Supplementary Material

1. NearFarMix Augmentation

Figure 1 elegantly illustrates the operation of the proposed NearFarMix augmentation, delineating the four regions—near, pre-far, overlap, and exclusive—that contribute to the final augmented output. Despite the amalgamation of four regions, the pre-far, overlap, and exclusive regions primarily generate far regions, with overlap regions consequently subtracted. Importantly, the augmentation is designed for batch-wise application to enhance execution speed. The step-by-step implementation is comprehensively depicted in Algorithm 3.

Additionally, Fig. 3 and Fig. 2 provide additional examples between the proposed NearFarMix and DepthMix [2].

Algorithm 1 Batchwise Window Partition

\[
x \leftarrow \text{Input} \quad \triangleright \text{Features} \\
h_{w}, w_{w} \leftarrow \text{window\_size} \quad \triangleright \text{Size of each window}
\]

\# Partition features into local dense windows
\[
B, H, W, C = \text{shape}(x) \\
x = \text{reshape}(x, \text{shape} = [B, \frac{H}{h_{w}}, h_{w}, \frac{W}{w_{w}}, w_{w}, C]) \\
x = \text{transpose}(x, \text{permute\_axis} = [0, 1, 3, 2, 4, 5]) \\
x = \text{reshape}(x, \text{shape} = [B \times \frac{H}{h_{w} \times w_{w}}, h_{w}, w_{w}, C])
\]

Algorithm 2 Batchwise Grid Partition

\[
x \leftarrow \text{Input} \quad \triangleright \text{Features} \\
h_{g}, w_{g} \leftarrow \text{grid\_size} \quad \triangleright \text{Size of each grid}
\]

\# Partition features into global sparse grids
\[
B, H, W, C = \text{shape}(x) \\
x = \text{reshape}(x, \text{shape} = [B, h_{g}, \frac{H}{h_{g}}, w_{g}, \frac{W}{w_{g}}, w_{g}, C]) \\
x = \text{transpose}(x, \text{permute\_axis} = [0, 1, 3, 2, 4, 5]) \\
x = \text{reshape}(x, \text{shape} = [B \times \frac{H}{h_{g} \times w_{g}}, h_{g}, w_{g}, C])
\]

Algorithm 3 Batchwise NearFarMix Augmentation

\[
I_{1} \leftarrow \text{Images} \quad \triangleright \text{Input images} \\
D_{1} \leftarrow \text{Depths} \quad \triangleright \text{Input depths} \\
S_{1} \leftarrow \text{Semantics} \quad \triangleright \text{Input semantics} \\
U \leftarrow \text{random\_uniform} \quad \triangleright \text{Uniform distribution}
\]

\[
\begin{align*}
D_{\text{min}} & \leftarrow 20(\text{KITTI}) \quad \lor \quad 1.5(\text{NYUv2}) \quad \triangleright \text{Min depth} \\
D_{\text{max}} & \leftarrow 60(\text{KITTI}) \quad \lor \quad 6.5(\text{NYUv2}) \quad \triangleright \text{Max depth}
\end{align*}
\]

\# Roll batch for fast shuffling
\[
I_{2} = \text{roll}(I_{1}, \text{shift} = 1, \text{axis} = 0) \quad \triangleright \text{Roll Images} \\
D_{2} = \text{roll}(D_{1}, \text{shift} = 1, \text{axis} = 0) \quad \triangleright \text{Roll Depths} \\
S_{2} = \text{roll}(S_{1}, \text{shift} = 1, \text{axis} = 0) \quad \triangleright \text{Roll Semantics}
\]

\# Depth threshold range for batch
\[
d_{\text{min}} = \max(\min(D_{1}, \text{axis} = (1, 2, 3))) \quad \triangleright \text{Min depth} \\
d_{\text{max}} = \min(\max(D_{1}, \text{axis} = (1, 2, 3))) \quad \triangleright \text{Max depth}
\]

\# Threshold for Near-Far region
\[
\begin{align*}
th_{\text{min}} & = \max(D_{\text{min}}, d_{\text{min}}) \quad \triangleright \text{Clip min depth} \\
th_{\text{max}} & = \min(D_{\text{max}}, d_{\text{max}}) \quad \triangleright \text{Clip max depth} \\
\text{thr} & = U(\text{shape} = [B, 1, 1, 1], \triangleright \text{Random thresholds} \\
\text{min} & = \text{thr}_{\text{min}} \quad \triangleright \text{Random thresholds} \\
\text{max} & = \text{thr}_{\text{max}} \quad \triangleright \text{Random thresholds}
\end{align*}
\]

\# Compute binary masks of regions for blending
\[
\begin{align*}
M_{1} & = D_{1} < \text{thr} \quad \triangleright \text{Broadcasted Near region mask} \\
M_{2} & = D_{2} > \text{thr} \quad \triangleright \text{Broadcasted pre-Far region mask} \\
M_{o} & = M_{1} \land M_{2} \quad \triangleright \text{Overlap region mask} \\
M_{e} & = (1 - M_{1}) \land (1 - M_{2}) \quad \triangleright \text{Exclusive region mask}
\end{align*}
\]

\# Perform blending of regions
\[
\begin{align*}
I' & = (I_{1} \land M_{1})_{\text{near}} + ((I_{2} \land M_{2}) + (I_{2} \land M_{e}) - (I_{2} \land M_{o}))_{\text{far}} \quad \triangleright \text{Augmented image} \\
D' & = (D_{1} \land M_{1})_{\text{near}} + ((D_{2} \land M_{2}) + (D_{2} \land M_{e}) - (D_{2} \land M_{o}))_{\text{far}} \quad \triangleright \text{Augmented depths} \\
S' & = (S_{1} \land M_{1})_{\text{near}} + ((S_{2} \land M_{2}) + (S_{2} \land M_{e}) - (S_{2} \land M_{o}))_{\text{far}} \quad \triangleright \text{Augmented semantics}
\end{align*}
\]

2. Symbiotic Transformer

Transformers: Equation (1) presents the detailed mathematical expression for Symbiotic Transformer, which sym-
Figure 1. Proposed NearFarMix augmentation. The depth map undergoes thresholding to generate Near and pre-Far regions. Then, these regions are manipulated to produce Overlap and Exclusive regions. The Far region is then generated by combining the pre-Far, Overlap, and Exclusive regions, with the Exclusive region being subtracted and the remaining regions added. Finally, the Near and Far regions are combined to generate the augmented image.

Biotically enhances both depth and semantics via local-global cross-attention. In the equation, $F_{Q}^{x}$ represents query features of $x$, $F_{K}^{y}$ and $F_{V}^{y}$ denote key-value features of $y$, and $F_{y}$ signifies the output features contextualized by $x$. Specifically, for SGT, $x = s$ and $y = d$, while for DGT, $x = d$ and $y = s$. Moreover, DGT = LG-CAT_{DG} and SGT = LG-CAT_{SC} correspond to depth and semantics-guided local-global cross-attention transformers.

**Cross Attentions:** Under the hood, DGT and SGT employ semantics-guided cross attention (SG-CA) and depth-guided cross attention (DG-CA) respectively to contextualize features. SG-CA and DG-CA is mathematically elaborated in Eq. (2) and Eq. (3). In these equations, $W$ represents the weight of the FFN layer, $Q_{x}$ represents query features, $K_{y}$ and $V_{y}$ represent key and value features, $S_{max}$ denotes softmax, and $d$ is the query/key dimension. $B$ represents relative positional bias, sampled similar to [1]. The Local-Global Cross-Attention Transformer (LG-CAT), employed by both SGT and DGT, can be implemented using Algorithm 4.

**Partition Operation:** Further, the algorithms for WindowPartition and GridPartition operations which are used in Block and Grid attention also slightly different from the methods of Max-ViT [4], are detailed in Algorithm 1 and Algorithm 2, respectively. It is noteworthy that Max-ViT implemented these operations using einops [3].
Figure 2. Additional qualitative comparisons between proposed NearFarMix and DepthMix augmentation on NYUv2 dataset.

Figure 3. Additional qualitative comparisons between proposed NearFarMix and DepthMix augmentation on KITTI dataset.

\[
\text{Sym-T}(\mathbf{F}_d', \mathbf{F}_s') = \begin{bmatrix}
\mathbf{F}^Q_d &=& \mathbf{F}^{KV}_d &=& \mathbf{F}'_d \\
\mathbf{F}^Q_s &=& \mathbf{F}^{KV}_s &=& \mathbf{F}'_s \\
\mathbf{F}_d &=& \text{LG-CAT}_{sd}(\mathbf{F}^Q_s, \mathbf{F}^{KV}_d) \\
\mathbf{F}_d &=& \text{LG-CAT}_{rd}(\mathbf{F}^Q_d, \mathbf{F}^{KV}_s)
\end{bmatrix}
\]

(1)

\[
\text{SG-CA} = \text{CA}(\mathbf{Q}_d, \mathbf{K}_d, \mathbf{V}_d) \\
= \mathcal{S}_{\text{Lux}} \left( \frac{\mathbf{Q}_d \mathbf{K}_d^T}{\sqrt{d}} + \mathbf{B} \right) \mathbf{V}_d \\
= \mathcal{S}_{\text{Lux}} \left( \frac{(\mathbf{F}^Q_d \cdot \mathbf{W}^Q_s)(\mathbf{F}^{KV}_d \cdot \mathbf{W}^{KV}_s)^T}{\sqrt{d}} + \mathbf{B} \right) (\mathbf{F}^{KV}_d \cdot \mathbf{W}^Q_s)
\]

(2)
Algorithm 4 Local-Global Cross-Attention Transformer (LG-CAT)

\[
\begin{align*}
F^Q, F^K_V & \leftarrow \text{inputs} \quad \triangleright \text{Input features} \\
x & \leftarrow F^Q \quad \triangleright \text{Query features of depth/semantics} \\
y & \leftarrow F^K_V \quad \triangleright \text{Key-value features of semantics/depth} \\
i & \leftarrow 0 \quad \triangleright \text{Initialize counter}
\end{align*}
\]

\[
\text{while } i \neq 2 \text{ do}
\]

# Block Cross Attention
\[
x_1 = \text{layer_norm}(x) \\
y_1 = \text{layer_norm}(y) \\
x_{1,q} = \text{FFN}(\text{window_partition}(x_1)) \quad \triangleright \text{Query gen.} \\
y_{1,k}, y_{1,v} = \text{FFN}(\text{window_partition}(y_1)) \quad \triangleright \text{KeyValue} \\
y_2 = \text{CA}(x_{1,q}, y_{1,k}, y_{1,v}) \quad \triangleright \text{Apply cross-attention} \\
y_2 = y_1 + \text{window_reverse}(%\text{FFN}(y_2)) \quad \triangleright \text{Residual}
\]

# Grid Cross Attention
\[
y_2 = \text{layer_norm}(y_2) \\
x_{2,q} = \text{FFN}(\text{grid_partition}(x_1)) \quad \triangleright \text{Query} \\
y_{2,k}, y_{2,v} = \text{FFN}(\text{grid_partition}(y_2)) \quad \triangleright \text{KeyValue} \\
y_3 = \text{CA}(x_{2,q}, y_{2,k}, y_{2,v}) \quad \triangleright \text{Apply cross-attention} \\
y_3 = y_2 + \text{grid_reverse}(%\text{FFN}(y_3)) \quad \triangleright \text{Residual} \\
y = y_3 \quad \triangleright \text{Reset variable for loop} \\
i = i + 1 \quad \triangleright \text{Increment counter}
\]

end while

# FusedMBConv - Channel Attention
\[
\hat{y} = \text{DWConv3x3}(y) \quad \triangleright \text{Depthwise convolution} \\
\hat{y} = \text{GELU}(\hat{y}) \quad \triangleright \text{Apply activation} \\
\hat{y} = \text{SE}(\hat{y}) \quad \triangleright \text{Squeeze-Excitation} \\
\hat{y} = \text{Conv1x1}(\hat{y}) \quad \triangleright \text{Convolution} \\
\hat{y} = y + \hat{y} \quad \triangleright \text{Residual}
\]

\[
\text{output} \leftarrow \hat{y}
\]

