Refign: Align and Refine for Adaptation of Semantic Segmentation to Adverse Conditions

Supplementary Material

David Bruggemann Christos Sakaridis Prune Truong Luc Van Gool
ETH Zurich, Switzerland
{brdavid, csakarid, truongp, vangool}@vision.ee.ethz.ch

A. Mathematical Derivations

**Derivation of Log-Likelihood Loss** We model the likelihood with an uncorrelated, bivariate Gaussian with mean $\hat{F} = [\hat{F}_u, \hat{F}_v]^T$ and variance $\hat{\Sigma} = \hat{\Sigma}^u = \hat{\Sigma}^v$ for flow directions $u$ and $v$.

$$L_{I' \rightarrow I}^{prob} = -\log p(W|I, I') = -\log \left( \frac{1}{\sqrt{2\pi\hat{\Sigma}^u_{I' \rightarrow I}}} e^{-\frac{1}{2\hat{\Sigma}^u_{I' \rightarrow I}} (\hat{F}_u_{I' \rightarrow I} - W^u)^2} \cdot \frac{1}{\sqrt{2\pi\hat{\Sigma}^v_{I' \rightarrow I}}} e^{-\frac{1}{2\hat{\Sigma}^v_{I' \rightarrow I}} (\hat{F}_v_{I' \rightarrow I} - W^v)^2} \right)$$

$$= -\log \left( \frac{1}{2\pi\hat{\Sigma}_{I' \rightarrow I}} \left\| \hat{F}_{I' \rightarrow I} - W \right\|^2 \right) + \log \hat{\Sigma}_{I' \rightarrow I}$$

$$\propto \frac{1}{2\hat{\Sigma}_{I' \rightarrow I}} \left\| \hat{F}_{I' \rightarrow I} - W \right\|^2 + \log \hat{\Sigma}_{I' \rightarrow I}$$

Derivation of Confidence Map We integrate the bivariate Gaussian density function over a circle with radius $r$ (subscripts are omitted).

$$P_R = p(\left\| \hat{F} - \hat{F} \right\| \leq r)$$

$$= \int_0^{2\pi} 1 r e^{-\frac{1}{2\hat{\Sigma}} r^2 \rho \phi} d\rho d\phi$$

$$= 1 - \exp \frac{-r^2}{2\Sigma}$$

B. Training Details

In this section, we describe training settings and implementation details. Both alignment and segmentation networks were trained using Automatic Mixed Precision on a single consumer RTX 2080 Ti GPU.

**B.1. Alignment Network**

UAWarpC training almost exactly follows the setup of [27]. The training consists of two stages: In the first stage, the network is trained without the visibility mask, as the visibility mask estimate is still inaccurate. In the second stage, the visibility mask is activated and more data augmentation is used.

**Data Handling** The alignment network is trained using MegaDepth [13], consisting of 196 scenes reconstructed from 1,070,468 internet photos with COLMAP [22]. 150 scenes are used for training, encompassing around 58,000 sampled image pairs. 1800 image pairs sampled from 25 different scenes are used for validation. No ground-truth correspondences from SfM reconstructions are used to train UAWarpC.

During training, the image pairs $I, J$ are resized to 750×750 pixels, and a dense flow $W$ is sampled to create $I'$. Finally, all three images $I, J, I'$ are center-cropped to resolution 520×520. In the first training stage, $W$ consists of sampled color jitter, Gaussian blur, homography, TPS, and affine-TPS transformations. In the second stage, local elastic transformations are added, and the strength of the transformations is increased. For the detailed augmentation parameters, we refer to [27].

**Architecture and Loss Function** Again following [27], a modified GLU-Net [25] is used as a base architecture for flow prediction. GLU-Net is a four-level pyramidal network with a VGG-16 [23] encoder. The encoder is initialized with ImageNet weights and frozen. GLU-Net requires an additional low-resolution input of 256×256 to establish global correlations, followed by repeated levels of upscaling and local feature correlations. As in [27], our flow decoder uses
Data Handling  Input images are resized to half resolution for Cityscapes [3], ACDC [20], and Dark Zurich [21]. For RobotCar Correspondence [17, 10] and CMU Correspondence [11,10], we resize to 720x720 and 540x720, respectively. Data augmentation consists of random cropping to 512x512 and random horizontal flipping. For the coarsely labeled extra target images in the semi-supervised domain adaptation for RobotCar and CMU, we additionally apply random rotation with maximum 10° and color jittering.

Optimization Schedule  We use the AdamW [15] optimizer with a weight decay of 0.01. The learning rate follows a linear warmup for 1500 steps, followed by linear decay. The peak learning rate is $6 \cdot 10^{-4}$. On ACDC and Dark Zurich, we train for 40k iterations; on RobotCar and CMU, we train for 20k iterations. A batch size of 2 is used throughout.

To mitigate the risk of overfitting, we use the coarsely labeled extra target images in semi-supervised domain adaptation on RobotCar and CMU only in every second training iteration.

C. Small vs. Large Static Classes

To motivate the distinction between small and large static classes (as defined in Sec. 4), we generate ACDC [20] reference image predictions using a SegFormer [33] trained on Cityscapes [3], and warp them onto the corresponding adverse-image viewpoint. As shown in Fig. C-1, we observe a correlation between the resulting IoU and the average size of the connected class component for static classes (pearson correlation coeff. of 0.70). The classes pole, traffic light, and traffic sign are drastically smaller than the rest, and consequentially have lower accuracy. On the other hand, such indiscriminate warping (i.e., without $P_R$) is surprisingly accurate for the large static classes.

Furthermore, we analyze the mIoU improvement when only considering pixels above a certain $P_R$ threshold for the above mentioned warped SegFormer predictions, see Fig. C-2. While the performance increases monotonically for both dynamic and small static classes, it remains mostly flat for large static classes. This suggests that large static classes are largely insensitive to the warping confidence, while both dynamic and small static classes benefit greatly from confidence guidance.

D. Additional Experimental Results

Due to space restrictions, we present the full class-wise performances of state-of-the-art UDA methods on Dark Zurich-test here in Table D-1. The models reported in Tables 1,2 and D-1 all use the same image input size at test-time for fairness of comparison. Table D-2 presents models which do not follow that protocol. Using Cityscapes-pretrained weights for initialization, Refign added on top of
Refign-DAFormer currently ranks first on public leaderboards. Refign-HRDA [8] achieves 72.1 mIoU and 63.9 mIoU on ACDC and Dark Zurich-test, respectively, ranking first on the public leaderboards of these benchmarks at the time of publication. In Table D-3, we report the performance of Cityscapes→ACDC Refign-DAFormer on the four different conditions of the ACDC validation set. Refign improves markedly over the baseline for all conditions.

We also compare the Cityscapes→ACDC Refign-DAFormer model with state-of-the-art foggy scene understanding methods in Table D-4. All methods are trained with Cityscapes as source domain, however the foggy scene understanding methods utilize both synthetic foggy data and a larger pool of real foggy data as targets. Surprisingly, our model achieves state-of-the-art performance despite this handicap.

Finally, we conduct experiments substituting the SegFormer [45] based architecture of DAFormer [7] with DeepLabv2 [2]. On both ACDC and Dark Zurich validation sets, this version of Refign improves substantially over the baseline, as reported in Table D-5.

E. