Language Conditioned Traffic Generation Supplementary Material

Anonymous Author(s)
Affiliation
Address
email

A Interpreter

A.1 Structured Representation Details

The map specific z^m is a 6-dim integer vector. Its first four dimensions denote the number of lanes in each direction (set as north for the ego vehicle). The fifth dimension represents the discretized distance in 5-meter intervals from the map center to the nearest intersection (0-5, 5-10...). The sixth dimension indicates the ego vehicle's lane id, starting from 1 for the rightmost lane.

For agent i, the agent-specific z_i^a is an 8-dim integer vector describing this agent relative to the ego vehicle. The first dimension denotes the quadrant index (1-4), where quadrant 1 represents the front-right of the ego vehicle. The second dimension is the discretized distance to the ego vehicle with a 20m interval, and the third denotes orientation (north, south, east, west). The fourth dimension indicates discretized speed, set in 2.5m/s intervals. The last four dimensions describe actions over the next four seconds (one per second) chosen from a discretized set of seven possible actions: lane changes (left/right), turns (left/right), moving forward, accelerating, decelerating, and stopping.

A.2 Generation prompts

The scenario generation prompt used for `Interpreter` consists of several sections:

1. **Task description**: simple description of task of scenario generation and output formats.
2. **Chain-of-thought prompting [1]**: For example, "summarize the scenario in short sentences", "explain for each group of vehicles why they are put into the scenario".
3. **Description of structured representation**: detailed description for each dimension of the structured representation. We separately inform the model Map and Actor formats.
4. **Guidelines**: several generation instructions. For example, "Focus on realistic action generation of the motion to reconstruct the query scenario".
5. **Few-shot examples**: A few input-output examples. We provide a Crash Report example.

We show the full prompt below:

Prompt 1: Full prompt for `Interpreter` scenario generation.

```
You are a very faithful format converter that translate natrual language traffic scenario
    descriptions to a fix-form format to appropriately describe the scenario with motion
    action. You also need to output an appropriate map description that is able to support
    this scenario. Your ultimate goal is to generate realistic traffic scenarios that
    faithfully represents natural language descriptions normal scenes that follows the
    traffic rule.

Answer with a list of vectors describing the attributes of each of the vehicles in the
    scenario.

Desired format:
```

Submitted to the 7th Conference on Robot Learning (CoRL 2023). Do not distribute.

```
Summary: summarize the scenario in short sentences, including the number of vehicles. Also
    explain the underlying map description.
Explaination: explain for each group of vehicles why they are put into the scenario and how
    they fullfill the requirement in the description.
Actor Vector: A list of vectors describing the attributes of each of the vehicles in the
    scenario, only output the values without any text:
- 'V1': [,,,,,,,]
- 'V2': [,,,,,,,]
- 'V3': [,,,,,,,]
Map Vector: A vector describing the map attributes, only output the values without any text:
- 'Map': [,,,,,]

Meaning of the Actor vector attribute:
- dim 0: 'pos': [-1,3] - whether the vehicle is in the four quadrant of ego vechile in the
    order of [0 - 'front left', 1 - 'back left', 2- 'back right', 3 - 'front right']. -1 if
     the vehicle is the ego vehicle.
- dim 1: 'distance': [0,3] - the distance range index of the vehicle towards the ego vehicle
    ; range is from 0 to 72 meters with 20 meters interval. 0 if the vehicle is the ego
    vehicle. For example, if distance value is 15 meters, then the distance range index is
    0.
- dim 2: 'direction': [0,3] - the direction of the vehicle relative to the ego vehicle, in
    the order of [0- 'parallel_same', 1-'parallel_opposite', 2-'perpendicular_up', 3-'
    perpendicular_down']. 0 if the vehicle is the ego vehicle.
- dim 3: 'speed': [0,20] - the speed range index of the vehicle; range is from 0 to 20 m/s
    with 2.5 m/s interval. For example, 20m/s is in range 8, therefore the speed value is
    8.
- dim 4-7: 'action': [0,7] - 4-dim, generate actions into the future 4 second with each two
    actions have a time interval of 1s (4 actions in total), the action ids are [0 - 'stop
    ', 1 - 'turn left', 2 - 'left lane change', 3- 'decelerate', 4- 'keep_speed', 5-'
    accelerate',  6-'right lane change', 7-'turn right'].

Meaning of the Map attributes:
- dim 0-1: 'parallel_lane_cnt': 2-dim. The first dim is the number of parallel same-
    direction lanes of the ego lane, and the second dim is the number of parallel opposite-
    direction lanes of the ego lane.
- dim 2-3: 'perpendicular_lane_cnt': 2-dim. The first dim is the number of perpendicular
    upstream-direction lanes, and the second dim is the number of perpendicular downstream-
    direction lanes.
- dim 4: 'dist_to_intersection': 1-dim. the distance range index of the ego vehicle to the
    intersection center in the x direction, range is from 0 to 72 meters with 5 meters
    interval. -1 if there is no intersection in the scenario.
- dim 5: 'lane id': 1-dim. the lane id of the ego vehicle, counting from the rightmost lane
    of the same-direction lanes, starting from 1. For example, if the ego vehicle is in the
     rightmost lane, then the lane id is 1; if the ego vehicle is in the leftmost lane,
    then the lane id is the number of the same-direction lanes.

Transform the query sentence to the Actor Vector strictly following the rules below:
- Focus on realistic action generation of the motion to reconstruct the query scenario.
- Follow traffic rules to form a fundamental principle in most road traffic systems to
    ensure safety and smooth operation of traffic. You should incorporate this rule into
    the behavior of our virtual agents (vehicles).
- Traffic rule: in an intersection, when the vehicles on one side of the intersection are
    crossing, the vehicles on the other side of the intersection should be waiting. For
    example, if V1 is crossing the intersection and V2 is on the perpendicular lane, then
    V2 should be waiting.
- For speed and distance, convert the unit to m/s and meter, and then find the interval
    index in the given range.
- Make sure the position and direction of the generated vehicles are correct.
- Describe the initialization status of the scenario.
- During generation, the number of the vehicles is within the range of [1, 32].
- Always generate the ego vehicle first (V1).
- Always assume the ego car is in the center of the scene and is driving in the positive x
    direction.
- In the input descriptions, regard V1, Vehicle 1 or Unit #1 as the ego vehicle. All the
    other vehicles are the surrounding vehicles. For example, for "Vehicle 1 was traveling
    southbound", the ego car is Vehicle 1.
- If the vehicle is stopping, its speed should be 0m/s (index 0). Also, if the first action
    is 'stop', then the speed should be 0m/s (index 0).
- Focus on the interactions between the vehicles in the scenario.
- Regard the last time stamp as the time stamp of 5 second into the future.

Generate the Map Vector following the rules below:
```

```
- If there is vehicle turning left or right, there must be an intersection ahead.
- Should at least have one lane with the same-direction as the ego lane; i.e., the first dim
     of Map should be at least 1. For example, if this is a one way two lane road, then the
     first dim of Map should be 2.
- Regard the lane at the center of the scene as the ego lane.
- Consider the ego car's direction as the positive x direction. For example, for "V1 was
    traveling northbound in lane five of a five lane controlled access roadway", there
    should be 5 lanes in the same direction as the ego lane.
- The generated map should strictly follow the map descriptions in the query text. For
    example, for "Vehicle 1 was traveling southbound", the ego car should be in the
    southbound lane.
- If there is an intersection, there should be at least one lane in either the upstream or
    downstream direction.
- If there is no intersection, the distance to the intersection should be -1.
- There should be vehicle driving vertical to the ego vehicle in the scene only when there
    is an intersection in the scene. For example, when the road is just two-way, there
    should not be any vehicle driving vertical to the ego vehicle.
- If no intersection is mentioned, generate intersection scenario randomly with real-world
    statistics.


Query: The crash occurred during daylight hours on a dry, bituminous, two-lane roadway under
     clear skies.  There was one northbound travel lane and one southbound travel lane with
     speed limit of 40 km/h (25 mph).   The northbound lane had a -3.6 percent grade and
    the southbound lane had a +3.6 percent grade.  Both travel lanes were divided by a
    double yellow line. A 2016 Mazda CX-3 (V1) was in a parking lot attempting to execute a
     left turn to travel south.  A 2011 Dodge Charger (V2/police car) was traveling north
    responding to an emergency call with lights sirens activated. V1 was in a parking lot (
    facing west) and attempted to enter the roadway intending to turn left.   As V1 entered
     the roadway it was impacted on the left side by the front of V2 (Event 1).  V1 then
    rotated counterclockwise and traveled off the west road edge and impacted an embankment
     with its front left bumper (Event 2).  After initial impact V2 continued on in a
    northern direction and traveling to final rest approximately 40 meters north of impact
    area facing north in the middle of the roadway.  V1 and V2 were towed from the scene
    due to damage.

Summary: V1 attempts to turn left from a parking lot onto a two-lane roadway and is struck
    by V2, a police car traveling north with lights and sirens activated. There are 2
    vehicles in this scenario. This happens on a parking lot to a two-lane two-way road
    with intersection.
Explanation:
- V1 (ego vehicle) is attempting to turn left from a parking lot onto the roadway. We cannot
     find V1's speed in the query. Because V1 tries to turn left, its initial speed should
    be set low. We set V1's speed as 5 m/s, which has the index of 2. V1 turns left, so its
     actions are all 1 (turn left).
- V2 is a police car traveling north with lights and sirens activated. As V1 is turning left
    , 5 seconds before the crash, V1 is facing west and V2 is coming from northbound,
    crossing the path of V1. In the coordinates of V1 (which is facing west initially), V2
    comes from the front and is on the left side. Hence, V2's position is "front left" (3).
     As V1 is facing west and V2 facing north, V2 is moving in the perpendicular down
    direction with V1. Therefore its direction is 3 (perpendicular_down). We cannot find V2
    's speed in the query. Because V2 is a police car responding to an emergency call, we
    assume V2's init speed is 10 m/s (index 4). Given this speed, V2's distance to V1 is 10
    m/s * 5s = 50m (index 10). V2 keeps going straight, so its actions are all 4 (keep
    speed).
- Map: V1 tries to turn left from a partking lot onto a two-lane roadway. There are a one-
    way exit lane from parking lot (one same-direction parallel) and the ego vehicle is in
    the left turn lane with lane id 1. On the perpendicular side there is a two-lane
    roadway. V1 is about to turn left, so the distance to the intersection is set to be 10m
     (index 2).
Actor Vector:
- 'V1': [-1, 0, 0, 2, 1, 1, 1, 1]
- 'V2': [0, 10, 3, 4, 4, 4, 4, 4]
Map Vector:
- 'Map': [1, 0, 1, 1, 2, 1]

Query: INSERT_QUERY_HERE

Output:
```

