6. Supplemental

6.1. Implementation Detils

Our implementation is built upon the MMDetection3D library [23]. We leverage the Sparse4Dv3 codebase [19] to implement query-based tracking, ensuring compatibility with state-of-the-art vision-based tracking-by-attention methods. For feature extraction, we employ the SECOND LiDAR backbone pretrained on nuScenes for detection provided by FocalFormer3D [6, 30]. This pretrained backbone is used as a frozen BEV feature extractor during training to provide high-quality spatial representations.

To enhance generalization and robustness, standard LiDAR data augmentations, such as rotations, scaling, and flipping, are applied to the LiDAR sequences in a temporally consistent manner. Track sampling augmentation [7] is also applied to introduce temporally consistent synthetic object tracks to each sequence, increasing the diversity of training scenarios. Temporal instance denoising [19] is used during training to improve the convergence by adding denoising query groups that are propagated over multiple iterations.

Proposal queries are initialized using 900 anchor boxes derived from k-means clustering and are further refined through training [19]. Temporal instance denoising, as described in Sparse4Dv3 [19], provides additional auxiliary queries with known ground truth assignments to improve convergence. Following the detection decoder, the top 300 most confident proposal queries are appended to the track queries before the track decoder. Track Query Dropout is applied before propagating track queries to the next iteration. The main track query group passes the top 600 most confident queries, and the auxiliary group passes 600 random queries as track queries, resulting in a total of two track query groups used during training. Deformable attention is used for all cross-attention operations between the queries and the BEV features [6, 38]. The detection decoder consists of one transformer decoder layer, while the tracking decoder consists of five layers [19].

To train SCATR sequentially [19, 28], each training scene is divided into clips of up to 10 annotated frames. For inference, we use the full sequences of 40 frames. A custom data sampler handles the batching of sequences of varying lengths. To improve minority class performance, we use a modified class-balanced data sampling method [23, 37]. Rather than using the set of classes present in each frame to sample individual frames in a class-balanced manner, the set of classes present in each clip is used to sample subsequences in a class-balanced manner. This ensures that the sequential data loading is maintained rather than having individual frames of a sequence sampled multiple times.

All experiments were trained using a cyclic learning rate and momentum scheduler with the AdamW optimizer [20]. Models were trained using 4 NVIDIA Tesla V100 GPUs for 70,000 iterations and a total batch size of 24, with track sampling and Track Query Dropout disabled after 52,500 iterations. SCATR performs inference at 2.9 Hz on a single V100 GPU.

6.2. Qualitative Results

BEV visualizations on the nuScenes mini split can be found at https://drive.google.com/drive/folders/1_l5nF5_qw_I-CAjFHL0BtPf9xI1UphQN?usp=sharing, showing the detection and tracking performance of SCATR. Track IDs are illustrated with consistent colours in the sequence, and ground truths are illustrated with dashed black boxes.