1. Network Details

Below we describe each network used in our proposed DepthFormer architecture, and Table 1 shows detailed diagrams for each of them. Note that our contributions do not require any network architecture in particular, and can be extended to incorporate recent developments for potential further improvements in performance [7, 16, 17]. Open-source training and inference code, as well as pre-trained models, will be made available upon publication.

1.1. Cross-Attention Network

Similar to [15], we use an hourglass-shaped architecture as the encoder, modified with residual connections and spatial pyramid pooling modules [1]. The decoder consists of transposed convolutions, dense-blocks [10], and a final convolution layer. The final feature map has the same spatial resolution as the input image, encoding both local and global contexts. This feature map is then downsampled to the cost volume resolution using bilinear interpolation.

1.2. Single-Frame Depth Network

We use a ResNet18 backbone [6] as the single-frame encoder, followed by a decoder that outputs multi-scale depth maps at four different resolutions: one-eighth, one-fourth, one-half, and the original input dimension. Following [21], we concatenate the $H/4 \times W/4 \times 128$ encoded features with the $H/4 \times W/4 \times D$ multi-frame cost volume. A bottleneck convolutional layer, with kernel size 3, is then used to combine these two sources of features (single-frame and multi-frame) into a $H/4 \times W/4 \times 128$ feature map for further encoding and decoding (Figure 4, main paper).

1.3. Context Adjustment Network

The input to our context adjustment network is a $H/4 \times W/4 \times 4$ tensor created by concatenating the normalized high-response depth map $\tilde{D}_H$ and the target image $I_t$. Depth map normalization is done as such:

$$\tilde{D}_H = \left( D_H - \text{mean}(\tilde{D}_H) \right) / \text{std}(\tilde{D}_H)$$

This normalized high-response depth map is refined through a series of residual blocks that expand the channel dimensions, before a ReLU activation restores it to the original shape. The high-response depth map is concatenated with the output of each residual block, and added to the final output using a long skip connection. This final output is then un-normalized using the original statistics, generating a context-adjusted predicted depth map $\hat{D}_C$:

$$\hat{D}_C = \left( \tilde{D}_H + \theta_C(I_t, \tilde{D}_H) \right) \text{std}(\tilde{D}_H) + \text{mean}(\tilde{D}_H)$$

1.4. Pose Network

Our pose network uses a ResNet18 backbone, modified to accommodate two input images by duplicating the convolutional weights of the first layer [6]. The bottleneck feature maps are further processed using a series of convolutional layers, with the last one outputting a $H/32 \times H/32 \times 6$ feature map. This feature map is then averaged over the spatial dimensions, generating a 6-dimensional vector containing the relative translation and rotation between frames, in Euler angles. Following [21], we invert the order of input images when predicting backwards motion.

2. Comparison to Supervised Methods

Our DepthFormer architecture was designed for self-supervised learning, in which training is conducted without explicit supervision from ground-truth depth maps. As mentioned in the main paper (Section 2.1), this is a very challenging setting, due to limitations of the photometric objective in the presence of dynamic objects, static frames, changes in luminosity, and so forth. Even so, our contributions in multi-frame feature matching lead to a depth estimation performance that surpasses even current state-of-the-art single-frame supervised depth estimation methods. These results are summarized in Table 2. More specifically, we achieve comparable performance to BTS [3] when training and evaluating at half resolution ($640 \times 192$), and surpass it in almost all metrics when training and evaluating at the same full resolution ($1216 \times 352$). We believe the introduction of other self-supervised depth network architectures
**Network architectures used in our experiments.** BN stands for Batch Normalization [12]. Upsample and Downsample respectively increases and decreases spatial dimensions using bilinear interpolation to match the output resolution, ReLU are Rectified Linear Units, Sigmoid is the sigmoid activation function, and DenseBlock are densely connected convolutional layers from [10]. The symbol ⊕ indicates feature concatenation, and ⋆ indicates element-wise multiplication.