A.3 Instructional editing prompts

We also provide `Interpreter` another prompt for instructional scenario editing. This prompt follow a similar structure to the generation prompt. We mainly adopt the task description, guidelines, and examples to scenario editing tasks. Note that for the instructional editing task, we change the distance interval (second dimension) of agent-specific z_i^a from 20 meters to 5 meters. This is to ensure the unedited agents stay in the same region before and after editing.

We show the full prompt below:

Prompt 2: Full prompt for `Interpreter` instructional scenario editing.

```
You are a traffic scenario editor that edit fix-form traffic scenario descriptions according
     to the user's natural language instructions.

The user will input a fix-form traffic scenario description as well as the map description.
    The user also a natural language instruction to modify the scenario. You need to output
     a fix-form traffic scenario that is modified according to the instruction.

Input format:
- V1: [,,,,,,,]
- V2: [,,,,,,,]
- V3: [,,,,,,,]
- Map: [,,,,,]
Instruction: natural language instruction to modify the scenario.

Output format:
Summary: summarize the scenario in short sentences. summarize the user instruction, and
    indicate which part of the scenario should be modified.
Explaination: explain step-by-step how each part of the scenario is modified.
Actor Vector: A list of vectors describing the attributes of each of the vehicles. Only the
    vehicles that are modified should be included in the output.
- V2: [,,,,,,,]

Meaning of the Actor vector attribute:
- dim 0: 'pos': [-1,3] - whether the vehicle is in the four quadrant of ego vechile in the
    order of [0 - 'front left', 1 - 'back left', 2- 'back right', 3 - 'front right']. -1 if
     the vehicle is the ego vehicle.
- dim 1: 'distance': [0,14] - the distance range index of the vehicle towards the ego
    vehicle; range is from 0 to 72 meters with 5 meters interval. 0 if the vehicle is the
    ego vehicle.
- dim 2: 'direction': [0,3] - the direction of the vehicle relative to the ego vehicle, in
    the order of [0- 'parallel_same', 1-'parallel_opposite', 2-'perpendicular_up', 3-'
    perpendicular_down']. 0 if the vehicle is the ego vehicle.
- dim 3: 'speed': [0,8] - the speed range index of the vehicle; range is from 0 to 20 m/s
    with 2.5 m/s interval. For example, 20m/s is in range 8, therefore the speed value is
    8.
- dim 4-7: 'action': [0,7] - 4-dim, generate actions into the future 4 second with each two
    actions have a time interval of 1s (4 actions in total), the action ids are [0 - 'stop
    ', 1 - 'turn left', 2 - 'left lane change', 3- 'decelerate', 4- 'keep_speed', 5-'
    accelerate',  6-'right lane change', 7-'turn right'].

Meaning of the Map attributes:
- dim 0-1: 'parallel_lane_cnt': 2-dim. The first dim is the number of parallel same-
    direction lanes of the ego lane, and the second dim is the number of parallel opposite-
    direction lanes of the ego lane.
- dim 2-3: 'perpendicular_lane_cnt': 2-dim. The first dim is the number of perpendicular
    upstream-direction lanes, and the second dim is the number of perpendicular downstream-
    direction lanes.
- dim 4: 'dist_to_intersection': 1-dim. the distance range index of the ego vehicle to the
    intersection center in the x direction, range is from 0 to 72 meters with 5 meters
    interval. -1 if there is no intersection in the scenario.
- dim 5: 'lane id': 1-dim. the lane id of the ego vehicle, counting from the rightmost lane
    of the same-direction lanes, starting from 1. For example, if the ego vehicle is in the
     rightmost lane, then the lane id is 1; if the ego vehicle is in the leftmost lane,
    then the lane id is the number of the same-direction lanes.

Follow the instructions below:
- 'V1' is the ego vehicle, and the other vehicles are the surrounding vehicles.
```

```
- The user will input a fix-form traffic scenario description as well as the map description
    . The user also an natural language instruction to modify the scenario. You need to
    output a fix-form traffic scenario that is modified according to the instruction.
- First figure out which part of the scenario should be modified according to the
    instruction. For example, if the instruction is "the vehicle in front of me should turn
     left", then the vehicle in front of the ego vehicle should be modified.

Input:
Actor vector:
- V1: [-1,  0,  0,  0,  4,  4,  4,  4]
- V2: [ 2,  1,  0,  1,  4,  4,  4,  4]
- V3: [ 3,  3,  0,  1,  4,  4,  4,  0]
- V4: [ 3,  4,  0,  8,  4,  4,  2,  0]
- V5: [ 0,  9,  1,  8, -1,  4,  5, -1]
- V6: [ 3,  5,  0,  0,  0,  0,  0,  0]
- V7: [ 0,  9,  3,  0,  0,  0,  0,  0]
- V8: [ 3, 10,  3,  3,  4,  5,  1,  0]
- V9: [ 0, 10,  3,  0,  0,  0,  0, -1]
- V10: [ 3, 10,  2,  0,  0,  0,  0, -1]
- V11: [ 3, 11,  2,  0,  0,  0,  0,  0]
- V12: [ 3, 11,  2,  0,  0,  7,  0,  0]
- Map: [4, 3, 2, 3, 6, 4]

Instruction: move the vehicle behind the ego vehicle to the opposite lane and move faster.

Output:
Summary: The instruction is to move the vehicle behind the ego vehicle to the opposite lane
    and accelerate. First find which vehicle is behind the ego vehicle. There are only 1
    vechile behind the ego vehicle, that is V2 (with position=2, indicating on the right
    back side of the ego vehicle). Therefore, the vehicle V2 should be modified.
Explaination: The vehicle V2 is modified to move to the opposite lane and accelerate. The
    vehicle V2 is in the right back side of the ego vehicle, and the ego vehicle is in the
    rightmost lane of the same-direction lanes. Therefore, the vehicle V2 should move to
    the leftmost lane of the opposite-direction lanes. Therefore, V2's direction should be
    opposite to the ego vehicle, changed to 1 (parallel_opposite). In this lane, V2 should
    be moved to the left back of the ego car, its position should be changed to 1. V2
    should move faster, its speed should be changed to 10 (25 m/s).
Actor vector:
- V2: [ 1,  1,  1, 10,  4,  4,  4,  4]

Instruction: remove all the vehicles on the front of the ego car and moving in the same
    direction.

Output:
Summary: The instruction is to remove all the vehicles on the front of the ego car and
    moving in the same direction. First find which vehicles are on the front of the ego
    vehicle. V3-V12 are all on the front of the ego vehicle. Then, only V3, V4 and V6 has
    the same direction as the ego vehicle (0). Therefore, V3, V4 and V6 should be removed.
Explaination: V3, V4, V6 are on the front of the ego vehicle and moving in the same
    direction. V3, V4 and V6 are removed from the scenario.

Actor vector:
- V3: removed.
- V4: removed.
- V6: removed.

Input: INSERT_QUERY_HERE

Output:
```

B Generator

B.1 Training objectives

In the main paper, we show the full training objective of `Generator` as:

$$\mathcal{L}(p, \tau) = \mathcal{L}_{\text{position}}(p, \tau) + \mathcal{L}_{\text{attr}}(p, \tau) + \mathcal{L}_{\text{motion}}(p, \tau). \quad (1)$$

In this section, we provide details of each loss function. We first pair each agent $\hat{a}_i$ in p with a ground-truth agent a_i in τ based on the sequential ordering of the structured agent representation z^a. Assume there are in total N agents in the scenario.

For $\mathcal{L}_{\text{position}}$, we use cross-entropy loss between the per-lane categorical output $\hat{p}$ and the ground-truth lane segment id l. Specifically, we compute it as

$$\mathcal{L}_{\text{position}}(p, \tau) = \sum_{i=1}^{N} -\log \hat{p}_i(l_i), \tag{2}$$

where l_i is the index of the lane segment that the i-th ground-truth agent a_i is on.

For $\mathcal{L}_{\text{attr}}$, we use a negative log-likelihood loss, computed using the predicted GMM on the ground-truth attribute values. Recall that for each attribute of agent i, we use an MLP to predict the parameters of a GMM model $[\mu_i, \Sigma_i, \pi_i]$. Here, we use these parameters to construct a GMM model and compute the likelihood of ground-truth attribute values. Specifically, we have

$$\begin{aligned}\mathcal{L}_{\text{attr}}(p, \tau) = \sum_{i=1}^{N}(&-\log \text{GMM}_{\text{heading,i}}(h_i) - \log \text{GMM}_{\text{vel,i}}(vel_i) \\ &- \log \text{GMM}_{\text{size,i}}(bbox_i) - \log \text{GMM}_{\text{pos,i}}(pos_i)),\end{aligned} \tag{3}$$

where $\text{GMM}_{\text{heading,i}}, \text{GMM}_{\text{vel,i}}, \text{GMM}_{\text{size,i}}, \text{GMM}_{\text{pos,i}}$ represent the likelihood function of the predicted GMM models of agent i's heading, velocity, size and position shift. These likelihood values are computed using the predicted GMM parameters. Meanwhile, h_i, vel_i, $bbox_i$ and pos_i represent the heading, velocity, size and position shift of the ground-truth agent a_i respectively.

For $\mathcal{L}_{\text{motion}}$, we use MSE loss for the predicted trajectory closest to the ground-truth trajectory following the multi-path motion prediction idea [2]. Recall that for each agent $\hat{a}_i$, we predict K' different future trajectories and their probabilities as $\{\text{pos}_{i,k}^{2:T}, \text{prob}_{i,k}\}_{k=1}^{K'} = \text{MLP}(q_i^*)$. For each timestamp t, $\text{pos}_{i,k}^t$ contains the agent's position and heading. We assume the trajectory of ground-truth agent a_i is $\text{pos}_i^{*2:T}$. We can compute the index k^* of the closest trajectory from the K' predictions as $k^* = \arg\min_k \sum_{t=2}^{T}(\text{pos}_{i,k}^t - \text{pos}_i^{*t})^2$. Then, we compute the motion loss for agent i as:

$$\mathcal{L}_{\text{motion},i} = -\log \text{prob}_{i,k^*} + \sum_{t=2}^{T}(\text{pos}_{i,k^*}^t - \text{pos}_i^{*t})^2, \tag{4}$$

where we encourage the model to have a higher probability for the cloest trajectory k^* and reduce the distance between this trajectory with the ground truth. The full motion loss is simply:

$$\mathcal{L}_{\text{motion}}(p, \tau) = \sum_{i}^{N} \mathcal{L}_{\text{motion},i} \tag{5}$$

where we sum over all the motion losses for each predicted agent in p.

