Language-Guided Traffic Simulation and Rare Event Synthesis for an Urban Intersection

Chengbo Zang[1,*] Wangshu Zhu[1,*] Mehmet Kerem Turkcan[1] Siyi Liu[1] Hanlin Wang[1]
Xudong Chen[1] Boshra Khalili[1] Noshin Saiyara Ahmad[2] Peter Jing Jin[2]
Javad Ghaderi[1] Gil Zussman[1] Zoran Kostic[1]

[1]Columbia University [2]Rutgers University [*]Equal Contributions

Abstract

We propose a data synthesis pipeline for generating realistic traffic scenes and safety-critical rare events under natural language instructions while providing agent relation annotations. The pipeline structurally comprises the scene planner, agent generator, waypoint filter, event reasoner, and trajectory refiner, while incorporating a language model backend for controlled inference. By decomposing high-level semantic reasoning and low-level scene execution, our framework is able to produce physically grounded agent trajectories that satisfy the social relation specifications. The pipeline is used to generate a dataset of 370 traffic scenes based on an urban traffic intersection, featuring agent relations such as collision and yielding, which are safety-critical but challenging to specify in real world traffic data. We evaluate the quality of the synthesized agent trajectories by the simulation-to-reality gaps, where the pipeline achieves an 84% of instruction satisfaction rate equipped with the Claude3.5-Sonnet backend. We further showcase the usage of the synthesized dataset by testing traffic scene perception and precognition using a simple agentic pipeline, both outperforming non-LLM baselines by a noticeable margin.

1. Introduction

Accurate modeling of traffic flows within simulation environments is critical for addressing the complex transportation challenges in modern cities. Due to the safety-critical nature of urban traffic, simulation-based approaches are becoming increasingly important to avoid conducting experiments directly on real world streetscapes [10]. Yet realistic traffic simulation in urban environments is particularly challenging considering multi-agent (*i.e.* traffic participant such as pedestrian or vehicle) interaction, varying physical and social constraints, and safety-critical rare events. Traditional rule-based simulators such as SUMO [22] often suffer from the burden of meticulous definition of complex behavioral heuristics.

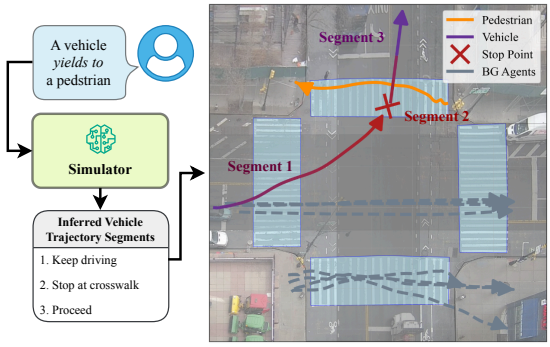


Figure 1. Example of a synthesized yielding scene. The LLM-driven simulator infers the behavioral segments of the foreground pedestrian and vehicle, and then refine them using a DL trajectory prediction model. Background (BG) agents are injected to amplify overall scene complexity.

To overcome the rigidity of rule-based models, an emerging trend is to introduce a large language model (LLM) as an *interactive reasoning agent* which interprets high-level natural language instructions and builds executable traffic simulation scenarios [14]. However, simulators that rely purely on LLMs for end-to-end motion planning have limited ability to interact with the physical world. While LLMs excel at high-level semantic logic and relational inference, they struggle with exact spatial reasoning, geometric grounding, and physical kinematics. Moreover, safety-critical edge cases such as collisions are hardly found in standard datasets, and ground-truth annotation of such events, including social relations between the participating agents, is extremely time-consuming.

In this paper, we propose a novel traffic scene-synthesis pipeline for generating realistic agent trajectories for an urban intersection under natural language instructions. The

pipeline provides exact social relation annotation for safety-critical cases such as collision and yielding. We adopt a five-stage architecture including the scene planner, agent generator, waypoint filter, event reasoner, and trajectory refiner, decoupling (i) the semantic reasoning using an LLM from (ii) physical trajectory generation using learned statistical priors and finetuned deep learning (DL) trajectory prediction models. As illustrated in Fig. 1, the reasoning module decomposes agent motion into plausible behavioral segments, and a refinement module generates the actual trajectories based on the inferred segments.

The main contributions of this paper can be summarized as follows:

- We propose a novel multi-stage traffic synthesis pipeline in Sec. 3, effectively bridging the gap between semantic reasoning and physical execution of language-guided scene generation.
- We provide a synthetic data set in Sec. 4.1 with automatic annotations of ground-truth agent-relation, including rare but safety-critical events such as collision and yielding.
- We conduct quantitative evaluations of the dataset and pipeline quality in Secs. 4.2 and 4.3, and demonstrate the dataset usage on agentic traffic perception and precognition in Sec. 5.

2. Related Work

Traffic Simulation. Traffic simulation is a long-standing problem central to the development and evaluation of autonomous driving systems. Rule-based simulators such as SUMO [22] encode traffic regulations and car-following models to produce large-scale flows, yet struggle to capture the diversity and irregularity of real driving behavior. To address this, data-driven approaches such as TrafficSim and InterSim [30, 31] generate socially consistent multi-agent trajectories using implicit latent models and explicit relational reasoning, respectively. To further enhance controllability, recent methods allow users to steer agent behaviors during rollout via multimodal prompts (ProSim [33]), or condition generation on latent personality variables and quantified social dispositions (TrafficBots [40], Editing Driver Character [5]). On the intersection-specific front, Data-Driven Traffic Simulation (DDTS) [39] augments real world trajectories to evaluate traffic dynamics in a controlled setting. AGENTS-LLM [38] employs an agentic framework to augment real world driving scenarios via natural language instructions, demonstrating that an agentic design maintains high output quality even with smaller language models.

Trajectory Prediction. Trajectory prediction aims to forecast the future positions of traffic agents given their observed histories and scene context. While the agent dynamics are usually handled by sequence prediction models such as recurrent neural networks [13] or Transformers [35], researchers extensively study different ways to encode agent interactions. Social-LSTM [2] and Social-GAN [12] introduce a social pooling module that aggregates the hidden states of the pedestrians in the scene, enabling the model to *implicitly* account for their relations. Trajectron++ [29] proposes to encode the scene using a graph-based representation which integrates agent dynamics and semantic maps. Context-Aware Timewise VAEs [36] adopts a dual attention mechanism that accounts for both the social relations and the environmental constraints. These methods are trained on standard datasets such as ETH/UCY [18, 26], nuScenes [4], and the Waymo dataset [8], and are evaluated primarily by displacement error metrics. While trajectory prediction models excel at learning the dynamics of real world agent motion, they do not address the generation of full traffic scenes from high-level semantic specifications.