DG-CA = CA (Q_d, K_s, V_s)
\[
=D_{\text{mx}} \left( \frac{Q_d K_s T}{\sqrt{d}} + B \right) V_s \\
=D_{\text{mx}} \left( \frac{(F^Q_d \cdot W^Q_d) (F^K_V \cdot W^K_V) T}{\sqrt{d}} + B \right) (F^K_V \cdot W^V_V)
\]

3. Architecture Details

The architectural specifications, encompassing input, output, layer name, and layer details, are succinctly laid out in Table 1. Here, \( E \) and \( D \) represent the input/output of the encoder/decoder, while \( ST \) and \( N \) denote to the Symbiotic Transformer and the Neck.
Table 1. Architectural Specifications of proposed method where $h$, $w$ signify attention heads and window size; $Conv$ denotes 2D convolution with $k$, $s$, $c$ as kernel size, stride size, and output channels; $act$ and $norm$ represent activation and normalization types; $sc$ indicates upscale size.

<table>
<thead>
<tr>
<th>Layer Name</th>
<th>Input</th>
<th>Output</th>
<th>Output Size</th>
<th>Architecture</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Stem</strong></td>
<td><em>Image</em></td>
<td>$E_0$</td>
<td>$H_4 \times W_2 \times 128$</td>
<td>$Conv(c=128, k=3, s=2)$</td>
</tr>
<tr>
<td><strong>Encoder</strong></td>
<td></td>
<td>$E_0$</td>
<td>$E_1$</td>
<td>$H_8 \times W_4 \times 128$</td>
</tr>
<tr>
<td>Stage 1</td>
<td></td>
<td>$E_1$</td>
<td>$E_2$</td>
<td>$H_{16} \times W_8 \times 256$</td>
</tr>
<tr>
<td>Stage 2</td>
<td></td>
<td>$E_2$</td>
<td>$E_3$</td>
<td>$H_{32} \times W_{16} \times 512$</td>
</tr>
<tr>
<td>Stage 3</td>
<td></td>
<td>$E_3$</td>
<td>$E_4/D_4$</td>
<td>$H_{32} \times W_{32} \times 1024$</td>
</tr>
<tr>
<td><strong>Decoder</strong></td>
<td>($D_4, E_3$)</td>
<td>$D_3$</td>
<td></td>
<td>$H_{32} \times W_{32} \times 512$</td>
</tr>
<tr>
<td>Stage 3</td>
<td></td>
<td>($D_3, E_2$)</td>
<td>$D_2$</td>
<td>$H_{16} \times W_{16} \times 256$</td>
</tr>
<tr>
<td>Stage 1</td>
<td>($D_2, E_1$)</td>
<td>$D_1$</td>
<td></td>
<td>$H_4 \times W_4 \times 128$</td>
</tr>
<tr>
<td><strong>Neck</strong></td>
<td>$D_1$</td>
<td>($N_d, N_s$)</td>
<td></td>
<td>$(H_4 \times W_4 \times 150, H_4 \times W_4 \times 150)$</td>
</tr>
<tr>
<td><strong>Symbiotic</strong></td>
<td>($N_d, N_s$)</td>
<td>($ST_d, ST_s$)</td>
<td></td>
<td>$(H_4 \times W_4 \times 150, H_4 \times W_4 \times 150)$</td>
</tr>
<tr>
<td>Transformer</td>
<td></td>
<td></td>
<td></td>
<td>$[\text{Block-Cross-Attention(h=4, w=7)} \times 2, \text{Grid-Cross-Attention(h=4, w=7)}]$</td>
</tr>
<tr>
<td><strong>Head</strong></td>
<td>($ST_d, ST_s$)</td>
<td>($Depth, Semantics$)</td>
<td>($H \times W \times 1, H \times W \times 150$)</td>
<td>$(\text{Conv(k=3, s=1, act='sigmoid')} \times 2, \text{Upsample(sc=4)}, \text{Conv(k=3, s=1, act='softmax')} \times 2)$</td>
</tr>
</tbody>
</table>