<table>
<thead>
<tr>
<th>Layer Description</th>
<th>K</th>
<th>S</th>
<th>Output Dim.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ResidualBlock (#0)</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>#1 Conv2d (#0a) → BN → ReLU</td>
<td>K</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>#2 Conv2d → BN → ReLU</td>
<td>K</td>
<td>S</td>
<td></td>
</tr>
<tr>
<td>#3 Downsample(#0) + #2 → ReLU</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td><strong>UpsampleBlock (#0, #s)</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>#1 Conv2d → BN → ReLU → Upsample</td>
<td>3</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>#2 Conv2d (#1 ⊕ #s) → BN → ReLU</td>
<td>3</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td><strong>InverseDepth (#0)</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>#1 Conv2d → Sigmoid</td>
<td>K</td>
<td>S</td>
<td></td>
</tr>
<tr>
<td>#2 (max - min) ⊕ #1 + min</td>
<td>-</td>
<td>-</td>
<td>3×H×W</td>
</tr>
<tr>
<td>#0a Input RGB image</td>
<td>-</td>
<td>-</td>
<td>128×H/4×W/4</td>
</tr>
<tr>
<td>#0b Input cost volume</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td><strong>Encoder</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>#1 Conv2d → BN → ReLU</td>
<td>7</td>
<td>1</td>
<td>64×H×W</td>
</tr>
<tr>
<td>#2 Max. Pooling</td>
<td>3</td>
<td>2</td>
<td>64×H/2×W/2</td>
</tr>
<tr>
<td>#3 ResidualBlock (#2) x2</td>
<td>3</td>
<td>1-2</td>
<td>64×H/4×W/4</td>
</tr>
<tr>
<td>#4 ⊕ #0b ⊕ Conv2d → BN → ReLU</td>
<td>3</td>
<td>1</td>
<td>64×H/4×W/4</td>
</tr>
<tr>
<td>#5 ResidualBlock (#4) x2</td>
<td>3</td>
<td>1-2</td>
<td>128×H/8×W/8</td>
</tr>
<tr>
<td>#6 ResidualBlock (#5) x2</td>
<td>3</td>
<td>1-2</td>
<td>256×H/16×W/16</td>
</tr>
<tr>
<td>#7 ResidualBlock (#6) x2</td>
<td>3</td>
<td>1-2</td>
<td>512×H/32×W/32</td>
</tr>
<tr>
<td><strong>Decoder</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>#8 UpsampleBlock (#7,#6)</td>
<td>3</td>
<td>1</td>
<td>256×H/16×W/16</td>
</tr>
<tr>
<td>#9 UpsampleBlock (#8,#5)</td>
<td>3</td>
<td>1</td>
<td>128×H/8×W/8</td>
</tr>
<tr>
<td>#10 InverseDepth (#8)</td>
<td>3</td>
<td>1</td>
<td>1×H/8×W/8</td>
</tr>
<tr>
<td>#11 UpsampleBlock (#9,#3)</td>
<td>3</td>
<td>1</td>
<td>64×H/4×W/4</td>
</tr>
<tr>
<td>#12 InverseDepth (#11)</td>
<td>3</td>
<td>1</td>
<td>1×H/4×W/4</td>
</tr>
<tr>
<td>#13 UpsampleBlock (#11,#2)</td>
<td>3</td>
<td>1</td>
<td>32×H/2×W/2</td>
</tr>
<tr>
<td>#14 InverseDepth (#13)</td>
<td>3</td>
<td>1</td>
<td>1×H/2×W/2</td>
</tr>
<tr>
<td>#15 UpsampleBlock (#13,#-)</td>
<td>3</td>
<td>1</td>
<td>32×H×W</td>
</tr>
<tr>
<td>#16 InverseDepth (#15)</td>
<td>3</td>
<td>1</td>
<td>1×H×W</td>
</tr>
<tr>
<td><strong>ResidualBlock (#0): N × H × W</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>#1 Conv2d → BN → ReLU</td>
<td>K</td>
<td>3</td>
<td>3N×H×W</td>
</tr>
<tr>
<td>#2 Conv2d → BN → ReLU</td>
<td>K</td>
<td>S</td>
<td>N×H×W</td>
</tr>
<tr>
<td>#3 Downsample(#0) + #2 → ReLU</td>
<td>-</td>
<td>-</td>
<td>N×H×W</td>
</tr>
<tr>
<td><strong>Decoder</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>#7 Conv2d → ReLU</td>
<td>1</td>
<td>1</td>
<td>256×H/32×W/32</td>
</tr>
<tr>
<td>#8 Conv2d → ReLU</td>
<td>3</td>
<td>1</td>
<td>256×H/32×W/32</td>
</tr>
<tr>
<td>#9 Conv2d → ReLU</td>
<td>3</td>
<td>1</td>
<td>256×H/32×W/32</td>
</tr>
<tr>
<td>#10 Conv2d → ReLU</td>
<td>3</td>
<td>1</td>
<td>6×H/32×W/32</td>
</tr>
<tr>
<td>#11 Global Avg. Pooling</td>
<td>-</td>
<td>-</td>
<td>6×1×1</td>
</tr>
</tbody>
</table>