LLM in Traffic Synthesis and Decision Making. Traffic applications integrate LLMs mainly along two axes: scene generation and decision-making. On the generation side, Chat2Scenario [41] uses an LLM to search for and extract driving scenarios from existing datasets, and then converts the samples that matches the criterion into the required simulation formats. LeGEND [34] proposes a top-down workflow that uses LLMs to translate scenarios described by natural language into intermediate logic, followed by genetic programming techniques to diversify the generations. TrafficGen [9] proposes a data-driven framework for generating realistic traffic scenarios, making it highly relevant to traffic-scene generation. LCTGen [32] further extends this by incorporating language conditioning, enabling generation from high-level textual descriptions. Generating Traffic Scenarios via In-Context Learning [1] translates textual scene descriptions into simulator scripts through in-context learning, and integrates a CARLA simulator [6] for realistic scene rendering. On the decision-making side, literature focuses predominantly on autonomous driving. Agent-Driver [23] introduces an LLM-based cognitive agent that leverages a tool library and commonsense memory to perform chain-of-thought reasoning, task planning, and self-reflection for driving trajectory generation. LLMLight [17] formulates adaptive traffic signal control as a language reasoning problem, distilling GPT-4 decisions into a compact domain-specific model that achieves competitive throughput and interpretable phase selection.

Unlike traditional traffic simulators that focus on low-level execution, or LLM-based cognitive agents designed solely for high-level decision-making, our work diverges from these paradigms by unifying the semantic reasoning capabilities of LLMs with the realistic physical dynamics to synthesize explicitly annotated datasets. We utilize a decoupled, LLM in-the-loop pipeline to construct fully verifiable

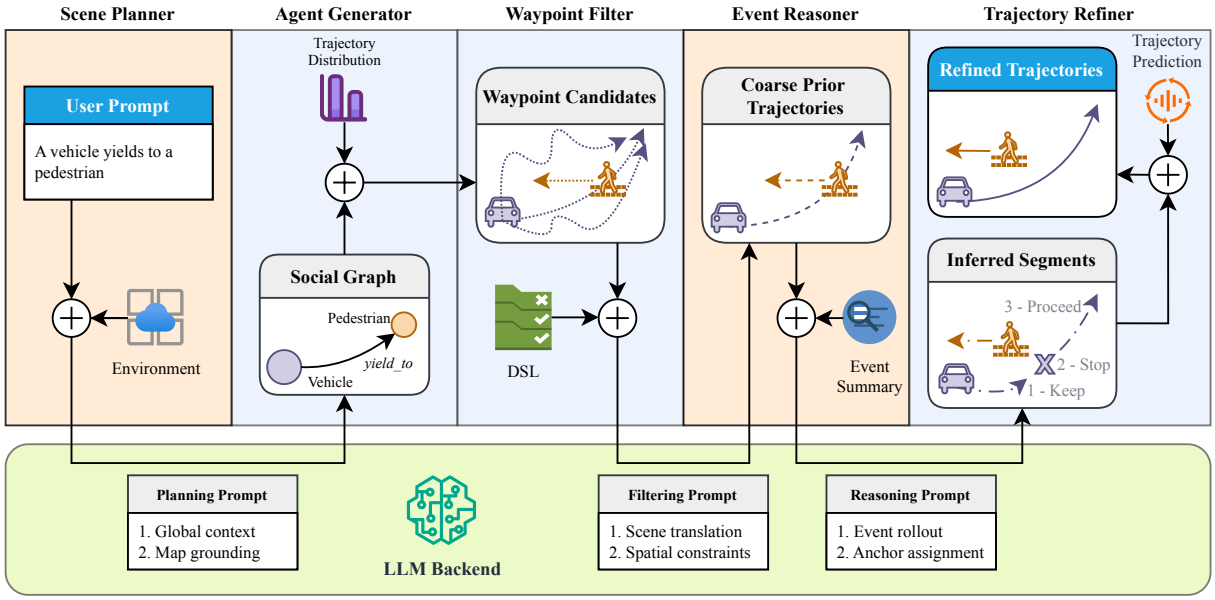


Figure 2. Workflow of language-driven traffic synthesis. Each box represents a stage in the pipeline, where the orange ones focus on high-level semantic reasoning, and the blue ones focus on low-level scene execution.

agent trajectories and relations, with deliberate synthesis of safety-critical rare events including collision and yielding, providing a reliable benchmark to evaluate the quality of intelligent transportation systems.

3. Method

The data synthesis pipeline is composed of five cascading stages illustrated in Fig. 2: the scene planner, agent generator, waypoint filter, behavior reasoner, and trajectory refiner. After the user inputs a natural language instruction (*e.g.* "A vehicle yields to a pedestrian"), the planner, filter, and reasoner query an LLM backend at different abstraction levels, handling semantic parsing, constraint specification, and behavioral reasoning, respectively. Meanwhile, the generator samples the agent waypoints from the spatiotemporal trajectory distributions fitted on real world data, which is later modified according to the behavioral segments inferred by the reasoner, and eventually executed by the refiner to generate realistic and physically-grounded agent trajectories.

3.1. Scene Planner

We begin by converting the free-form user instruction into a structured relation-centric *social graph* [19, 21, 25] using a large language model. This step extracts the semantic entities and relations implied by the instruction, while deferring geometric realization to later stages. The social graph isolates high-level scene semantics from low-level trajectory generation, enabling subsequent modules to operate under explicit relational constraints.

Social Graph Representation. The social graph is a *directed graph* defined as $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{C})$, where $\mathcal{V}$ is the set of agents as the nodes, $\mathcal{E}$ is the set of pairwise relationship between the agents as the edges, and $\mathcal{C}$ denotes the global scene context. Each agent node $v_i \in \mathcal{V}$ encodes meta information such as the class of the agent (*e.g.* pedestrian or vehicle) and the spawn time, together with high-level intention attributes including preferred traveling direction, and goal region. Each edge $e_{ij} \in \mathcal{E}$ specifies an intended relationship that the scene is expected to realize, such as *yield_to*, *collide_at*, or *follow*. The global context $\mathcal{C}$ captures scene-level conditions including traffic density and time-of-day.

Hierarchical Scene Description. A key challenge in language-conditioned scene generation is that many prompts differ lexically while describing the same underlying traffic event. We address this by organizing scene semantics hierarchically according to traffic theme, interaction subtype, and conflict pattern. This hierarchy captures both broad behavioral families such as pedestrian-vehicle interactions and multi-vehicle priority conflicts, as well as specific interaction templates with well-defined agent roles and contextual conditions.

Map Grounding. In order to bind the social graph $\mathcal{G}$ to the physical reality, we ground it to the map by associat-

ing semantic directions and relationships to concrete entities such as lanes, crosswalks, stop lines, and goal regions. The purpose is to reduce the ambiguity of free-form language and provide the geometric anchors required by the filter and the reasoner.