C Experiment Details

C.1 Baseline implementation

TrafficGen [3]. We use the official implementation[1]. For a fair comparison, we train its Initialization and Trajectory Generation modules on our dataset for 100 epochs with batch size 64. We modify $T = 50$ in the Trajectory Generation to align with our setting. We use the default values for all the other hyper-parameters. During inference, we enforce TrafficGen to generate N vehicles by using the result of the first N autoregressive steps of the Initialization module.

[1] https://github.com/metadriverse/trafficgen

MotionCLIP [4]. The core idea of MotionCLIP is to learn a shared space for the interested modality embedding (traffic scenario in our case) and text embedding. Formally, this model contains a scenario encoder E, a text encoder $\hat{E}$, and a scenario decoder D. For each example of scene-text paired data (τ, L, m), we encode scenario and text separately with their encoders $\mathbf{z} = E(\tau)$, $\hat{\mathbf{z}} = \hat{E}(L)$. Then, the decoder takes $\mathbf{z}$ and m and output a scenario $p = D(\mathbf{z}, m)$. MotionCLIP trains the network with $\mathcal{L}_{\text{rec}}$ to reconstruct the scenario from the latent code:

$$\mathcal{L}_{\text{rec}} = \mathcal{L}_{\text{position}}(p, \tau) + \mathcal{L}_{\text{attr}}(p, \tau) + \mathcal{L}_{\text{motion}}(p, \tau), \tag{6}$$

where we use the same set of loss functions as ours (Equation 1). On the other hand, MotionCLIP aligns the embedding space of the scenario and text with:

$$\mathcal{L}_{\text{align}} = 1 - \cos(\mathbf{z}, \hat{\mathbf{z}}), \tag{7}$$

which encourages the alignment of scenario embedding $\mathbf{z}$ and text embedding $\hat{\mathbf{z}}$. The final loss function is therefore

$$\mathcal{L} = \mathcal{L}_{\text{rec}} + \lambda \mathcal{L}_{\text{align}}, \tag{8}$$

where we set $\lambda = 100$.

During inference, given an input text L and a map m, we can directly use the text encoder to obtain latent code and decode a scenario from it, formally $\tau = D(\hat{E}(L), m)$.

For the scenario encoder E, we use the same scenario encoder as in [3], which is a 5-layer *multi-context gating* (MCG) block [2] to encode the scene input τ and outputs $\mathbf{z} \in \mathbb{R}^{1024}$ with the context vector output c of the final MCG block. For text encoder $\hat{E}$, we use the sentence embedding of the fixed GPT-2 model. For the scenario decoder D, we modify our `Generator` to take in latent representation $\mathbf{z}$ with a dimension of 1024 instead of our own structured representation. Because D does not receive the number of agents as input, we modify `Generator` to produce the $N = 32$ agents for every input and additionally add an MLP decoder to predict the objectiveness score of each output agent. Here objectiveness score is a binary probability score indicating whether we should put each predicted agent onto the final scenario or not. During training, for computation of $\mathcal{L}_{\text{rec}}$, we use Hungarian algorithm to pair ground-truth agents with the predicted ones. We then supervise the objectiveness score in a similar way as in DETR.

Note that we need text-scenario paired data to train MotionCLIP. To this end, we use a rule-based method to convert a real dataset τ to a text L. This is done by describing different attributes of the scenario with language. Similar to our Attribute Description dataset, in each text, we enumerate the scenario properties 1) sparsity; 2) position; 3) speed and 4) ego vehicle's motion. Here is one example: "the scene is very dense; there exist cars on the front left of ego car; there is no car on the back left of ego car; there is no car on the back right of ego car; there exist cars on the front right of ego car; most cars are moving in fast speed; the ego car stops".

We transform every scenario in our dataset into a text with the format as above. We then train MotionCLIP on our dataset with the same batch size and number of iterations as `LCTGen`.

C.2 Metric

We show how to compute MMD in this section. Specifically, MMD measures the distance between two distributions q and p.

$$\begin{aligned}\text{MMD}^2(p, q) =& \mathbb{E}_{x,x' \sim p}[k(x, x')] + \mathbb{E}_{y,y' \sim q}[k(y, y')] \\ & - 2\mathbb{E}_{x \sim p, y \sim q}[k(x, y)],\end{aligned} \tag{9}$$

where k is the kernel function (a Gaussian kernel in this work). We use Gaussian kernel in this work. For each pair of real and generated data $(\tau, \hat{\tau})$, we compute the distribution difference between them per attribute.

C.3 Dataset

Crash Report. We use 38 cases from the CIREN dataset [5] from the NHTSA crash report search engine. Each case contains a long text description of the scenario as well as a PDF diagram showing the scenario. Because the texts are very long and require a long time for humans to comprehend, in our human study, along with each text input, we will also show the diagram of the scenario as a reference. We show example crash reports in Section D.3. We also refer the reader to the NHTSA website [2] to view some examples of the crash report.

Attribute Description. We create text descriptions that highlight various attributes of a traffic scenario. Specifically, we use the following attributes and values:

1. Sparsity: "the scenario is {nearly empty/sparse/with medium density/very dense}".
2. Position: "there are only vehicles on the {left/right/front/back} side(s) of the center car" or "there are vehicles on different sides of the center car".
3. Speed: "most cars are moving in {slow/medium/fast} speed" or "most cars are stopping".
4. Ego-vehicle motion: "the center car {stops/moves straight/turns left/turns right}".

