

We include additional technical details and results supplementing the main paper due to page limits. Appendix A gives details on the model architecture and trainings, Appendix B gives details for experiments, and Appendix C and D provide additional results to supplement those in the main paper. In Appendix E, we discuss the broader impacts of our methodology.

A Model Details

A.1 BMT Architecture

Preprocessing. The model takes real-world scenarios in the ScenarioNet format [8]. It performs a series of preprocessing steps on both static map features and dynamic agent trajectories. We compute the global bounds to extract a consistent map center and heading. Each map feature is decomposed into a sequence of vectorized segments, where each vector is represented by start and end points in 3D coordinates. These vectors are augmented with relative directional positions, segment headings, and lengths. Semantic attributes are binary indicators that encode feature types. Map feature types (e.g., lanes, crosswalks, lane markings, stop signs, etc.) are encoded as semantic indicators. Features are then centered at the map center. Traffic-light states are also encoded and aligned with timesteps. Similarly, agent trajectories are centered for agent-feature encoding, and position, heading, velocity, shape, and type are extracted over time to form temporal sequences. We extract a 16-dimensional state vector at each timestep. This preprocessing ensures that all trajectories are represented in a consistent spatial frame and expressed in an egocentric coordinate system.

BMT Motion Tokenization. We formulate motion prediction as an autoregressive next-token prediction problem, where each motion token corresponds to a discretized control input over a fixed time interval. The tokenization process maps continuous agent motion—longitudinal acceleration and yaw rate—into discrete 2D bins. During tokenization, we consider candidate tokens sampled from a fixed grid; for each candidate, we simulate the resulting motion over a short duration and select the best-matching action by minimizing the contour-alignment error between the predicted agent footprint and the ground-truth pose (position and heading). We adopt a simplified bicycle model to parameterize agent motion using longitudinal acceleration and yaw rate within predefined bounds: acceleration $\in [-10, 10] \text{ m/s}^2$ and yaw rate $\in [-\pi/2, \pi/2] \text{ rad/s}$. Given predicted motion-token sequences, the trajectory is reconstructed by mapping tokens back to acceleration and yaw-rate changes.

In both forward and reverse directions, the same tokenizer is used to decode motion tokens into continuous trajectories. Given an initial state $\tau_t = (x_t, y_t, \theta_t, v_t)$, forward decoding simulates the next state τ_{t+1} by applying a tokenized control action $z_t = (a_t, \delta_t) \in \mathcal{A}$, where \mathcal{A} is the discrete token space; repeating this autoregressively over the predicted tokens reconstructs the full trajectory $\tau_{t:t+T}$. For reverse decoding, given a known future state τ_{t+1} , the model evaluates candidate tokens $z_t \in \mathcal{A}$, simulates their inverse dynamics with $\Delta t \rightarrow -\Delta t$, and selects the token that best reconstructs the preceding state τ_t . Operating in both directions is a key distinction of our approach: forward prediction enables open-loop rollouts of future behaviors, whereas reverse prediction traces from a desired outcome (e.g., a collision state) back to plausible initiating actions.

BMT Motion Decoder. The decoder follows a GPT-like structure composed of stacked cross-attention blocks. Each block integrates three structured attention modules: agent-to-agent (A2A), agent-to-temporal (A2T), and agent-to-scene (A2S). Relational information across modalities is encoded via relation embeddings: agent-to-agent relation embedding, agent-to-time relation embedding, and agent-to-scene relation embedding, each capturing context-specific spatial/temporal cues. For token construction, we use several embeddings, including the agent-type embedding, agent-shape embedding, agent-ID embedding, and motion-token embedding (which embeds discrete control tokens). A continuous-motion feature embedding encodes acceleration and yaw-rate attributes. Auxiliary embeddings include the special-token embedding (to indicate sequence boundaries) and the reverse-prediction indicator embedding (to distinguish forward from reverse prediction modes). For each attention edge, a relative-relation embedding is computed using a Fourier encoder [20] and added to the key and/or value vectors. The input agent tokens are progressively updated across layers by aggregating contextual features from other agents, their temporal histories, and relevant map elements.

A.2 BMT Motion Decoding

Input. Specifically, each input agent token to the BMT motion decoder is constructed by summing of embeddings of:

- the motion token from the previous step (representing discretized acceleration and yaw rate),
- agent shape (length, width, height),
- agent type (e.g., vehicle, pedestrian),
- agent identifier (embedded optionally),
- special token type (e.g., <start>, <end>, or padding),

as well as a continuous motion delta feature embedded via a Fourier encoder. These components are projected into the same hidden dimension and summed to form the input motion token embedding.