3.2. Agent Generator

Given the structured scene specification from the planner, we next generate agents and their candidate trajectories using data-driven models. Due to the strong spatial and temporal pattern exhibited by urban traffic intersections, it is reasonable to fit statistical models of the agent trajectories and utilize them to generate new agents. Specifically, let $p_{\text{tod}}(\cdot|t)$ denote the distribution of the total number of agents in the intersection at time-of-day t, and $p_{\text{wp}}(\cdot|c, x_{\text{s}}, x_{\text{e}})$ be the distribution of the waypoints along an agent trajectory of class c which starts at x_{s} and ends at x_{e}. Following DDTS [39], we use the Gaussian Mixture Model (GMM) as the conditional generative distribution underlying both p_{tod} and p_{wp}.

Scene Amplification. The social graph $\mathcal{G}$ generated by the planner includes only the agents that are involved in the relation specified by the user prompt and is often limited in size. In order that the generation resemble real world traffic, we amplify the scene by injecting a random number of *background agents* simultaneously traveling through the intersection. Given the time-of-day t, we sample $N \sim p_{\text{tod}}(\cdot|t)$ as the total number of agents that should be present in the scene according to the temporal distribution. If the number of agents $|\mathcal{V}|$ is smaller than N, we artificially inject new background agents so that

$$\tilde{\mathcal{V}} := \mathcal{V} \cup \left\{ v^{|\mathcal{V}|+1}, v^{|\mathcal{V}|+2}, \ldots, v^{N} \right\}, \quad |\tilde{\mathcal{V}}| = N. \tag{1}$$

To maintain the correctness and clarity of the annotations of social relationships (*i.e.* the relations in $\mathcal{E}$), we specifically require that the background agents in $\tilde{\mathcal{V}} \setminus \mathcal{V}$ do *not* interact with the background agents in the original $\mathcal{V}$.

Waypoint Sampling. Once the entries and exits of agents are specified, waypoint candidates can be directly sampled from the learned trajectory distribution p_{wp}. For example, if v^i is a vehicle entering the intersection from west traveling east, we construct a set of M waypoint candidates of v^i as $\mathcal{W}^i := \{w^{i,m}, m = 1, \ldots, M\}$, where

$$w^{i,m} \sim p_{\text{wp}}(\cdot|c = \text{vehicle}, x_{\text{s}} = \text{west}, x_{\text{e}} = \text{east}). \tag{2}$$

We iterate over $\tilde{\mathcal{V}}$ to obtain the waypoint candidates for all foreground and background agents. Following standard approach [24], we add a random arrival time interval (following an exponential distribution) between each agent to distribute them temporally without crowding.

3.3. Waypoint Filter

Waypoint candidates are sampled independently for each agent and therefore do not reflect relation-specific constraints across agents. We introduce a filtering stage to remove combinations that violate basic geometric and map conditions. As in DDTS [39], we construct a coarse *prior trajectory* from the candidate waypoints of each agent $v_i \in \tilde{\mathcal{V}}$ using a cubic Spline, *i.e.*

$$\mathcal{X}^i_{\text{pr}} := \left\{ x^{i,m}_{\text{pr}} = \textsc{Spline}(w^m) : w^{i,m} \in \mathcal{W}^i \right\}. \tag{3}$$

Since there are N agents in total, let $\mathcal{X}_{\text{pr}} := \mathcal{X}^1_{\text{pr}} \times \cdots \times \mathcal{X}^N_{\text{pr}}$ be the set of all possible candidate combinations of prior trajectories for all agents in $\tilde{\mathcal{V}}$.

Constraint Validator. We use a large language model to convert the user instruction and the social graph into a compact constraint domain-specific language (DSL) [7, 11], a formal specification designed to capture domain-level semantics in a structured form. For every candidate combination $\boldsymbol{x}_{\text{pr}} \in \mathcal{X}_{\text{pr}}$ where $\boldsymbol{x}_{\text{pr}} = (x^1_{\text{pr}}, \ldots, x^N_{\text{pr}})$, we check with the DSL validators to remove obvious failures such as being off-road, overly short, or missing geometric conditions for the intended relation. For example, in a collision scene, the selected trajectories should pass through a common region with a small temporal offset; in a yielding scene, they should place the relevant agents near the same crosswalk with compatible spatial and temporal layout.

Candidate Selection. After passing the validators, we select the most probable combination that survives in $\mathcal{X}_{\text{pr}}$ by the *negative log-likelihood* (NLL) given by the GMM, *i.e.* $\boldsymbol{x}^\star_{\text{pr}} = \arg\min_{\boldsymbol{x}_{\text{pr}} \in \mathcal{X}_{\text{pr}}} \sum_{i=i}^{N} \text{NLL}(x^i_{\text{pr}})$, where

$$\text{NLL}(x^i_{\text{pr}}) := -\sum_{i=1}^{N} \log p_{\text{wp}}(w^i|\cdot), \tag{4}$$

and w^i is the waypoint associated with x^i_{pr}. The original trajectory only provides a geometrically plausible initialization for the reasoning stage and does not yet contain the final targeted relation behavior between agents instructed by users. This will be handled by the following stages.

3.4. Event Reasoner

Although the filtered trajectories are geometrically valid, they do not always satisfy the target relation specified by the user. We therefore introduce a reasoning stage to transform the coarse prior trajectories $\boldsymbol{x}^\star_{\text{pr}}$ into a structured behavioral program that enforces the desired relation while remaining grounded to the map. The reasoner follows a programmatic reasoning paradigm, where the LLM decomposes the task into structured behavioral segments, while the execution is delegated to deterministic trajectory editing modules, similar to program-aided language models [11, 37].

Event Summary. The reasoner takes as input a structured event summary extracted from the prior trajectories by a large language model, together with the social graph $\mathcal{G}$ and scene facts derived from the map, such as lane and crosswalk semantics, agent-to-map associations, and candidate stopping or goal anchors. The event summary records trajectory snippets, temporal boundaries, and key evidence for the target scenario including crosswalk entry and exit, stop-line reach, and closest approach between the relevant agents. Specifically, the prior trajectory x^i_{pr} for agent v^i is summarized as a sequence of semantic anchors $\mathcal{A}^i := \{a^{i,1}, a^{i,2}, \dots\}$ and their corresponding times $\mathcal{T}^i := \{t^{i,1}, t^{i,2}, \dots\}$. Each anchor denotes a grounded map entity such as a crosswalk entry, stop line, or goal region. This representation provides structured evidence for the LLM reasoner and reduces the ambiguity of inferring behavior directly from raw coordinates.

