↓ | YV ↓ | LI ↓ | |
| w/o DSL Validation | 0.745 | **0.318** | 0.010 | 0.000 | 0.749 | 0.533 | 0.312 | 0.574 | 0.501 | 0.418 | 0.369 |
| with DSL Validation | **0.691** | 0.366 | **0.000** | 0.000 | **0.814** | **0.593** | **0.211** | **0.553** | **0.496** | **0.381** | **0.402** |

## D.4. Preliminary Evaluation of DSL Validation Accuracy

We conduct a preliminary evaluation to assess the accuracy of DSL validation and to examine the extent to which its outputs are affected by LLM hallucinations. Specifically, we sample 120 natural language scenario descriptions, manually annotate the ground-truth DSL, and compare the DSL outputs generated by the DSL converter against these annotations. We find that 13 out of 120 files contain hallucinations in the generated DSL, indicating that the DSL converter is affected by LLM hallucinations. Nevertheless, the error rate is relatively limited, and the hallucinations are concentrated in a small number of fields. The most frequent hallucination is `lane_number`, which appears in 9 files, followed by `one-way or two-way`, which appears in 4 files. We also observe one file each with a hallucination in `lane_index` and the actor `type`.

Overall, these results indicate that hallucinations are present but not pervasive in DSL validation on this preliminary benchmark.

# E. Additional Details on LLM Alignment

## E.1. Implementation Details for LLM Alignment

We implement the fine-tuning pipeline using the Unsloth framework to leverage 4-bit quantization, enabling memory-efficient training of the Llama-3.2-3B-Instruct model on a single NVIDIA Tesla T4 GPU. The model is trained with a maximum sequence length of $2,048$ tokens. For Parameter-Efficient Fine-Tuning (PEFT), we configure Low-Rank Adaptation (LoRA) with a rank $r = 16$ and a scaling factor $\alpha = 16$. We apply LoRA adapters to all linear projection layers, including the query (`q_proj`), key (`k_proj`), value (`v_proj`), output (`o_proj`), and the MLP layers (`gate_proj`, `up_proj`, `down_proj`), with no dropout. Optimization is performed using the 8-bit AdamW optimizer with a weight decay of $0.01$. We utilize a linear learning rate scheduler with $5$ warmup steps, targeting a peak learning rate of $2e-4$. The training process uses a per-device batch size of $2$ with $4$ gradient accumulation steps. To ensure the model learns strictly from the desired output distributions, we employ a response-only masking strategy, where loss is calculated exclusively on the assistant's generated tokens, masking the system instructions and user prompts.

## E.2. Overview of the Six LLM Alignment Datasets

Table 6 summarizes the composition of the six LLM alignment datasets used in our study across diverse geographic locations.

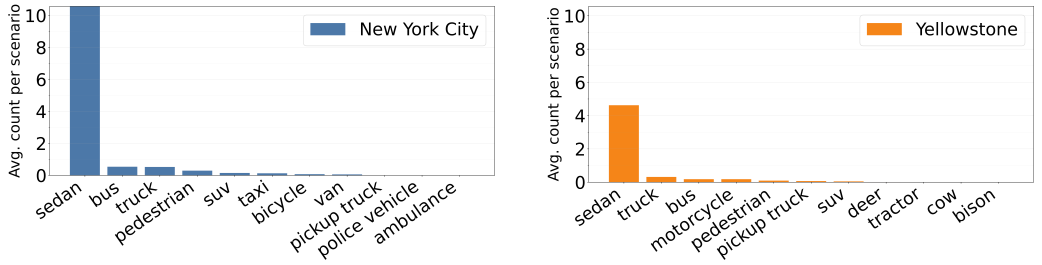
Table 6. Summary statistics of the six LLM alignment datasets across diverse geographical locations.

| **Location** | **# Videos** | **Avg. # Actors** |
|---|---|---|
| Los Angeles | 82 | 13.8 |
| New York City | 84 | 13.3 |
| Yellowstone | 14 | 2.5 |
| Yosemite | 20 | 2.3 |
| Small Towns in PA | 15 | 3.2 |
| Switzerland | 46 | 11.1 |