(a) Single-frame depth network. The target image I_t is used as input, as well as the cross-attention cost volume A_{t→c}. Bold numbers indicate the 4 multi-scale output inverse depth maps, at increasing resolutions. Each sigmoid output is converted to depth using min and max ranges.

<table>
<thead>
<tr>
<th>Layer Description</th>
<th>K</th>
<th>S</th>
<th>Output Dim.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ResidualBlock (#0)</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>#1 Conv2d → BN → ReLU</td>
<td>K</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>#2 Conv2d → BN → ReLU</td>
<td>K</td>
<td>S</td>
<td></td>
</tr>
<tr>
<td>#3 Downsample(#0) + #2 → ReLU</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td><strong>SpatialPyramidBlock (#0, N)</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>#1 Avg. Pool</td>
<td>N</td>
<td>N</td>
<td></td>
</tr>
<tr>
<td>#2 Conv2d → BN → ReLU</td>
<td>K</td>
<td>S</td>
<td></td>
</tr>
<tr>
<td>#3 ResidualBlock(#3) x2</td>
<td>3</td>
<td>1</td>
<td>32×H/2×W/2</td>
</tr>
<tr>
<td>#4 ResidualBlock(#3) x2</td>
<td>3</td>
<td>1</td>
<td>64×H/4×W/4</td>
</tr>
<tr>
<td>#5 ResidualBlock(#3) x2</td>
<td>3</td>
<td>1</td>
<td>128×H/8×W/8</td>
</tr>
<tr>
<td>#6 SpatialPyramidBlock(#5,16)</td>
<td>1</td>
<td>1</td>
<td>32×H/16×W/16</td>
</tr>
<tr>
<td>#7 SpatialPyramidBlock(#5,8)</td>
<td>1</td>
<td>1</td>
<td>32×H/8×W/8</td>
</tr>
<tr>
<td>#8 SpatialPyramidBlock(#5,4)</td>
<td>1</td>
<td>1</td>
<td>32×H/16×W/16</td>
</tr>
<tr>
<td>#9 SpatialPyramidBlock(#5,2)</td>
<td>1</td>
<td>1</td>
<td>32×H/8×W/4</td>
</tr>
<tr>
<td>#10 Downsample(#6 ⊕ #7 ⊕ #8 ⊕ #9)</td>
<td>-</td>
<td>-</td>
<td>128×H/32×W/32</td>
</tr>
<tr>
<td>#11 DenseBlock(#0 ⊕ #10)</td>
<td>1</td>
<td>1</td>
<td>128×H×W</td>
</tr>
<tr>
<td>#12 DenseBlock(#0 ⊕ #11)</td>
<td>1</td>
<td>1</td>
<td>128×H×W</td>
</tr>
<tr>
<td>#13 DenseBlock(#5 ⊕ #12)</td>
<td>1</td>
<td>1</td>
<td>128×H×W</td>
</tr>
<tr>
<td>#14 DenseBlock(#10 ⊕ #13)</td>
<td>1</td>
<td>1</td>
<td>128×H×W</td>
</tr>
</tbody>
</table>