Prediction Head and Output. After processing through all decoding layers, the final hidden state for each valid token is passed through a two-layer MLP head to produce logits over the motion token space:

$$\text{MLP}(\mathbf{h}) = W_2 \cdot \phi(W_1 \cdot \mathbf{h}) \in \mathbb{R}^{K^2},$$

where $\phi(\cdot)$ denotes the GELU activation and K^2 is the number of discrete motion tokens (from a $K \times K$ acceleration–yaw bin grid). The resulting logits are used to predict the next motion token at each time step. During inference, we generate motion tokens using nucleus (top- p) sampling.

Trajectory Reconstruction Our model makes a prediction in the interval of 0.5 seconds. To simulate the effect of a motion token over a fixed time step $\Delta t = 0.5$ s, we adopt midpoint integration based on a simplified bicycle model. In forward predictions, given a current state $\mathbf{s}_t = (x_t, y_t, \theta_t, v_t)$, the model computes the next speed and heading as $v_{t+1} = v_t + a \cdot \Delta t$ and $\theta_{t+1} = \theta_t + \omega \cdot \Delta t$. The average speed and heading are then defined as $\bar{v} = \frac{v_t + v_{t+1}}{2}$ and $\bar{\theta} = \left(\frac{\theta_t + \theta_{t+1}}{2}\right)$. In the reverse direction, the process is inverted. Given a future state \mathbf{s}_{t+1} , the model enumerates all possible token candidates and inverts the dynamics: $v_t = v_{t+1} - a \cdot \Delta t$ and $\theta_t = \theta_{t+1} - \omega \cdot \Delta t$. The average quantities \bar{v} and $\bar{\theta}$ are computed similarly and used to derive the previous position:

$$x_t = x_{t+1} - \bar{v} \cdot \cos(\bar{\theta}) \cdot \Delta t, \quad y_t = y_{t+1} - \bar{v} \cdot \sin(\bar{\theta}) \cdot \Delta t.$$

A.3 BMT Training

Training Loss. The decoder produces a logit tensor $\hat{\mathbf{z}} \in \mathbb{R}^{B \times T \times N \times |\mathcal{A}|}$, where $|\mathcal{A}|$ is the number of motion tokens (i.e., discretized acceleration–yaw pairs). The supervision target is the ground-truth token sequence $\mathbf{z}^* \in \mathbb{N}^{B \times T \times N}$, derived by tokenizing agent trajectories. A binary mask $\mathbf{m} \in \{0, 1\}^{B \times T \times N}$ specifies which tokens are valid and should contribute to the training loss. The training objective is computed over all valid entries using the cross-entropy loss:

$$\mathcal{L}_{\text{main}} = \frac{1}{\sum_{b,t,n} m_{b,t,n}} \sum_{b,t,n} m_{b,t,n} \cdot \text{CE}(\hat{z}_{b,t,n}, z_{b,t,n}^*),$$

where CE denotes the standard cross-entropy loss between the predicted logits and the ground-truth discrete token.

Reverse Prediction. During reverse prediction, the model measures metrics separately for forward and reverse token predictions. Let $\mathbf{b} \in \{0, 1\}^{B \times T \times N}$ be a binary indicator for whether each token comes from reverse prediction. Then we compute separate metrics:

$$\begin{aligned} \text{Accuracy}^{\text{reverse}} &= \frac{\sum m_{b,t,n} \cdot b_{b,t,n} \cdot \mathbf{1}[\hat{z}_{b,t,n} = z_{b,t,n}^*]}{\sum m_{b,t,n} \cdot b_{b,t,n}}, \\ \text{Entropy}^{\text{reverse}} &= \frac{1}{\sum m \cdot b} \sum m_{b,t,n} \cdot b_{b,t,n} \cdot \mathcal{H}(\hat{z}_{b,t,n}), \end{aligned}$$

with analogous expressions for forward prediction (i.e., for $1 - b_{b,t,n}$).

