Supplementary Material for SparseBEV

Haisong Liu 1  Yao Teng 1  Tao Lu 1  Haiguang Wang 1  Limin Wang 1, 2
1 State Key Laboratory for Novel Software Technology, Nanjing University 2 Shanghai AI Lab
{liuhs, yaoteng, taolu, haiguangwang}@smail.nju.edu.cn, lmwang@nju.edu.cn

Figure 1: Comparison between single-branch and dual-branch under different settings. Dual branch design brings more gain as the number of frames increases.

A. Details of Dual-branch SparseBEV

In this section, we provide detailed explanations and ablations on the dual branch design. As shown in Fig. 2, the input multi-camera videos are divided into a high-resolution “slow” stream and a low-resolution “fast” stream. Sampling points are projected to the two streams respectively and the sampled features are stacked before adaptive mixing. Experiments are conducted with a V2-99 [1] backbone pre-trained by FCOS3D [2] on the training set of nuScenes. 1

In Fig. 1, we compare our dual branch design with single branch baselines. If we use a single branch of 1600 \( \times \) 640 (orange curve) resolution, adding more frames does not provide as much benefit as it does at 800 \( \times \) 320 resolution (green curve). By using dual branch of 1600 \( \times \) 640 and 640 \( \times \) 256 resolution with 1:2 ratio, we decouple spatial appearance and temporal motion, unlocking better performance. As we can see from the blue curve, the longer the frame sequence, the more gain the dual branch design brings.

In Tab. 1, we provide detailed qualitative results. Under the setting of 8 frames (\( \sim \) 4 seconds), our dual branch design with only two high resolution (HR) frames surpasses the baseline with eight HR frames. By increasing the number of HR frames to 4, we further improve the performance by 0.8 mAP and 0.5 NDS. Moreover, increasing the resolution of the LR frames does not bring any improvement, which clearly demonstrates that appearance detail and temporal motion are decoupled to different branches.

Since the dual-branch design also enlarges the receptive field (smaller resolution provides larger receptive field) which may improve performance, we further analyse where the improvement comes from in Tab. 2. The first row is our baseline which takes 8 frames with a single branch of 1600 \( \times \) 640 as input. We first try to increase the receptive field by adding an extra \( C_6 \) feature map (Row 2), and observe that the performance is slightly improved. This demonstrates that a larger receptive field is required for high-resolution and long-term inputs. However, the spatial appearance and temporal motion is still coupling, limiting the performance. By using dual branches of 1600 \( \times \) 640

<table>
<thead>
<tr>
<th>Method</th>
<th>Feature Maps</th>
<th>Train. Cost</th>
<th>mAP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Single branch</td>
<td>( C_2, C_3, C_4, C_5 )</td>
<td>2d 17h</td>
<td>48.9</td>
</tr>
<tr>
<td>Single branch</td>
<td>( C_2, C_3, C_4, C_5, C_6 )</td>
<td>2d 18h</td>
<td>49.3</td>
</tr>
<tr>
<td>Dual branch</td>
<td>( C_2, C_3, C_4, C_5 )</td>
<td>1d 19h</td>
<td>50.2</td>
</tr>
</tbody>
</table>

Table 2: Detailed analyses on the dual-branch design. For single branch baselines, simply adding an extra \( C_6 \) feature map has limited effect. In contrast, our dual branch design can boost the performance significantly.

Note that the experiment setting used here is different from that in the main paper, since the experiments are conducted before the submission of ICCV 2023. After submission, we further improve our implementation to refresh our results. The conclusion is consistent between these different implementations.
Figure 2: Architecture of dual-branch SparseBEV. The input multi-camera videos are divided into a high-resolution “slow” stream and a low-resolution “fast” stream.

Table 3: Compared with SASA-beta, SASA not only has the ability of multi-scale feature aggregation, but generates adaptive receptive field for each query as well.

<table>
<thead>
<tr>
<th></th>
<th>Distance Function</th>
<th>NDS</th>
<th>mAP</th>
</tr>
</thead>
<tbody>
<tr>
<td>SASA-beta</td>
<td>$\tau D$</td>
<td>55.2</td>
<td>44.8</td>
</tr>
<tr>
<td>SASA</td>
<td>$\tau D$</td>
<td>55.6</td>
<td>45.4</td>
</tr>
</tbody>
</table>

Figure 3: The change of $\tau$ of each head in SASA-beta during training. Regardless of the initialization, each head learns a different $\tau$, enabling local and multi-scale feature aggregation.

B. Study on Scale-adaptive Self Attention

In this section, we’ll talk about how we came up with scale-adaptive self attention (SASA). In the main paper, the receptive field coefficient $\tau$ is specific to each head and adaptive to each query. In the development of SASA, there is an intermediate version (dubbed SASA-beta for convenience): the $\tau$ for each head is simply a learnable parameter shared by all queries.

In Fig. 3, we take a closer look at how $\tau$ changes with training. We surprisingly find that regardless of the initialization, each head learns a different $\tau$ from the others and all of them are distributed in range $[0, 2]$, enabling the network to aggregate local and multi-scale features from multiple heads.

Next, we improve SASA-beta by generating the $\tau$ adaptively from the query, which corresponds to the version in the main paper. Compared with SASA-beta, SASA not only has the ability of multi-scale feature aggregation, but generates adaptive receptive field for each query as well. The quantitative comparison between SASA-beta and SASA is shown in Tab. 3.

C. More Visualizations

In Fig. 4, we provide more visualizations of the sampling points from different stages. In the initial stage, the sampling points have the shape of pillars. In later stages, they are refined to cover objects with different sizes.
Figure 4: Visualized sampling points from different stages. Different instances are distinguished by colors.

References