Behavioral Inference. Based on the event summary and the relations $\mathcal{E}$ in the social graph, the reasoner then infers a sequence of behavioral segments $\Phi^i := \{\phi^{i,k}, k = 1, 2, \dots\}$ for each agent v^i. Each segment $\phi^{i,k} \in \Phi^i$ is defined as

$$\phi^{i,k} := \left(t^{i,k-1}, a^{i,k}, s^{i,k}\right), \quad t^{i,k} \in \mathcal{T}^i, \quad a^{i,k} \in \mathcal{A}^i, \tag{5}$$

where $s^{i,k}$ is a discrete *status code* representing the behavior of the agent v^i during segment $\phi^{i,k}$ (*e.g.* `Keep`, `Stop`, `Truncate` or `Proceed`). Each segment $\phi^{i,k}$ corresponds to the time interval $[t^{i,k-1}, t^{i,k}]$, during which the agent evolves from anchor $a^{i,k-1}$ to $a^{i,k}$ under the control of $s^{i,k}$ (detailed in Sec. A.1). We set $t^{i,0}$ as the spawn time and $a^{i,0}$ as the entry point of the agent.

Conceptually, Φ^i indicates how $\boldsymbol{x}^\star_{\text{pr}}$ should be modified to satisfy the intended relation. For instance, the vehicle in a *yield_to* relation is rewritten into the `Keep-Stop-Proceed` pattern anchored to the crosswalk; a collision scene can be converted into trajectories that terminate at the point of collision. The reasoner is the interface between symbolic relation reasoning and geometric trajectory generation. By separating behavior inference from continuous motion synthesis, the framework can enforce user-specified relation semantics while retaining flexibility in the later trajectory editing and refinement steps.

3.5. Trajectory Refiner

The trajectories produced by the reasoning stage provide a coarse, piecewise-structured prior but may lack smoothness and realistic motion dynamics. We therefore refine them using a learned trajectory prediction model. Specifically, we first translate each $\phi^{i,k} \in \Phi^i$ to actual coordinates and concatenate them to build a *modified* prior trajectory $\tilde{\boldsymbol{x}}^i_{\text{pr}}$. Then we use a finetuned DL trajectory prediction model under the supervision of $\tilde{\boldsymbol{x}}^i_{\text{pr}}$ to generate realistic and physically grounded agent trajectories.

Prior Modification. For each agent v_i with the original prior trajectory x^i_{pr}, every segment $\phi^{i,k} \in \Phi^i$ corresponds a snippet of the prior trajectory $x^i_{\text{pr}}[t^{i,k-1}{:}t^{i,k}]$. Based on the assigned status code $s^{i,k}$, the trajectory snippet is either preserved (`Keep`), replaced by the stationary spatial anchor $a^{i,k}$ (`Stop`), or populated by the deferred remainder of the original path (`Proceed`) $x^i_{\text{pr}}[t^{i,k-1}{:}]$, producing the modified snippet of the prior trajectory $\tilde{x}^i_{\text{pr}}[t^{i,k-1}{:}t^{i,k}]$. Spatial continuity during segment transition is strictly enforced, and necessary interpolation or resampling is performed to ensure time step consistency in each segment. The modified prior trajectory of agent v^i is given by

$$\tilde{x}^i_{\text{pr}} := \text{Concat}\left(\left\{\tilde{x}^i_{\text{pr}}[t^{i,k-1}{:}t^{i,k}] : \phi^{i,k} \in \Phi^i\right\}\right), \tag{6}$$

and we repeat for every agent $v^i \in \tilde{\mathcal{V}}$.

Neural Refinement. Following DDTS [39] (Algorithm 2), we adopt a custom finetuned deep neural network (DNN) predictor with *goal supervision* that models inter-agent dependencies across the whole scene and generates the refined trajectories, *i.e.*

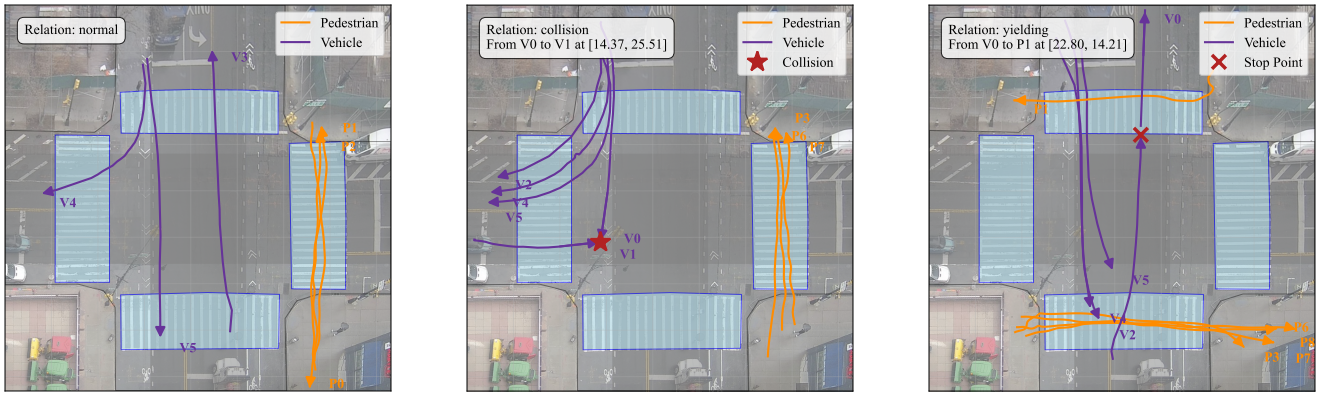
$$\boldsymbol{x}_{\text{rf}} := \textsc{Dnn}(\boldsymbol{x}_{\text{ob}}; \tilde{\boldsymbol{x}}_{\text{pr}}). \tag{7}$$

We use the a short snippet of the initial prior trajectory $\boldsymbol{x}^\star_{\text{pr}}$ as the initial observation input $\boldsymbol{x}_{\text{ob}}$, and the extracted segment goals $\tilde{\boldsymbol{x}}_{\text{pr}} := (\boldsymbol{x}^1_{\text{pr}}, \dots, \boldsymbol{x}^N_{\text{pr}})$ as a supervising signal. Then we proceed in an iterative scheme and use the previous prediction output $\boldsymbol{x}_{\text{ob}} \leftarrow \textsc{Concat}(\{\boldsymbol{x}_{\text{ob}}, \boldsymbol{x}_{\text{rf}}\})$ as the new observation input in the next refinement round. By conditioning on the edited prior and the segment goals, the refined trajectories $\boldsymbol{x}_{\text{rf}}$ preserves the intended relation semantics while restoring realistic motion continuity.

4. Evaluation

4.1. Dataset Description

Using the pipeline described in Sec. 3, we generate a dataset of 370 synthetic traffic scenes, including 100 normal scenes and 270 rare-event scenes (40 collision and 50 yielding for each of the three different LLM backends, respectively). To obtain a composite collection of the traffic data of an urban intersection, we follow DDTS and utilize the Cosmos-Trajectory dataset [39]. It comprises real world pedestrian and vehicle trajectories extracted from a New York City intersection within the COSMOS Testbed [15, 28], as well as spatiotemporal trajectory distributions (*i.e.* p_{td} and p_{wp}) fitted on the collected data.