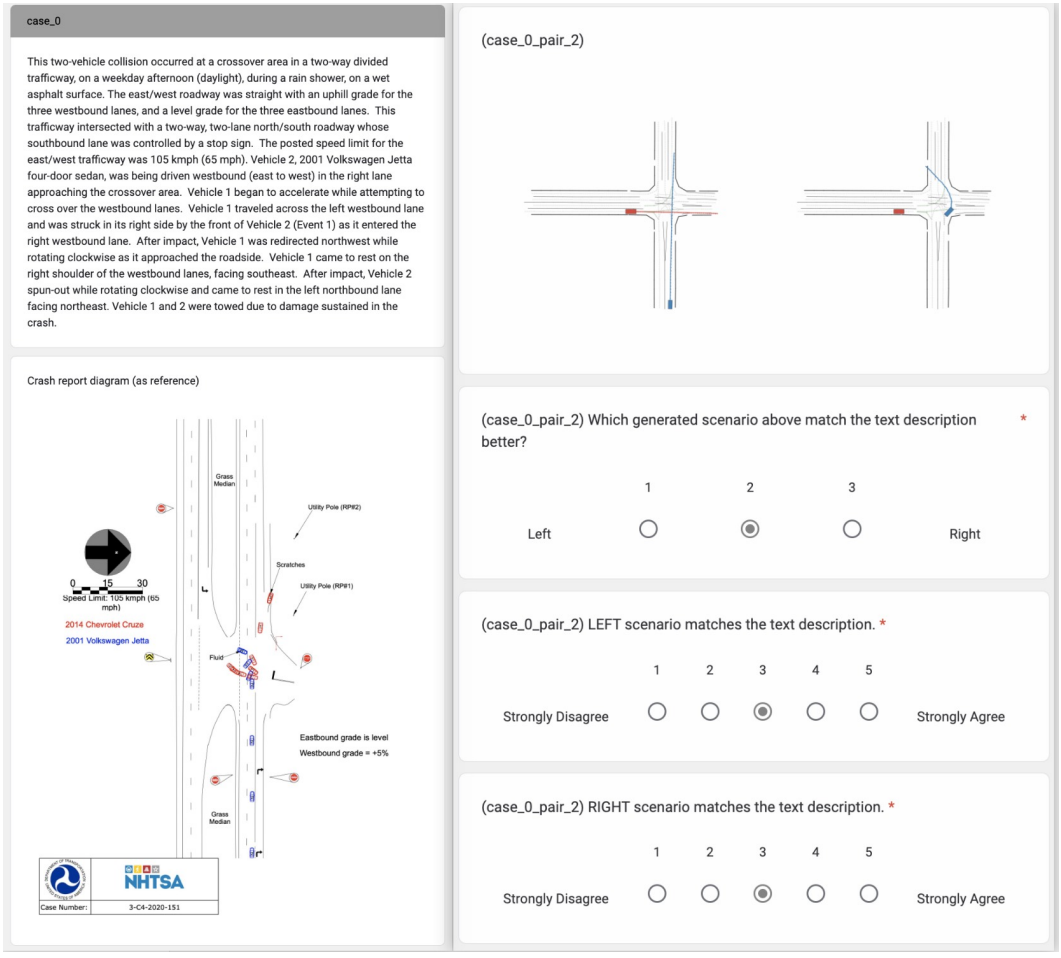

Figure A1: Human study user interface.

We create sentences describing each of the single attributes with all the possible values. We also compose more complex sentences by combining 2,3 or 4 attributes together with random values for

[2] https://crashviewer.nhtsa.dot.gov/CIREN/Details?Study=CIREN&CaseId=11

each of them. In total, we created 40 cases for human evaluation. Please refer to Section D.3 for some example input texts from this dataset.

C.4 Human study

We conduct the human study to access how well the generated scenario matches the input text. We showcase the user interface of our human study in Figure A1. We compose the output of two models with the same text input in random order and ask the human evaluator to judge which one matches the text description better. Then, we also ask them to give each output a 1-5 score. We allow the user to select "unsure" for the first question.

We invite 12 human evaluators for this study, and each of them evaluated all the 78 cases we provided. We ensure the human evaluators do not have prior knowledge of how different model works on these two datasets. On average, the human study takes about 80 minutes for each evaluator.

C.5 Qualitative result full texts

In Figure 4 of our main paper and Figure A2, we show 5 examples of the output of our model on Crash Report data on the first row. Recall that the texts we show in the figures are the summary from our `Interpreter` due to space limitations. We show the full input text for each example in this section.

Text 1: Full texts of examples in Figure 4 .

```
Figure 4 Column 1 (CIREN ID 594):
"This crash occurred during daylight hours on a dry, bituminous divided trafficway (median
    strip without positive barrier) under clear skies.   There were four east travel lanes
    (two through lanes, one left turn and one right turn) and four west travel lanes (two
    through lanes, one left and one right).   The east lanes have a slight right curve and
    the west lanes curve slightly to the left.  Both east/west travel lanes were level
    grade at point of impact and divided by a grass median.   The speed limit at this
    location is 80km/h (50 mph).  The intersecting north/south roadway consisted of one
    north travel lane and three south travel lanes (one through lanes, one left and one
    right).  These travel lanes were divided by a raised concrete median on the northern
    side of the intersection.  This intersection is controlled by overhead traffic signals.
     A 2017 Dodge Grand Caravan (V1) was traveling east in the left turn lane and a 2006
    Nissan Sentra (V2) was traveling west in the left through lane.  As V1 was traveling
    east it attempted to execute a left turn to travel north when its front bumper impacted
     the front bumper of V2 (Event 1).   After initial impact, V1 rotated counterclockwise
    approximately 80 degrees before traveling to its final resting position in the middle
    of the intersection facing north.  V2 was traveling west in the left through lane and
    attempting to travel through the intersection when its front bumper impacted the front
    bumper of V1.   After initial impact V2 rotated clockwise approximately 20 degrees
    before traveling to its final resting position in the middle of the intersection facing
     northwest.  V1 and V2 were towed from the scene due to damage sustained in the crash."

Figure 4 Column 2 (CIREN ID 31):
"A 2016 Kia Sedona minivan (V1) was traveling southwest in the right lane of three.  A 2015
    Chevrolet Silverado cab chassis pickup (V2) was ahead of V1 in the right lane.  V2 was
    a working vehicle picking up debris on the highway in a construction zone.  The driver
    of V2 stopped his vehicle in the travel lane.   The driver of V1 recognized an
    impending collision and applied the brakes while steering left in the last moment
    before impact.  V1 slid approximately three meters before the front of V1 struck the
    back plane of V2 in a rear-end collision with full engagement across the striking
    planes (Event 1).  Both vehicles came to rest approximately two meters from impact.  V1
     was towed due to damage while V2 continued in service."

Figure A2 Row 1 Column 1 (CIREN ID 77):
"A 2017 Chevrolet Malibu LS sedan (V1) was traveling southeast in the right lane cresting a
    hill. A 1992 Chevrolet C1500 pickup (V2) was traveling northwest in the second lane
    cresting the same hill. Vehicle 2 crossed left across the center turn lane, an oncoming
     lane, and then into V1\u2019s oncoming lane of travel. Vehicle 1 and Vehicle 2
    collided in a head-on, offset-frontal configuration (Event 1). Vehicle 1 attempted to
    steer left just before impact, focusing the damage to the middle-right of its front
    plane. Both vehicles rotated a few degrees clockwise before coming to rest in the
    roadway, where they were towed from the scene due to damage."
```

```
Figure A2 Row 1 Column 2 (CIREN ID 33):
"This two-vehicle collision occurred during the pre-dawn hours (dark, street lights present)
     of a fall weekday at the intersection of two urban roadways. The crash only involved
    the eastern leg of the intersection. The westbound lanes of the eastern leg consisted
    of four westbound lanes that included a right turn lane, two through lanes, and a left
    turn lane.  The three eastbound lanes of the eastern leg consisted of a merge lane from
     the intersecting road and two through-lanes. The roadway was straight with a speed
    limit of 89 kmph (55 mph), and the intersection was controlled by overhead, standard
    electric, tri-colored traffic signals. At the time of the crash, the weather was clear
    and the roadway surfaces were dry. As Vehicle 1 approached the intersection, its driver
     did not notice the vehicles stopped ahead at the traffic light. The traffic signal
    turned green and Vehicle 2 began to slowly move forward. The frontal plane of Vehicle 1
     struck the rear plane of Vehicle 2 (Event 1). Both vehicles came to rest in the left
    through-lane of the westbound lane facing in a westerly direction. Vehicle 1 was towed
    from the scene due to damage sustained in the crash. Vehicle 2 was not towed nor
    disabled. The driver of Vehicle 2 was transported by land to a local trauma center and
    was treated and released."


Figure A2 Row 1 Column 3 (CIREN ID 56):
"A 2013 Honda CR-V utility vehicle (V1) was traveling west in the right lane approaching an
    intersection. A 2003 Chevrolet Silverado 1500 pickup (V2) was stopped facing north at a
     stop sign.   Vehicle 2 proceeded north across the intersection and was struck on the
    right plane by the front plane of V1 (Event 1). The impact caused both vehicles to
    travel off the northwest corner of the intersection, where they came to rest. Both
    vehicles were towed due to damage.""
```

D Additional Results

D.1 Controllable Self-driving Policy Evaluation

We show how LCTGen can be utilized to generate interesting scenarios for controllable self-driving policy evaluation. Specifically, we leverage LCTGen to generate traffic scenario datasets possessing diverse properties, which we then use to assess self-driving policies under various situations. For this purpose, we input different text types into LCTGen: 1) Crash Report, the real-world crash report data from CIREN; 2) Traffic density specification, a text that describes the scenario as "sparse", "medium dense", or "very dense". For each type of text, we generate 500 traffic scenarios for testing. Additionally, we use 500 real-world scenarios from the Waymo Open dataset.