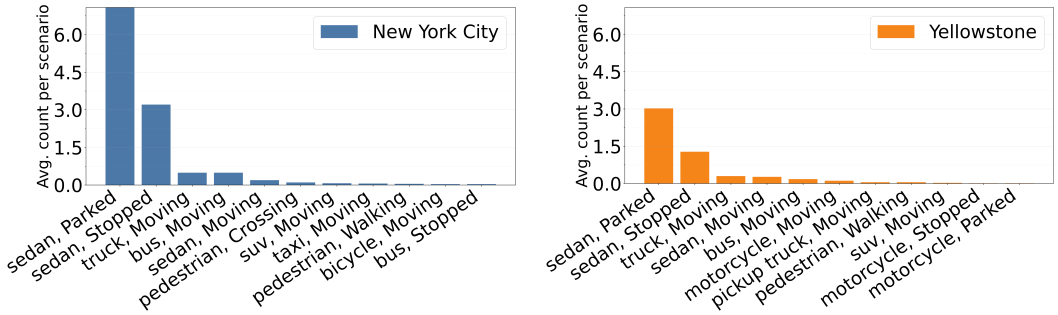
Across the six alignment datasets, the average number of actors clearly tracks the urban–rural divide. The three urban datasets, Los Angeles, New York City, and Switzerland, consistently exhibit dense interactions, with 13.8, 13.3, and 11.1 actors per scenario, respectively. In contrast, the rural and small-town datasets (Yellowstone, Yosemite, and Small Towns in PA) are much sparser, with only 2.3–3.2 actors on average.

## E.3. Behavior Difference Across Geographic Regions

We analyze generated scenarios from a metropolitan (New York City) and a rural (Yellowstone) dataset. As shown in Figure 6, New York City is more crowded, with frequent pedestrian crossings and more parked sedans, while Yellowstone is sparser and features rural-only actors (e.g., deer, cows, bison, tractors) absent in New York City.




(a) Actor type distributions.




(b) Actor type-behavior distributions.

Figure 6. Behavioral differences across geographic regions.

## E.4. Human Study on Scenario Semantic Fidelity

To evaluate whether TRAFFICALIGN-generated scenarios are semantically consistent with real-world traffic scenarios, we conduct a qualitative study in Section 4.4 and Section A. In this section, we further conduct a human study to assess the semantic fidelity of the generated scenarios. We sample 100 scenarios (50 New York City, 50 Yellowstone) and ask 6 graduate students to independently rate the semantic fidelity of each scenario on a 1–5 scale, measuring "how likely the traffic scenario is to occur in the specified location". The human study reports mean scores of 4.553 (New York City) and 4.627 (Yellowstone). The results show that TRAFFICALIGN generates semantically faithful, region-consistent traffic scenarios.

## E.5. TRAFFICALIGN Prompt for Real-World Aligned Traffic Scenario Generation

This section presents the complete prompt used in our experiments to query the aligned LLMs introduced in Section 3.3.

**Prompt for Aligned LLMs to Generate Traffic Scenarios**

**SYSTEM**

You are a traffic domain expert.

**TASK**

Design a traffic scenario in real life and provide a detailed description including the environment (weather, time of day), road network (road type, road context, lane number, one-way or two-way). Then design the position, behavior and speed of the ego vehicle and details of other actors, specifying actor type, behavior, speed, position, lane index, and any interactions with the ego vehicle. Wrap the generated textual description in <TEXT>...</TEXT> tags.

## F. Unaligned LLM Baseline Prompts

The prompt used to generate traffic scenarios using unaligned LLM baselines in the ablation study in Section 4.1 is shown in the following box.

### Prompt for Unaligned LLM baselines to Generate Traffic Scenarios

**SYSTEM**

You are a traffic domain expert. Your task is to design a traffic scenario for testing autonomous driving systems.

**TASK**

Design a traffic scenario take place in **[Insert the region name here]**. Approach this task step-by-step, take your time, and don't skip steps.