(b) Attention network. It processes the target I_t and context images I_c independently, and the output is used to generate the cross-attention cost volume A_{t→c}, as described in Section 3.2.2, main paper.

(d) Context adjustment network. Input high-response depth maps $\hat{D}_H$ are normalized using Equation 1, and output depth maps $\hat{D}_C$ are un-normalized using Equation 2.

Table 1.
Table 2. Depth results on the KITTI Eigen test split [2], for distances up to 80m with the Garg crop [5], evaluated on the improved depth maps from [18]. Superv. indicates the source of supervision (M for monocular self-supervision and D for depth supervision); and Multi-Fr. the use of multiple frames at test time. Monocular results are median-scaled at test time, to account for scale ambiguity. We report DepthFormer results in both half-resolution (MR, 640 × 192), and full resolution (HR, 1216 × 352), using the same training and architecture parameters (Section 4.2, main paper).

<table>
<thead>
<tr>
<th>Method</th>
<th>Superv.</th>
<th>AbsRel</th>
<th>SqRel</th>
<th>RMSE</th>
<th>RMSElog</th>
<th>δ &lt; 1.25</th>
<th>δ &lt; 1.25^2</th>
<th>δ &lt; 1.25^3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kuznietsov et al. [13]</td>
<td>D</td>
<td>0.113</td>
<td>0.741</td>
<td>4.621</td>
<td>0.189</td>
<td>0.862</td>
<td>0.960</td>
<td>0.986</td>
</tr>
<tr>
<td>Gan et al. [4]</td>
<td>D</td>
<td>0.098</td>
<td>0.666</td>
<td>3.933</td>
<td>0.173</td>
<td>0.890</td>
<td>0.964</td>
<td>0.985</td>
</tr>
<tr>
<td>Guizilini et al. [8]</td>
<td>D</td>
<td>0.072</td>
<td>0.340</td>
<td>3.265</td>
<td>0.116</td>
<td>0.934</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>DORN [3]</td>
<td>D</td>
<td>0.072</td>
<td>0.307</td>
<td>2.727</td>
<td>0.120</td>
<td>0.932</td>
<td>0.984</td>
<td>0.994</td>
</tr>
<tr>
<td>Yin et al. [22]</td>
<td>D</td>
<td>0.072</td>
<td>0.328</td>
<td>3.175</td>
<td>0.138</td>
<td>0.938</td>
<td>0.990</td>
<td><strong>0.998</strong></td>
</tr>
<tr>
<td>PackNet-SFM [7]</td>
<td>M</td>
<td>0.078</td>
<td>0.420</td>
<td>3.485</td>
<td>0.121</td>
<td>0.931</td>
<td>0.986</td>
<td>0.996</td>
</tr>
<tr>
<td>ManyDepth [21]</td>
<td>M</td>
<td>0.064</td>
<td>0.320</td>
<td>3.187</td>
<td>0.104</td>
<td>0.946</td>
<td>0.990</td>
<td>0.995</td>
</tr>
<tr>
<td>BTS [14]</td>
<td>M</td>
<td>0.059</td>
<td>0.245</td>
<td>2.756</td>
<td>0.096</td>
<td>0.956</td>
<td><strong>0.993</strong></td>
<td><strong>0.998</strong></td>
</tr>
</tbody>
</table>