We import all these scenarios into an interactive driving simulation, MetaDrive [6]. We evaluate the performance of the IDM [7] policy and a PPO policy provided in MetaDrive. In each scenario, the self-driving policy replaces the ego-vehicle in the scenario and aims to reach the original end-point of the ego vehicle, while all other agents follow the trajectory set out in the original scenario. We show the success rate and collision rate of both policies in Table A1. Note that both policies experience significant challenges with the Crash Report scenarios, indicating that these scenarios present complex situations for driving policies. Furthermore, both policies exhibit decreased performance in denser traffic scenarios, which involve more intricate vehicle interactions. These observations give better insight about the drawbacks of each self-driving policy. This experiment showcases LCTGen as a valuable tool for generating traffic scenarios with varying high-level properties, enabling a more controlled evaluation of self-driving policies.

Test Data	IDM [7]		PPO (MetaDrive) [6]	
	Success (%)	Collision (%)	Success (%)	Collision (%)
Real	93.60	3.80	69.32	14.67
LCTGen + Crash Report [5]	52.35	39.89	25.78	27.98
LCTGen + "Sparse"	91.03	8.21	41.03	21.06
LCTGen + "Medium"	84.47	12.36	43.50	26.67
LCTGen + "Dense"	68.12	19.26	38.89	32.41

Table A1: Controllable self-driving policy evaluation.

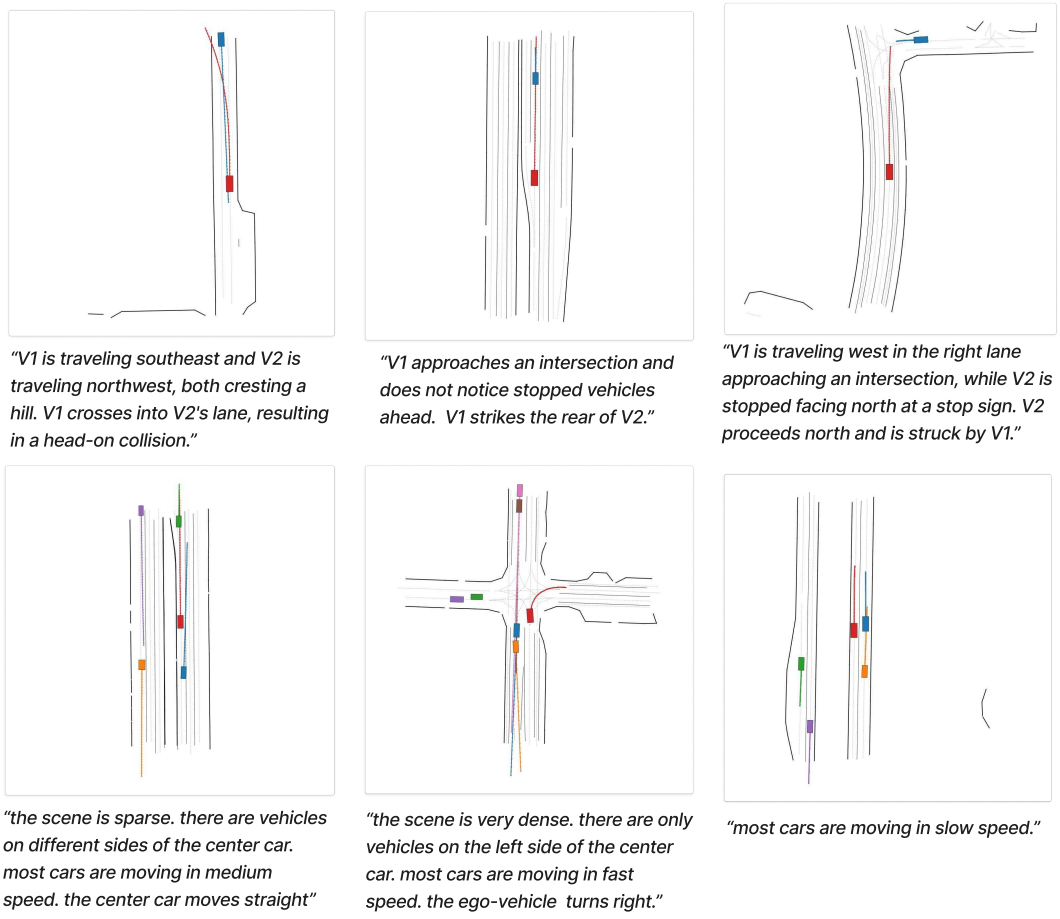

Figure A2: Qualitative results on text-conditioned generation.

D.2 Text-conditioned Simulation Qualitative Results

We show the more qualitative results of text-conditioned simulation in Figure A2. Here, the upper 3 examples are from the Crash Report dataset, the lower 3 examples are from the Attribute Description dataset.

D.3 `Interpreter` input-output examples

Here we show the full-text input and output of `Interpreter` for four examples in Figure 4. Specifically, we show two examples from Crash Report and two examples from Attribute Descriptions.