(a) Normal. (b) Collision. (c) Yielding.

Figure 3. Example scenes of different agent relations in the dataset. Collision and yielding scenes are annotated with the agents involved, the event locations, and the time of occurrence.

Table 1. Simulation-to-reality gaps of the synthesized trajectories from the Claude3.5-Sonnet subset for different relations and agent classes. The first row provides the reference NLLs for normal scenes. FG and BG denote the foreground and background agents, respectively.

Relation	#Scenes	Agent	Pedestrian	Vehicle
Normal	100	–	96.8	170.9
Collision	40	FG	**0.43**	0.01
		BG	0.02	< 0.01
Yielding	50	FG	**0.65**	**2.19**
		BG	0.03	< 0.01

Trajectory Prediction Model. Using the Cosmos-Trajectory dataset, we finetune several state-of-the-art trajectory prediction models including TrajNet++ [16], Trajectron++ [29], PPT [20], and Unitraj++ [42]. To utilized the models in the data synthesis pipeline, we train them under a goal-supervision scheme whenever possible. We observe that while advanced models such as the Unitraj++ family produce moderately better results in terms of the prediction error, they are often less flexible in the presence of missing values, which is very common in both the Cosmos-Trajectory dataset and real world scenarios. On the other hand, TrajNet++ performs reasonably well and is strongly dictated by the provided supervision signal, making it a suitable choice for the proposed pipeline.

Data Synthesis Protocol. For each LLM backend, we generate 90 scenes including 40 with collisions, 50 with yielding, and 100 normal scenes for reference, exemplified in Fig. 3. Each scene exhibits medium traffic density with 2 foreground agents and 5 to 10 background agents (see details in Sec. 3.2). For every scene, we record agent information such as the agent class, spawn time, and *refined* trajectory, as well as the summary of the event (*e.g.* "vehicle-2 yields to pedestrian-1"). We provide detailed description of the generated data format in Sec. A.2. We mainly consider the 190 scenes generated using the Claude3.5-Sonnet backend [3], but also provide statistics on the scenes generated by instructions-tuned Qwen2.5-7B and 3B [27] backends on an NVIDIA A100 GPU in Sec. 4.3.

4.2. Simulation Fidelity

Although the synthesized trajectories $\boldsymbol{x}_{\text{rf}}$ are produced by a DNN finetuned on real world data, it is supervised by a prior $\tilde{\boldsymbol{x}}_{\text{pr}}$ that is modified by the LLM reasoner (Secs. 3.4 and 3.5). We evaluate the quality of the refined trajectories by comparing to the spatial trajectory distribution p_{wp} using the simulation-to-reality (S2R) gap. Let $\mathcal{X}_R$ be the set of trajectories corresponding to the agents involved in relation R, and $\boldsymbol{x}_{\text{nm}}$ be the set of normal trajectories. Denote $\text{NLL}(\mathcal{X})$ as the average NLL over the set of trajectories $\mathcal{X}$ given by Eq. (4). Then the S2R gap of relation R is defined as the difference between the average NLL of normal trajectories and rl trajectories, *i.e.*

$$\text{S2R}_R := \frac{\text{NLL}(\mathcal{X}_{\text{nm}}) - \text{NLL}(\mathcal{X}_R)}{\text{NLL}(\mathcal{X}_{\text{nm}})}. \tag{8}$$

We provide the results in Tab. 1, stratified by agent classes and whether they are in the foreground or the background (see details in Sec. 3.2). We report the S2R gaps of other LLM backends in Appendix B.

It can be seen that the S2R gaps of *background* agents are negligible regardless of agent classes and relations, indicating that their trajectories closely resemble the real world agent trajectories. For the *foreground* agents, however, there are notable S2R gaps under collision and yielding for both

Table 2. Instruction satisfaction rates of different LLM backends for collision and yielding scene generation.

LLM-Backend	Collision	Yielding	Overall
Claude3.5-Sonnet	**90.0**	**80.0**	**84.44**
Qwen2.5-7B	72.5	62.0	66.66
Qwen2.5-3B	57.5	56.0	56.83

pedestrians and vehicles. Intuitively, the trajectories reflected by these relations are hardly observed in real world data (*e.g*. the Cosmos-Trajectory dataset contains predominantly normal scenes), and are thus not captured by the statistical models. In other words, it is not statistically feasible to learn these relations from real world data, and the ability of our pipeline in generating and annotating rare events is crucial for quantitative safety-critical research.

4.3. Pipeline Robustness

Occasionally, the generated scenes may not fully satisfy the scene specifications due to errors in the reasoning stage or violations of structural constraints. To assess the robustness of the data synthesis pipeline described in Sec. 3, we measure the *instruction satisfaction rate* (ISR), *i.e*. the proportion of valid scenes among all generated scenes given an LLM backend for collision and yielding. The validity of each generation is manually verified based on geometric consistency and compliance with the intended agent relation. The results are summarized in Tab. 2.

As expected, the Claude3.5-Sonnet [3] backend achieves the highest overall ISR of 84.44%, and performs consistently well on both collision and yielding cases. For smaller models, Qwen2.5-7B and 3B [27] models achieve overall ISRs of 66.66% and 56.83%, respectively. The yielding scenario is generally more difficult than collision, as it requires consistent temporal ordering and spatial alignment between agents. Since the synthesis pipeline runs offline, it is practical to use a stronger backend when available, although smaller models can still produce reasonable results.

5. Agentic Traffic Scene Understanding

In this section, we showcase the usage of the synthetic dataset for traffic scene understanding using a simple agentic pipeline. For each scene in the dataset which starts at time t_s, suppose we observe the trajectories of the agents up to time $t_s + \Delta t_{ob}$, where Δt_{ob} is a short observation window. We test two basic tasks:

- *Intent Prediction:* Given the observed trajectories, the learned statistical models, and the map features, the task is to predict from which direction will each agent eventually exit the intersection.
- *Relation Identification:* Further given the finetuned trajectory prediction model, the task is to identify the relations between the agents in the scene (*i.e*. *collision*, *yielding*, or *normal*).

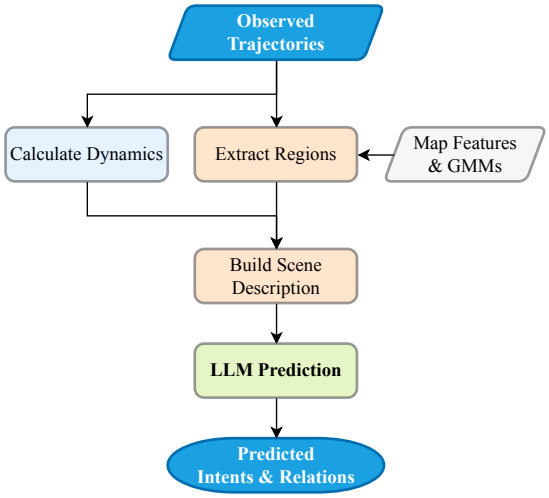


Figure 4. A simple agentic pipeline for traffic scene understanding (agent intent prediction and relation identification) using the synthesized dataset.