Table 6: BMT Model Parameters.

| Component | Parameters | Size (MB) |
|--|------------|-----------|
| Scene Encoder | 902,080 | 3.44 |
| Map Polyline Encoder | 22,656 | 0.09 |
| Traffic Light Embedding MLP | 1,024 | 0.00 |
| Scene Relation Embedding | 117,184 | 0.45 |
| Scene Transformer Encoder | 744,448 | 2.84 |
| Scene Encoder Output Projection | 16,512 | 0.06 |
| Scene Output Pre-Normalization | 256 | 0.00 |
| Motion Decoder | 4,385,025 | 16.73 |
| Multi-Cross Attention Decoder | 2,881,536 | 10.99 |
| Motion Prediction Head | 157,121 | 0.60 |
| Motion Prediction Pre-Normalization | 256 | 0.00 |
| Agent-to-Agent Relation Embedding | 418,432 | 1.60 |
| Agent-to-Time Relation Embedding | 418,432 | 1.60 |
| Agent-to-Scene Relation Embedding | 117,184 | 0.45 |
| Agent Type Embedding | 640 | 0.00 |
| Motion Token Embedding | 139,520 | 0.53 |
| Agent Shape Embedding | 17,152 | 0.07 |
| Agent ID Embedding | 16,384 | 0.06 |
| Continuous Motion Feature Embedding | 217,600 | 0.83 |
| Special Token Embedding | 512 | 0.00 |
| Reverse Prediction Indicator Embedding | 256 | 0.00 |
| Total | 5,287,105 | 20.17 |

Metrics. To measure the quality and diversity of the model’s predictions during training, we track the perplexity:

$$\text{Perplexity} = \exp \left(- \sum_{a \in \mathcal{A}} \bar{p}_a \log(\bar{p}_a + \epsilon) \right), \quad \text{where} \quad \bar{p}_a = \frac{1}{M} \sum_{i=1}^M \mathbf{1}[\hat{z}_i = a],$$

and M is the number of valid tokens. We also track the number of distinct tokens used by both predictions and ground truth:

$$\text{Cluster} = \sum_{a \in \mathcal{A}} \mathbf{1}[\bar{p}_a > 0].$$

Total Loss. The total loss is the sum of all enabled components:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{main}} + \lambda_{\text{map}} \mathcal{L}_{\text{map}} + \lambda_{\text{tg}} \mathcal{L}_{\text{tg-total}},$$

with default weights $\lambda_{\text{map}} = \lambda_{\text{tg}} = 1$.

A.4 Training Details

Our model has 5.2 million trainable parameters, with details indicated in Table 6. We trained BMT model on the training set of the Waymo Open Motion Dataset [4]. WOMB contains 480K real-world traffic with each scenario of length 9 seconds; traffic are composed by agents of vehicle, pedestrian, and bicycle; Each scenario comes with a high-fidelity road map. During training, we use 8 NVIDIA RTX A6000 GPUs for our model training and fine-tunings. We trained BMT in two stages, each with hyper-parameters indicated in Table 7. In the first stage, we pre-trained BMT for forward prediction only with 1 million steps. Then, we fine-tuned BMT with reverse motion prediction in fine-tuning with totally 1.5 million steps. We use AdamW optimizer for learning rate scheduling.

Table 7: BMT Training settings.

| Forward Prediction | | Reverse Prediction | |
|----------------------|-------|----------------------|-------|
| Hyper-parameter | Value | Hyper-parameter | Value |
| Training steps | 10E6 | Training steps | 15E6 |
| Batch sizes | 2 | Batch size | 2 |
| Training Time (h) | 185 | Training Time (h) | 310 |
| Sampling Topp | 0.95 | Sampling Topp | 0.95 |
| Sampling temperature | 1.0 | Sampling temperature | 1.0 |
| Learning Rates | 3E-4 | Learning Rates | 3E-4 |

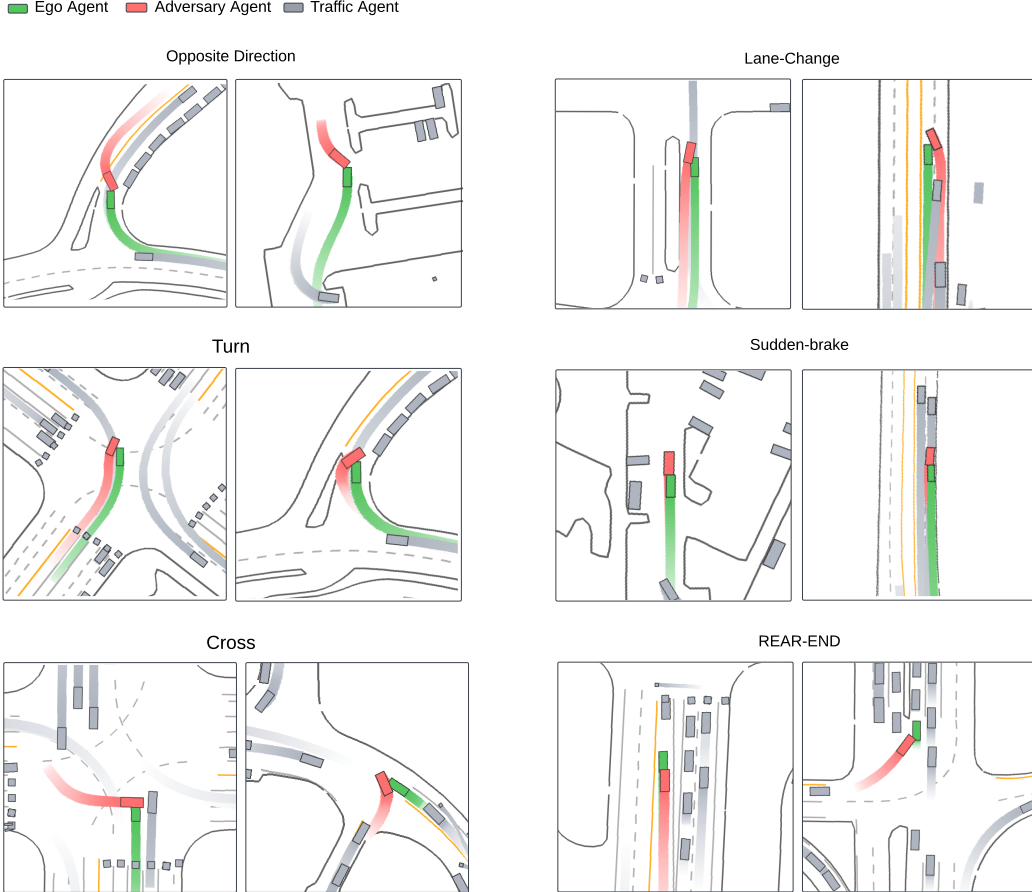


Figure 8: Diverse adversarial behaviors generated by Adv-BMT.

B Additional Experiment Results

Training Environment. We conduct our reinforcement learning experiments using the MetaDrive ScenarioEnv [9], which provides standardized driving environments for training and evaluating autonomous agents. Each environment encodes sensor observations including LiDAR-based surroundings and physical dynamics. Specifically, the observation space consists of three key components: (i) Ego state, which contains the ego vehicle’s current physical state such as speed, heading, and steering; and (ii) surroundings, which encodes nearby traffic objects.

Actions are continuous and correspond to low-level vehicle control commands. The agent outputs a 2D normalized vector, which is then mapped to steering angle, throttle (acceleration), and brake signal. The environment includes a compositional reward structure combining driving progress, collision

Table 8: RL training settings.

| Adv-BMT | | TD3 | |
|-----------------------|-------|-------------------|-------|
| Hyper-parameter | Value | Hyper-parameter | Value |
| Scenario Horizon | 9s | Discounted Factor | 0.99 |
| History Horizon | 0s | Train Batch Size | 1024 |
| Collision Step | 1s–9s | Learning Rate | 1E-4 |
| Prediction Mode | 8 | Policy Delay | 200 |
| Policy Training Steps | 10E6 | Target Network | 0.005 |

Table 9: Realism results for BMT’s reverse prediction.

| Method | SFDE _{avg} | SFDE _{min} | SADE _{avg} | SADE _{min} | Coll _{agent,min} | Coll _{agent,avg} |
|-----------------------------|---------------------|---------------------|---------------------|---------------------|---------------------------|---------------------------|
| Reverse | 3.92 | 2.69 | 5.53 | 5.11 | 0.03 | 0.05 |
| Reverse + Adv-init | 7.17 | 5.56 | 7.68 | 7.12 | 0.13 | 0.16 |
| Reverse + Adv-init + Filter | 6.93 | 5.30 | 7.58 | 7.02 | 0.12 | 0.15 |

Table 10: Diversity results for BMT’s reverse prediction.

| Method | FDD | ADD | JSD _{vel} | JSD _{acc} | JSD _{TTC} |
|-----------------------------|--------------|-------------|--------------------|--------------------|--------------------|
| Reverse | 8.35 | 3.13 | 0.17 | 0.71 | 0.02 |
| Reverse + Adv-init | 9.89 | 3.82 | 0.22 | 0.64 | 0.13 |
| Reverse + Adv-init + Filter | 10.27 | 3.99 | 0.22 | 0.73 | 0.09 |

penalties, and road boundary violations. Driving reward is measured by forward lane progress, while penalties are applied for collisions with other vehicles or drifting off-road.