Design the basic configurations of the traffic scenario:

Step 1. weather: one of [clear, cloudy, foggy, rainy, snowy], or describe in your own words

Step 2. time: one of [daytime, nighttime], or describe in your own words

Step 3. road type: one of [straight road, intersection, t-intersection, roundabout]

Step 4. one-way or two-way: one of [one-way, two-way]

Step 5. special lane: if there are any special lanes (e.g., bike lane, tram track, bus lane)

Step 6. lane number: use an integer.

Step 7. traffic sign: one of [stop sign, speed limit sign, yield sign, yield to pedestrian sign, school zone], or state "none"

Step 8. traffic light: one of [green, red, yellow, broken], or state "none"

Step 9. road context: one of [urban, residential, highway, country road, dirt road, hill road], or describe in your own words

Step 10. Design additional and detailed information of the traffic scenario:

- Whether the traffic is heavy or light; Is there a traffic jam?
- Whether there are any traffic accidents or other road hazards
- Whether there are any road works or lane closures
- Whether there are any emergency vehicles
- Describe the environment of the roadside, including the buildings, trees, and other objects.

Step 11. Consider any other information that you think is important.

Step 12. actors: For each detected actor (e.g., car, bus, pedestrian, bicycle), record the following attributes:

- type: The classification of the actor (e.g., sedan, SUV, truck, pedestrian, cyclist)
- current behavior: The actor's present action or intent (e.g., move forward, turn left, stop, yield)
- speed: The current speed of the actor, measured in kilometers per hour (km/h)
- position: The actor's location, which should be described in two ways: (1) relative position to the ego vehicle using position target and position relation, and (2) the absolute position in the road network using lane index
- position target and position relation: Relative position to the ego vehicle. Specify the actor's position concerning the ego vehicle.
  - For example, if another actor is "directly ahead of the ego vehicle in the same lane", position target is ego vehicle; position relation is front.
  - Another example, if another actor is "to the left rear of the ego vehicle in an adjacent lane", position target is ego vehicle; position relation is left behind.
- lane index: an integer stating the lane index within the road network. The lane index of the leftmost lane in the direction of the ego vehicle is 1. The lane index of the leftmost lane in the opposite direction of the ego vehicle is -1.
  - For example, if an actor is "parked in the rightmost lane", and the lane number is 3, then the lane index is 3.
  - Another example, if an actor is "moving forward in the leftmost lane" and the lane number is 2, then the lane index is 1.

Generate a concise, technical description of the traffic scenario that:
- Use precise traffic engineering terminology.
- Use languages to describe a traffic scenario rather than a image.
- Use active voice and present tense.
- Avoid subjective observations.
- Use assertive language (“is” / “are” instead of “appears” / “seems” / “possibly”).
- Avoid image-related terms (e.g., “shows”, “displays”, “depicts”).
- Use precise numbers (e.g., “10 cars” instead of “many cars” / “multiple cars”).
- Maintain all key information about the road network, the environment, and actors.
- Describe the road network first, then the environment, then the actors.
- For the road network, first describe the road context and the road type, one-way / two-way, and any special lanes (e.g., bike lane, tram track). Then describe the lane number, the traffic sign, and the traffic light.
- For the environment, describe the weather and time of day.
- Describe the additional and detailed information observed in step 10 and step 11.
- At the end, add the descriptions of the actors. For the actors, describe the type and count of each type of actor. Remember to include the precise number (in digits) of actors.
- Treat the actor section differently and start this section with a separate title **Actors** in a new line.

Start your generated description with <TEXT> and end with </TEXT>

**OUTPUT FORMAT**

<ANALYSIS>
[Your chain of thoughts following the step-by-step instructions provided in step 1 to step 12]
</ANALYSIS>
<TEXT>
[Insert paragraph description here]
**Actors**
[Insert paragraph describing the actors here]
</TEXT>

## References


[1] Yao Deng, Zhi Tu, Jiaohong Yao, Mengshi Zhang, Tianyi Zhang, and James Xi Zheng. TARGET: traffic rule-based test generation for autonomous driving via validated llm-guided knowledge extraction. *IEEE Trans. Software Eng.*, 51(7):1950–1968, 2025. 7