For both tasks, we consider the agentic pipeline illustrated in Fig. 4, and compare them respectively against a non-LLM baseline. The goal is to determine whether an agentic system is able to perform early perception and precognition in real world traffic.

Since the map features and the GMMs induce a semantic description of agent entrance and exit directions (*e.g*. a vehicle might enter from the *west* and exit from the *east*), we avoid raw coordinates and use this textual representation of the observed entrance directions for LLM inference. Similarly, we use the textual description of the exit directions for the ground truth labels of intent prediction. As described in DDTS [39], each GMM component corresponds to a specific entrance and exit direction. Thus we can compare the distances between the observed trajectory of an agent with the *mean trajectory* of each GMM component to obtain a simple statistical prediction as the baseline. This is a standard classification task which can be measured by the *classification accuracy*.

On the other hand, predicting the relations between the agents requires not only their anticipated future directions, but also dynamic features such as velocities and accelerations. Indeed, whether a vehicle slows down before a pedestrian can make the difference between collision and yielding. We also utilize a finetuned TrajNet++ to generate the potential future trajectories of all agents in the scene based on the observations. Then the entire (observed and predicted) trajectories are processed by a rule-based checker to

Table 3. Performance of agentic intention prediction measured by the intent classification accuracy, and relation identification measured by the agent-level F1-score. The average inference time per scene is measured on an NVIDIA RTX4090 GPU.

Method	Intent Prediction (Accuracy)			Relation Identification (F1-score)				Inf Time (s/scene)
	Pedestrian	Vehicle	Overall	Normal	Collision	Yielding	Overall	
GMM	0.39	30.72	12.76	–	–	–	–	< 0.01
TrajNet++	–	–	–	98.49	9.84	6.49	31.65	0.03
Qwen2.5-1.5B	**7.03**	15.36	11.73	62.59	10.96	2.47	33.89	0.76
Qwen2.5-7B	6.64	**33.73**	**21.94**	**84.57**	**20.00**	**9.43**	**48.20**	**3.54**

determine whether there are any *close encounters* between the agents. We first use the rule-based predictions as the baseline result, and then convert it into textual descriptions along with the map features for LLM reasoning. Since there are multiple agents in the scene, the model may predict the correct relation but between the wrong agents (*e.g*. the collision is between *vehicle-1* and *pedestrian-2*, but the model predicts *pedestrian-3*). Therefore, we use an *agent-level F1-score*, where true-positive is only achieved when both the relation and the agents involved are correctly identified.

We test the intent prediction and relation identification using instructions-tuned Qwen2.5-1.5B and 7B [27] backends on an NVIDIA RTX4090 GPU, and report their results in Tab. 3.

Intent Prediction Results. The larger 7B model demonstrates the strongest overall performance, achieving a leading accuracy of 21.94. While the GMM baseline performs reasonably well for vehicles with an overall accuracy of 30.72, it almost completely fails to predict pedestrian intent. Meanwhile, both LLMs evidently improve pedestrian intent accuracy. Even though the final accuracy of 9.43 given by Qwen2.5-7B is far from satisfaction, this still suggests that LLMs are better at semantic reasoning and understanding social constraints.

Relation Identification Results. The larger Qwen2.5-7B model again achieves the highest overall agent-level F1-score of 48.20 and significantly outperforms the TrajNet++ baseline of 31.65, while the 1.5B model is only marginally better than the baseline. This demonstrates the superiority of sufficiently large LLMs in identifying safety-critical events. Specifically, the Qwen2.5-7B model doubled the baseline accuracy of collision detection. Not surprisingly, the enhanced situational reasoning ability of LLMs comes with the cost of computation time. The 3.54 seconds latency for the Qwen2.5-7B model is a non-trivial amount of time, especially in safety-critical scenarios.

Note that the agentic pipeline discussed in this section is preliminary and serves only as a proof-of-concept. Instead of attempting to achieve state-of-the-art results, our primary objective is to demonstrate the dataset usage by the quantitative evaluation of agentic traffic perception and precognition systems, especially in the presence of safety-critical rare events. Future research is needed in developing more advanced LLM-driven architectures for traffic scene understanding.

6. Conclusion and Future Work

We introduce a multi-stage traffic simulation and rare event synthesis pipeline based on an urban intersection. The simulation framework effectively bridges the gap between natural language instructions and physical constraints by decomposing semantic reasoning and physical execution. We provide a synthetic dataset of 200 traffic scenes with social interaction annotations, including safety-critical cases such as collision and yielding that rarely occurs in real world datasets. The generated dataset provides a meaningful benchmark for future development of intelligent traffic systems that are robust in safety-critical scenarios. We conduct quantitative evaluations of the pipeline quality, showing that the pipeline is able to achieve an ISR of 84% with the Claude3.5-Sonnet backend. We further demonstrate the dataset usage by agentic traffic scene perception and precognition including agent intent prediction and relation identification.

The proposed pipeline is applied to one real world traffic intersection, for which we have extensive map features and learned statistical priors. The representativeness of the planner and reasoner is also confined by the semantic description provided to the LLM backend, and the generalization across different intersections remains to be studied. Future work will include exploring diverse geometric locations and developing a universally applicable pipeline which could work out-of-the-box for general urban traffic scenes.

Acknowledgments

This work was supported in part by NSF grant CNS-2450567 and EEC-2133516, NSF grant CNS-2038984 and corresponding support from the Federal Highway Administration (FHA), MediaTek Inc USA, NSF grant

CNS-2148128 and by funds from federal agency and industry partners as specified in the Resilient & Intelligent NextG Systems (RINGS) program, and ARO grant W911NF2210031. We would also like to thank the anonymous reviewers for their suggestions and insights that improved this work.

Data Availability

Dataset is available at this link.