Hyperparameter. The settings of our open-loop and closed-loop adversarial RL experiments are shown in table 8. Note that in our closed-loop learning, Adv-BMT takes one frame of agent information as input for adversarial generations, whereas all baseline methods take one second agent history.

C Quantitative Results

C.1 Ablation Study

In Table 9 and Table 10, reverse predictions with adversarial initializations exhibit greater deviation from the ground-truth data and yield improved diversity. This behavior is expected, since adversarial initializations modify the terminal positions and headings of selected agents, which propagates backward into more varied histories. Importantly, our rule-based filtering does not reduce diversity; rather, it preserves multimodality while improving realism metrics by removing physically implausible trajectories.

C.2 Waymo Open Sim Agents Challenge

We evaluate BMT results on 400 WOMB validation scenarios using the Waymo Open Sim Agents Challenge (WOSAC) 2025 [11]. Evaluation results are summarized in Table 11. For the metrics, smaller values of minADE indicate more accurate predictions, whereas larger values for the remaining metrics indicate better performance. From the results, we observe that forward prediction achieves much better performance than reverse prediction across all metrics, except for similar performance in angular speed, angular acceleration, distance to the nearest object, and TTC. Note that the training times for forward prediction and reverse prediction are similar (10E6 and 15E6 steps, respectively). The WOSAC results indicate that our BMT model is better at predicting future motion than historical motion.

Table 11: WOSAC Evaluation results of BMT.

| Metrics | Reverse | Forward |
|----------------------------|---------|---------|
| Linear speed | 0.375 | 0.393 |
| Linear acceleration | 0.394 | 0.405 |
| Angular speed | 0.441 | 0.428 |
| Angular acceleration | 0.594 | 0.593 |
| Distance to nearest object | 0.405 | 0.388 |
| Collision | 0.521 | 0.951 |
| Time to Collide | 0.840 | 0.840 |
| Distance to road edge | 0.675 | 0.683 |
| Offroad | 0.564 | 0.934 |
| Realism | 0.554 | 0.753 |
| Kinematic | 0.451 | 0.455 |
| Interactive | 0.566 | 0.801 |
| Map | 0.596 | 0.862 |
| minADE | 2.148 | 1.344 |
| Metametric | 0.554 | 0.753 |

Upon reviewing the collisions detected in reverse predictions with real initializations, we found that most occurred in crowded parking-lot areas, where clusters of parked vehicles or pedestrians are close together. These were flagged as collisions by WOSAC’s evaluation API, even though they may not represent meaningful agent-agent interactions. This explains why metrics such as ADE and FDE remain comparable across methods despite differences in collision scores.

The performance gap is primarily due to our training strategy: the model was pre-trained for forward prediction (around 800k steps) and then fine-tuned for bidirectional prediction (around 1.5M steps). We have included these details in Section A.4 (Training Details).

D Qualitative Results

Visualizations. Figure 8 and Figure 9 present a set of example scenarios results generated by Adv-BMT. Across different scenarios, the adversarial agent exhibits a diverse range of safety-critical driving behaviors, demonstrating their ability to interact plausibly with realistic traffic participants. The visualizations illustrate that Adv-BMT can generate multiple distinct collision outcomes from a single driving log. This highlights a key advantage of Adv-BMT over baseline methods, which tend to produce identical or highly similar adversarial behaviors for the same input scenario.

Demo Video. We submit a video within our supplementary materials. Here we provide visualizations with case studies through animated simulations of Adv-BMT scenarios, which include different types of vehicles, pedestrians, and bicycle agents. More demos are available at <https://metadriverse.github.io/adv-bmt/>.