Text 2: Input-output examples of `Interpreter`.

```
Figure 4 Column 1 (CIREN ID 594):
Input:
"This crash occurred during daylight hours on a dry, bituminous divided trafficway (median
    strip without positive barrier) under clear skies.  There were four east travel lanes
    (two through lanes, one left turn and one right turn) and four west travel lanes (two
    through lanes, one left and one right).  The east lanes have a slight right curve and
    the west lanes curve slightly to the left.  Both east/west travel lanes were level
    grade at point of impact and divided by a grass median.  The speed limit at this
    location is 80km/h (50 mph).  The intersecting north/south roadway consisted of one
    north travel lane and three south travel lanes (one through lanes, one left and one
    right).  These travel lanes were divided by a raised concrete median on the northern
    side of the intersection.  This intersection is controlled by overhead traffic signals.
     A 2017 Dodge Grand Caravan (V1) was traveling east in the left turn lane and a 2006
    Nissan Sentra (V2) was traveling west in the left through lane.  As V1 was traveling
```

```
     east it attempted to execute a left turn to travel north when its front bumper impacted
      the front bumper of V2 (Event 1).   After initial impact, V1 rotated counterclockwise
     approximately 80 degrees before traveling to its final resting position in the middle
     of the intersection facing north.  V2 was traveling west in the left through lane and
     attempting to travel through the intersection when its front bumper impacted the front
     bumper of V1.   After initial impact V2 rotated clockwise approximately 20 degrees
     before traveling to its final resting position in the middle of the intersection facing
      northwest.  V1 and V2 were towed from the scene due to damage sustained in the crash."

Output:
Summary: V1 is traveling east in the left turn lane and attempts to turn left when it
     collides with V2 traveling west in the left through lane. There are 2 vehicles in this
     scenario. This happens at the intersection of an eight-lane divided trafficway and a
     four-lane north/south roadway, controlled by traffic signals.
Explanation:
- V1 (ego vehicle) is traveling east in the left turn lane and attempts to turn left. We
     cannot find V1's speed in the query. Because V1 tries to turn left, its initial speed
     should be set low. We set V1's speed as 5 m/s, which has the index of 2. V1 turns left,
      so its actions are all 1 (turn left).
- V2 is traveling west in the left through lane. As V1 is turning left, 5 seconds before the
      crash, V2 is coming from the opposite direction (westbound), crossing the path of V1.
     In the coordinates of V1 (which is facing east initially), V2 comes from the front and
     is on the left side. Hence, V2's position is \"front left\" (0). As V1 is facing east
     and V2 facing west, V2 is moving in the parallel opposite direction with V1. Therefore
     its direction is 1 (parallel_opposite). We cannot find V2's speed in the query. Because
      V2 is traveling west and hit by V1 5 seconds later, we assume V2's init speed is 8 m/s
      (index 3). Given this speed, V2's distance to V1 is 8m/s * 5s = 40m (index 8). V2
     keeps going straight, so its actions are all 4 (keep speed).
- Map: This happens at the intersection of an eight-lane divided trafficway (4 lanes for
     eastbound and westbound traffic) and a four-lane north/south roadway (1 lane for
     northbound and 3 lanes for southbound traffic). The ego vehicle is in the left turn-
     only lane with lane id 4. There is an intersection with a four-lane north/south road
     ahead. V1 is about to cross the interaction, therefore the distance to the intersection
      is with 15 meter index 3.
Actor Vector:
- 'V1': [-1, 0, 0, 2, 1, 1, 1, 1]
- 'V2': [0, 8, 1, 3, 4, 4, 4, 4]
Map Vector:
- 'Map': [4, 4, 1, 3, 3, 4]

Figure A2 Row 1 Column 2 (CIREN ID 33):
Input:
"This two-vehicle collision occurred during the pre-dawn hours (dark, street lights present)
      of a fall weekday at the intersection of two urban roadways. The crash only involved
     the eastern leg of the intersection. The westbound lanes of the eastern leg consisted
     of four westbound lanes that included a right turn lane, two through lanes, and a left
     turn lane.  The three eastbound lanes of the eastern leg consisted of a merge lane from
      the intersecting road and two through-lanes. The roadway was straight with a speed
     limit of 89 kmph (55 mph), and the intersection was controlled by overhead, standard
     electric, tri-colored traffic signals. At the time of the crash, the weather was clear
     and the roadway surfaces were dry. As Vehicle 1 approached the intersection, its driver
      did not notice the vehicles stopped ahead at the traffic light. The traffic signal
     turned green and Vehicle 2 began to slowly move forward. The frontal plane of Vehicle 1
      struck the rear plane of Vehicle 2 (Event 1). Both vehicles came to rest in the left
     through-lane of the westbound lane facing in a westerly direction. Vehicle 1 was towed
     from the scene due to damage sustained in the crash. Vehicle 2 was not towed nor
     disabled. The driver of Vehicle 2 was transported by land to a local trauma center and
     was treated and released."

Output:
Summary: V1 approaches an intersection and does not notice stopped vehicles ahead. The
     traffic light turns green, and V2 starts moving forward slowly. V1 strikes the rear of
     V2. There are 2 vehicles in this scenario. This happens at the intersection of two
     urban roadways with multiple lanes and traffic signals.
Explanation:
- V1 (ego vehicle) is approaching the intersection and does not notice the stopped vehicles
     ahead. We cannot find V1's speed in the query. Because V1 is moving and does not stop,
     we assume V1's initial speed is 10 m/s (index 4). V1 keeps going straight, so its
     actions are all 4 (keep speed).
- V2 is stopped at the traffic light and starts moving forward slowly when the light turns
     green. As V1 is moving straight and hits V2 from behind, 5 seconds before the crash, V2
      is in front of V1. Hence, V2's position is \"front right\" (3). As V1 and V2 are both
```

```
    moving westbound, V2 is moving in the parallel same direction with V1. Therefore its
    direction is 0 (parallel_same). V2's speed is slow, so we set it to 2.5 m/s (index 1).
    Given this speed, V2's distance to V1 is 10m/s * 5s - 2.5m/s * 5s = 37.5m (index 7). V2
     keeps going straight, so its actions are all 4 (keep speed).
- Map: This happens at the intersection of two urban roadways with multiple lanes and
    traffic signals. The ego vehicle is in the left through-lane of the westbound lane with
     lane id 3. There are 4 westbound lanes and 3 eastbound lanes. The distance of the ego
    vehicle to the intersection is set to be 10m/s * 5s = 50m (index 10).
Actor Vector:
- 'V1': [-1, 0, 0, 4, 4, 4, 4, 4]
- 'V2': [3, 7, 0, 1, 4, 4, 4, 4]
Map Vector:
- 'Map': [4, 3, 0, 0, 10, 3]


Figure A2 Row 2 Column 1:
Input:
"the scene is with sparse density. there are vehicles on different sides of the center car.
    most cars are moving in medium speed. the center car moves straight"

Output:
Summary: The ego vehicle (V1) moves straight at a medium speed in a sparse traffic scenario.
     There are vehicles on different sides of the ego vehicle, corresponding to the sparse
    density. We generate 4 vehicles in the scene on a six-lane two-way road. The ego car is
     placed on the third lane from the right.
Explanation:
- V1 (ego vehicle) is in the center and moves straight. Because V1 moves with a medium speed
    , its speed is set to 4 (medium speed) and action indices are 4 (keep speed).
- V2 is in the closest distance range (index 0). It is moving in the same direction as the
    ego vehicle (index 0) and is located in the back right (index 2) of the ego vehicle. It
     is moving with a medium speed (index 4), and its actions are all to keep speed (index
    4).
- V3 is in a further distance range (index 1). It is moving in the opposite direction of the
     ego vehicle (index 1), therefore it is on the left side of the ego vehicle, locating
    in the back left (index 1) of the ego vehicle. It is moving with a medium speed (index
    4), and its actions are all to keep speed (index 4).
- V4 is in the farthest distance range (index 2). It is moving in the same direction as the
    ego vehicle (index 0) and is located in the front right (index 3) of the ego vehicle.
    It is moving with a medium speed (index 4), and its actions are all to keep speed (
    index 4).
- V5 is in the farthest distance range (index 2). It is moving in the opposite direction of
    the ego vehicle (index 1), therefore it is on the left side of the ego vehicle,
    locating in the front left (index 0) of the ego vehicle. It is moving with a medium
    speed (index 4), and its actions are all to keep speed (index 4).
- Map: Given the sparse density and the presence of vehicles on different sides of the ego
    vehicle, we assume this happens on a 6-lane two-way road. The ego vehicle is in the
    third lane from the right, with lane id 3. There is no intersection nearby, indicated
    by -1.
Actor Vector:
- 'V1': [-1, 0, 0, 4, 4, 4, 4, 4]
- 'V2': [2, 0, 0, 4, 4, 4, 4, 4]
- 'V3': [1, 1, 1, 4, 4, 4, 4, 4]
- 'V4': [3, 2, 0, 4, 4, 4, 4, 4]
- 'V5': [0, 2, 1, 4, 4, 4, 4, 4]
Map Vector:
- 'Map': [3, 3, 0, 0, -1, 3]

Figure 4 Column 3:
Input:
"the scene is with sparse density. there are only vehicles on the back side of the center
    car. most cars are moving in fast speed. the center car turns left"

Output:

Summary: The ego vehicle (V1) turns left at a fast speed in a sparse traffic scenario. All
    other vehicles are located behind the ego vehicle, which corresponds to the scenario's
    sparse density. We generate 3 vehicles in the scene on a six-lane two-way road with an
    intersection in front. The ego car is placed on the left-turn lane.
Explanation:
- V1 (ego vehicle) is in the center and turns left. Because V1 moves with a fast speed, its
    speed is set to 6 (fast speed) and action indices are 1 (turn left).
```

```
- V2-V3 are in the closest distance range (index 0). They are moving in the same direction
    as the ego vehicle (index 0). Because the ego car is on the left-turn lane, they are
    both located in the back right (index 2) of the ego vehicle. They are both moving with
    a fast speed (index 6 or 7), and their actions are all to keep speed (index 4).
- V4 is in a further distance range (index 1). It is moving in the opposite direction of the
     ego vehicle (index 1), therefore it is on the left side of the ego vehicle, locating
    in the back left (index 1) of the ego vehicle. It is moving with a fast speed (index 6)
    , in the opposite direction as the ego vehicle (index 1). Its actions are all to keep
    speed (index 4).
- Map: Given the sparse density and the presence of vehicles behind the ego vehicle, we
    assume this happens on a 6-lane two-way road. The ego vehicle is in the left-turn lane,
     with lane id 3. There is an intersection 10 meters ahead (index 2) as the ego vehicle
    is turning left.
Actor Vector:
- 'V1': [-1, 0, 0, 6, 1, 1, 1, 1]
- 'V2': [2, 0, 0, 6, 4, 4, 4, 4]
- 'V3': [2, 0, 0, 7, 4, 4, 4, 4]
- 'V4': [1, 1, 1, 6, 4, 4, 4, 4]
Map Vector:
- 'Map': [3, 3, 2, 2, 2, 3]
```