References

[1] Aizierjiang Aiersilan. Generating Traffic Scenarios via In-Context Learning to Learn Better Motion Planner. *Proceedings of the AAAI Conference on Artificial Intelligence*, 39 (14):14539–14547, 2025. 2

[2] Alexandre Alahi, Kratarth Goel, Vignesh Ramanathan, Alexandre Robicquet, Li Fei-Fei, and Silvio Savarese. Social LSTM: Human Trajectory Prediction in Crowded Spaces. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 2

[3] Anthropic. Claude 3.5 Sonnet. Technical report, Anthropic, 2024. 6, 7

[4] 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 (CVPR)*, 2020. 2

[5] Wei-Jer Chang, Chen Tang, Chenran Li, Yeping Hu, Masayoshi Tomizuka, and Wei Zhan. Editing Driver Character: Socially-Controllable Behavior Generation for Interactive Traffic Simulation. *IEEE Robotics and Automation Letters*, 8(9):5432–5439, 2023. 2

[6] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. CARLA: An Open Urban Driving Simulator. In *Proceedings of the 1st Annual Conference on Robot Learning*, pages 1–16. PMLR, 2017. 2

[7] Karim Elmaaroufi, Devan Shanker, Ana Cismaru, Marcell Vazquez-Chanlatte, Alberto Sangiovanni-Vincentelli, Matei Zaharia, and Sanjit A. Seshia. ScenicNL: Generating Probabilistic Scenario Programs from Natural Language. In *First Conference on Language Modeling*, 2024. 4

[8] Scott Ettinger, Shuyang Cheng, Benjamin Caine, Chenxi Liu, Hang Zhao, Sabeek Pradhan, Yuning Chai, Ben Sapp, Charles R. Qi, Yin Zhou, Zoey Yang, Aurélien Chouard, Pei Sun, Jiquan Ngiam, Vijay Vasudevan, Alexander McCauley, Jonathon Shlens, and Dragomir Anguelov. Large Scale Interactive Motion Forecasting for Autonomous Driving: The Waymo Open Motion Dataset. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 9710–9719, 2021. 2

[9] Lan Feng, Quanyi Li, Zhenghao Peng, Shuhan Tan, and Bolei Zhou. TrafficGen: Learning to Generate Diverse and Realistic Traffic Scenarios. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3567–3575, London, United Kingdom, 2023. IEEE. 2

[10] Yongjie Fu, Mehmet Kerem Turkcan, Mahshid Ghasemi, Zhaobin Mo, Chengbo Zang, Abhishek Adhikari, Zoran Kostic, Gil Zussman, and Xuan Di. AI-Powered CPS-Enabled Vulnerable-User-Aware Urban Transportation Digital Twin: Methods and Applications. *IEEE Transactions on Intelligent Transportation Systems*, pages 1–18, 2026. 1

[11] Luyu Gao, Aman Madaan, Shuyan Zhou, Uri Alon, Pengfei Liu, Yiming Yang, Jamie Callan, and Graham Neubig. PAL: Program-aided Language Models. In *Proceedings of the 40th International Conference on Machine Learning*, pages 10764–10799. PMLR, 2023. 4

[12] Agrim Gupta, Justin Johnson, Li Fei-Fei, Silvio Savarese, and Alexandre Alahi. Social GAN: Socially Acceptable Trajectories With Generative Adversarial Networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 2

[13] Sepp Hochreiter and Jürgen Schmidhuber. Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780, 1997. 2

[14] Minwoo Jeong, Jeeyun Chang, and Yoonjin Yoon. Speak to Simulate: An LLM-Guided Agentic Framework for Traffic Simulation in SUMO. In *Proceedings of the 8th ACM SIGSPATIAL International Workshop on Geospatial Simulation*, pages 45–48, The Graduate Hotel Minneapolis Minneapolis MN USA, 2025. ACM. 1

[15] Zoran Kostic, Alex Angus, Zhengye Yang, Zhuoxu Duan, Ivan Seskar, Gil Zussman, and Dipankar Raychaudhuri. Smart City Intersections: Intelligence Nodes for Future Metropolises. *Computer*, 55(12):74–85, 2022. 5

[16] Parth Kothari, Sven Kreiss, and Alexandre Alahi. Human Trajectory Forecasting in Crowds: A Deep Learning Perspective. *IEEE Transactions on Intelligent Transportation Systems*, 23(7):7386–7400, 2022. 6

[17] Siqi Lai, Zhao Xu, Weijia Zhang, Hao Liu, and Hui Xiong. LLMLight: Large Language Models as Traffic Signal Control Agents. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V.1*, pages 2335–2346, Toronto ON Canada, 2025. ACM. 2

[18] Alon Lerner, Yiorgos Chrysanthou, and Dani Lischinski. Crowds by Example. *Computer Graphics Forum*, 26(3):655–664, 2007. 2

[19] Xin Li, Xiaowen Ying, and Mooi Choo Chuah. GRIP: Graph-based Interaction-aware Trajectory Prediction. In *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, pages 3960–3966, Auckland, New Zealand, 2019. IEEE. 3

[20] Xiaotong Lin, Tianming Liang, Jianhuang Lai, and Jian-Fang Hu. Progressive Pretext Task Learning for Human Trajectory Prediction. In *Computer Vision – ECCV 2024*, pages 197–214. Springer Nature Switzerland, Cham, 2025. Series Title: Lecture Notes in Computer Science. 6

[21] Yao Liu, Binghao Li, Xianzhi Wang, Claude Sammut, and Lina Yao. Attention-Aware Social Graph Transformer Networks for Stochastic Trajectory Prediction. *IEEE Transactions on Knowledge and Data Engineering*, 36(11):5633–5646, 2024. 3

[22] Pablo Alvarez Lopez, Evamarie Wiessner, Michael Behrisch, Laura Bieker-Walz, Jakob Erdmann, Yun-Pang Flotterod,

Robert Hilbrich, Leonhard Lucken, Johannes Rummel, and Peter Wagner. Microscopic Traffic Simulation using SUMO. In *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pages 2575–2582, Maui, HI, 2018. IEEE. 1, 2

[23] Jiageng Mao, Junjie Ye, Yuxi Qian, Marco Pavone, and Yue Wang. A Language Agent for Autonomous Driving. In *First Conference on Language Modeling*, 2024. 2

[24] Adolf D. May and Adolf Darlington May. *Traffic flow fundamentals*. Prentice Hall, Englewood Cliffs, N.J., 1990. 4

[25] Abduallah Mohamed, Kun Qian, Mohamed Elhoseiny, and Christian Claudel. Social-STGCNN: A Social Spatio-Temporal Graph Convolutional Neural Network for Human Trajectory Prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 3

[26] S Pellegrini, A Ess, K Schindler, and L Van Gool. You'll never walk alone: Modeling social behavior for multi-target tracking. In *2009 IEEE 12th International Conference on Computer Vision*, pages 261–268, Kyoto, 2009. IEEE. 2

[27] Qwen, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 Technical Report, 2024. Version Number: 2. 6, 7, 8

[28] Dipankar Raychaudhuri, Ivan Seskar, Gil Zussman, Thanasis Korakis, Dan Kilper, Tingjun Chen, Jakub Kolodziejski, Michael Sherman, Zoran Kostic, Xiaoxiong Gu, Harish Krishnaswamy, Sumit Maheshwari, Panagiotis Skrimponis, and Craig Gutterman. Challenge: COSMOS: A city-scale programmable testbed for experimentation with advanced wireless. In *Proceedings of the 26th Annual International Conference on Mobile Computing and Networking*, pages 1–13, London United Kingdom, 2020. ACM. 5