DepthFormer (MR) | M ✓     | **0.055** | 0.271 | 2.917 | 0.095   | 0.955   | 0.991     | **0.998** |
DepthFormer (HR) | M ✓     | **0.055** | 0.265 | 2.723 | **0.092** | **0.959** | **0.992** | **0.998** |

Table 2. Depth results on the KITTI Eigen test split [2], for distances up to 80m with the Garg crop [5], evaluated on the improved depth maps from [18]. Superv. indicates the source of supervision (M for monocular self-supervision and D for depth supervision); and Multi-Fr. the use of multiple frames at test time. Monocular results are median-scaled at test time, to account for scale ambiguity. We report DepthFormer results in both half-resolution (MR, 640 × 192), and full resolution (HR, 1216 × 352), using the same training and architecture parameters (Section 4.2, main paper).

3. Qualitative Examples

Some examples of predicted depth maps, including common failure cases due to lack of camera motion and dynamic objects, are shown in Figures 1 and 2 for the KITTI and DDAD datasets respectively. High-response depth maps (Section 3.3.1, main paper) are masked out using our proposed low-confidence threshold. These masked out regions usually include far-away objects towards the vanishing point, including the sky, and interestingly also occluded areas and dynamic objects. Context-adjusted depth maps (Section 3.3.2, main paper) are able to reason over these low-confidence areas by conditioning with information from the target image. However, they still fail in situations where multi-frame matching is inaccurate or ill-posed (e.g., lack of camera motion or dynamic objects). By introducing single-frame features for joint decoding (Section 3.3.3, main paper), we are able to also reason over these situations and achieve our reported state-of-the-art results. Quantitative evaluation of these intermediate depth maps is provided in Table 2 of the main paper.

4. Reconstructed Pointclouds

We also show examples of reconstructed KITTI and DDAD pointclouds in Figures 3 and 4. These pointclouds are obtained by unprojecting pixel colors to 3D space using known camera intrinsics, predicted depth maps, and predicted relative motion between frames. We reiterate here that no ground-truth is used at training or inference time, only videos. Even so, our architecture is able to reconstruct the observed environment, including low-texture regions, object boundaries, and dynamic objects to a high degree of accuracy, as shown in our quantitative evaluation (Table 2). For examples of pointcloud reconstruction over entire sequences, please refer to the supplementary video.

5. Negative Impact

Because our proposed method operates on a monocular self-supervised setting, it can process arbitrarily large amounts of unlabeled visual data without human intervention. However, more does not necessarily mean better, and some amount of data curation is still desirable, to avoid the introduction of biases in trained models due to data imbalance. Another potential issue is privacy, and proper procedures should be taken when processing large quantities of data without supervision, to preserve individual anonymity.

6. Limitations

Our proposed method increases robustness to some of the common challenges found in self-supervised monocular depth estimation, such as dynamic objects and static frames, by improving feature matching across frames. However, it does not explicitly address these issues, which would require 3D motion modeling in the form of scene flow [11] or tracking [23]. Another common limitation of self-supervised monocular depth estimation is scale ambiguity, since models trained purely on image information cannot produce metrically-accurate predictions. Scale-aware results are necessary for downstream tasks that ingest our reconstructed pointclouds, such as 3D object detection [20]. Some works have addressed this limitation in the self-supervised setting by introducing weak velocity supervision [7] or additional geometric information such as camera height [19] or multi-camera extrinsics [9]. Our proposed method does not address this issue, however it can directly benefit from these works to produce scale-aware estimates.
Figure 1. Qualitative depth estimation results of our proposed DepthFormer architecture, on the KITTI dataset.
Figure 2. Qualitative depth estimation results of our proposed DepthFormer architecture, on the DDAD dataset.
Figure 3. **Pointcloud reconstructions** obtained using our DepthFormer architecture, on the KITTI dataset.
Figure 4. **Pointcloud reconstructions** obtained using our DepthFormer architecture, on the DDAD dataset.
References