E Broader Impacts

Our work introduces a novel model for generating safety-critical traffic scenarios, aiming to improve the safety reliability and driving robustness of AD systems. By modeling both forward and reverse motion trajectories, our framework enables controllable and diverse simulation of rare and high-risk traffic events. Our framework, Adv-BMT, benefits the development and testing of safer autonomous agents by exposing failure cases under challenging interactions. However, generating adversarial scenarios may potentially raise concerns about potential misuse, such as crafting unrealistic or malicious simulations. To address this, our approach is designed for research and evaluation within closed simulation environments. We encourage responsible usages of our model and encourage integrating them into safety validation pipelines with appropriate regulations.

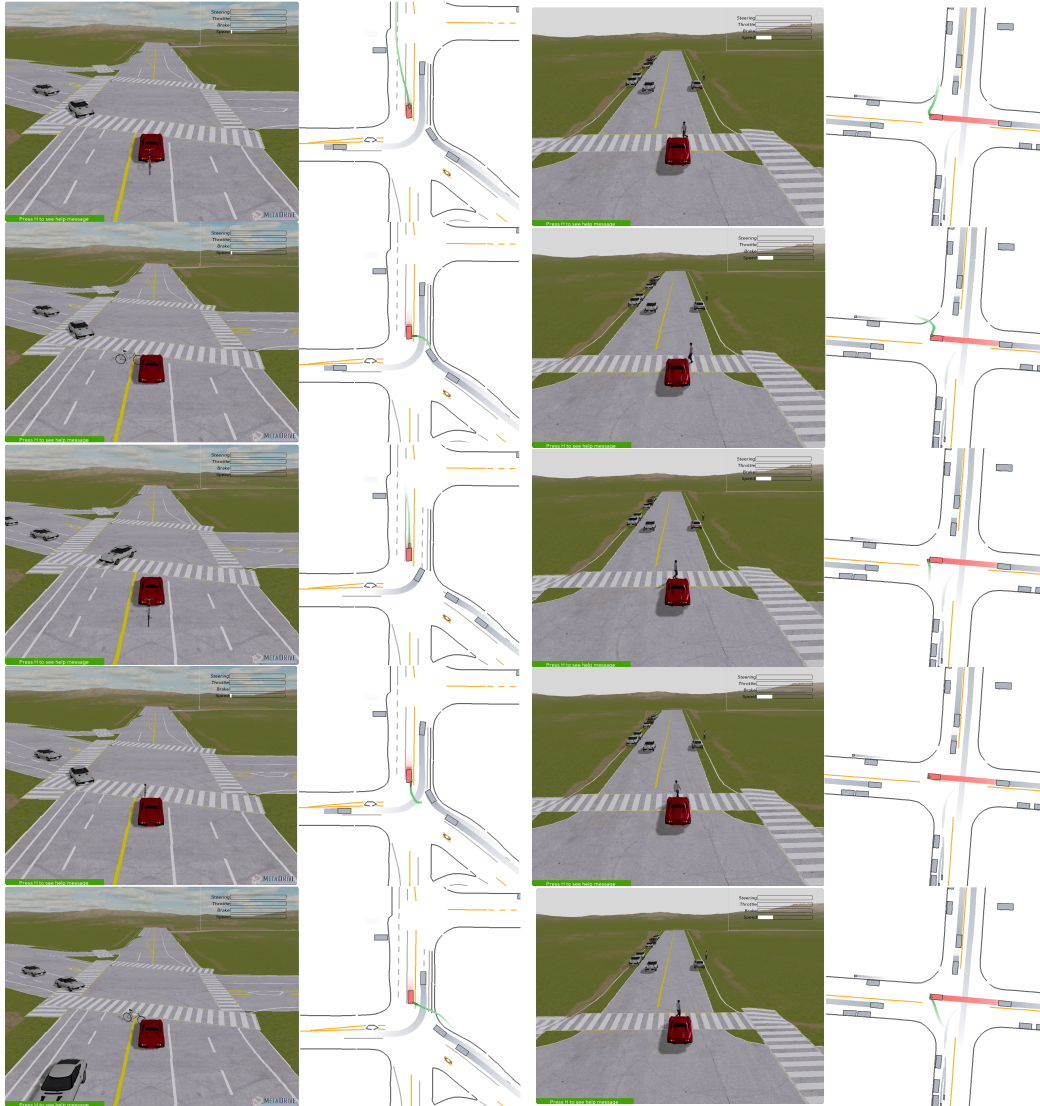


Figure 9: Diverse adversarial behaviors generated by Adv-BMT.