[29] Tim Salzmann, Boris Ivanovic, Punarjay Chakravarty, and Marco Pavone. Trajectron++: Dynamically-Feasible Trajectory Forecasting with Heterogeneous Data. In *Computer Vision – ECCV 2020*, pages 683–700. Springer International Publishing, Cham, 2020. Series Title: Lecture Notes in Computer Science. 2, 6

[30] Qiao Sun, Xin Huang, Brian C. Williams, and Hang Zhao. InterSim: Interactive Traffic Simulation via Explicit Relation Modeling. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 11416–11423, Kyoto, Japan, 2022. IEEE. 2

[31] Simon Suo, Sebastian Regalado, Sergio Casas, and Raquel Urtasun. TrafficSim: Learning To Simulate Realistic Multi-Agent Behaviors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10400–10409, 2021. 2

[32] Shuhan Tan, Boris Ivanovic, Xinshuo Weng, Marco Pavone, and Philipp Kraehenbuehl. Language Conditioned Traffic Generation. In *Proceedings of The 7th Conference on Robot Learning*, pages 2714–2752. PMLR, 2023. 2

[33] Shuhan Tan, Boris Ivanovic, Yuxiao Chen, Boyi Li, Xinshuo Weng, Yulong Cao, Philipp Kraehenbuehl, and Marco Pavone. Promptable Closed-loop Traffic Simulation. In *8th Annual Conference on Robot Learning*, 2024. 2

[34] Shuncheng Tang, Zhenya Zhang, Jixiang Zhou, Lei Lei, Yuan Zhou, and Yinxing Xue. LeGEND: A Top-Down Approach to Scenario Generation of Autonomous Driving Systems Assisted by Large Language Models. In *Proceedings of the 39th IEEE/ACM International Conference on Automated Software Engineering*, pages 1497–1508, Sacramento CA USA, 2024. ACM. 2

[35] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention Is All You Need, 2017. arXiv:1706.03762 [cs]. 2

[36] Pei Xu, Jean-Bernard Hayet, and Ioannis Karamouzas. Context-Aware Timewise VAEs for Real-Time Vehicle Trajectory Prediction. *IEEE Robotics and Automation Letters*, 8 (9):5440–5447, 2023. 2

[37] Shunyu Yao, Jeffrey Zhao, Dian Yu, Izhak Shafran, Karthik R. Narasimhan, and Yuan Cao. ReAct: Synergizing Reasoning and Acting in Language Models. In *NeurIPS 2022 Foundation Models for Decision Making Workshop*, 2022. 4

[38] Yu Yao, Salil Bhatnagar, Markus Mazzola, Vasileios Belagiannis, Igor Gilitschenski, Luigi Palmieri, Simon Razniewski, and Marcel Hallgarten. AGENTS-LLM: Augmentative GENeration of Challenging Traffic Scenarios with an Agentic LLM Framework. In *2025 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 18400–18407, Hangzhou, China, 2025. IEEE. 2

[39] Chengbo Zang, Mehmet Kerem Turkcan, Gil Zussman, Javad Ghaderi, and Zoran Kostic. Data-Driven Traffic Simulation for an Intersection in a Metropolis. In *The First Workshop on Populating Empty Cities – Virtual Humans for Robotics and Autonomous Driving at CVPR 2024*, 2024. 2, 4, 5, 7

[40] Zhejun Zhang, Christos Sakaridis, and Luc Van Gool. TrafficBots V1.5: Traffic Simulation via Conditional VAEs and Transformers with Relative Pose Encoding, 2024. arXiv:2406.10898 [cs]. 2

[41] Yongqi Zhao, Wenbo Xiao, Tomislav Mihalj, Jia Hu, and Arno Eichberger. Chat2Scenario: Scenario Extraction From Dataset Through Utilization of Large Language Model. In *2024 IEEE Intelligent Vehicles Symposium (IV)*, pages 559–566, Jeju Island, Korea, Republic of, 2024. IEEE. 2

[42] Yuanshao Zhu, James Jianqiao Yu, Xiangyu Zhao, Xun Zhou, Liang Han, Xuetao Wei, and Yuxuan Liang. UniTraj: Learning a Universal Trajectory Foundation Model from Billion-Scale Worldwide Traces. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. 6

Language-Guided Traffic Simulation and Rare Event Synthesis for an Urban Intersection

Supplementary Material

A. Implementation Details

A.1. Behavioral Segment

For the k-th behavioral segment $\phi^{i,k} \in \Phi^i$ of agent $v_i \in \mathcal{V}$, we define the possible status of the agent as follows (see Sec. 3.4):

- `Keep`: The agent maintains their trajectory along the waypoints specified by the prior trajectory x^i_{pr};
- `Stop`: The agent stops at the target anchor $a^{i,k}$ specified by the reasoner.
- `Proceed`: The agent resumes motion towards the subsequent goal.
- `Terminate`: The agent trajectory is truncated at the anchor due to events like collision.

A.2. Data Format

We generate each traffic scene in the following format, with annotations of scene meta, agent information, and event summary.

Listing 1. Data Schema

```
{
    "meta": {
        "scene_id": string,
        "map_id": int,
        "timestamp": float,
        "user_prompt": string
    },
    "agents": [{
        "id": int,
        "type": "pedestrian | vehicle",
        "spawn_time": float,
        "goal": [float, float],
        "trajectory": [[float, float], ...]
    }, ...],
    "events": [{
        "type": "collision" | "yielding" |
    "normal",
        "from": int,
        "to": int,
        "time": float | null,
        "location": [float, float] | null
    }, ...]
}
```

B. Additional Results

We calculate the S2R gaps of different LLM backends to evaluate the fidelity of the generated trajectories (see details in Sec. 4.2). The results are given in Tabs. 4 and 5. They are consistent with the aforementioned findings that the proposed pipeline is able to synthesize statistically rare events.

Table 4. Simulation-to-reality gaps of the synthesized trajectories in the dataset using the Qwen2.5-7B backend for different relations and agent classes. FG and BG denote the foreground and background agents, respectively.

Interaction	Agent	Pedestrian	Vehicle
Collision	FG	-	<0.01
	BG	0.02	<0.01
Yielding	FG	**0.72**	0.14
	BG	0.02	<0.01

Table 5. Simulation-to-reality gaps of the synthesized trajectories in the dataset using the Qwen2.5-3B backend for different relations and agent classes. FG and BG denote the foreground and background agents, respectively.

Interaction	Agent	Pedestrian	Vehicle
Collision	FG	-	**0.12**
	BG	0.06	0.01
Yielding	FG	**0.32**	**0.12**
	BG	<0.01	<0.01