D.4 Attribute Description result split

Method	Density	Position	Speed	Ego-car Motion
TrafficGen [3]	2.75	2.03	2.34	2.27
MotionCLIP [4]	1.89	2.24	1.91	1.78
LCTGen	**4.24**	**4.28**	**4.38**	**4.40**

Table A2: Human study result split analysis on Attribute Description scores.

We generate the Attribute Description dataset with different attributes. In this section, we split the matching score result for the full dataset into different attributes. We show the result in Table A2. We observe our method has nearly identical performance over all the attributes. TrafficGen the best results with Density, while MotionCLIP performs the best with Position.

D.5 Ablation study

Method	Initialization				Motion		
	Pos	Heading	Speed	Size	mADE	mFDE	SCR
w/o Quad.	0.092	0.122	0.076	0.124	2.400	4.927	8.087
w/o Dist.	0.071	0.124	0.073	0.121	1.433	3.041	6.362
w/o Ori.	0.067	0.132	0.082	0.122	1.630	3.446	7.300
w/o Speed	0.063	0.120	0.104	0.122	2.611	5.188	7.150
w/o Action	0.067	0.128	0.173	0.128	2.188	5.099	7.146
w/o x_i	0.067	0.133	0.076	0.124	1.864	3.908	**5.929**
w/o GMM	0.064	0.128	0.078	0.178	1.606	3.452	8.216
LCTGen	**0.062**	**0.115**	**0.072**	**0.120**	**1.329**	**2.838**	6.700

Table A3: Ablation study of LCTGen

In our main paper, we split the ablation study into two different groups. Here we show the full results of all the ablated methods in Table A3. We additionally show the effect of 1) using the learnable query x_i and 2) using the GMM prediction for attributes.

D.6 Instructional Traffic Scenario Editing

We show another example of instructional traffic scenario editing in Figure A3. Different from the compound editing in Figure 5 in the main paper, here every example is edited from the input scenario.

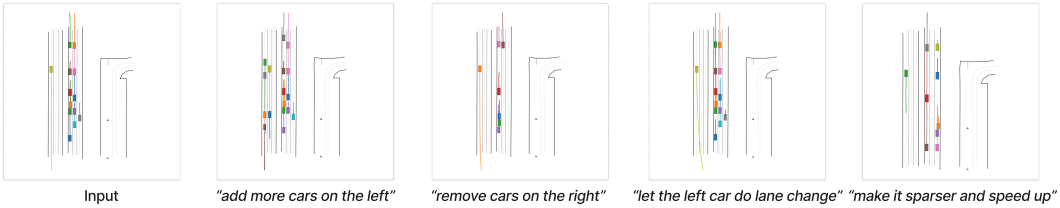

Figure A3: Instructional editing on a real-world scenario

References

[1] J. Wei, X. Wang, D. Schuurmans, M. Bosma, E. Chi, Q. Le, and D. Zhou. Chain of thought prompting elicits reasoning in large language models. *arXiv preprint arXiv:2201.11903*, 2022.

[2] B. Varadarajan, A. Hefny, A. Srivastava, K. S. Refaat, N. Nayakanti, A. Cornman, K. Chen, B. Douillard, C. P. Lam, D. Anguelov, and B. Sapp. Multipath++: Efficient information fusion and trajectory aggregation for behavior prediction, 2021.

[3] L. Feng, Q. Li, Z. Peng, S. Tan, and B. Zhou. Trafficgen: Learning to generate diverse and realistic traffic scenarios. *arXiv preprint arXiv:2210.06609*, 2022.

[4] G. Tevet, B. Gordon, A. Hertz, A. H. Bermano, and D. Cohen-Or. Motionclip: Exposing human motion generation to clip space. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXII*, pages 358–374. Springer, 2022.

[5] N. H. T. S. Administration. Crash injury research engineering network (ciren). https://crashviewer.nhtsa.dot.gov/CIREN/SearchIndex, 2016.

[6] Q. Li, Z. Peng, L. Feng, Q. Zhang, Z. Xue, and B. Zhou. Metadrive: Composing diverse driving scenarios for generalizable reinforcement learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022.

[7] M. Treiber, A. Hennecke, and D. Helbing. Congested traffic states in empirical observations and microscopic simulations. *Physical Review E*, 62(2):1805–1824, aug 2000. doi:10.1103/physreve.62.1805. URL https://doi.org/10.1103%2Fphysreve.62.1805.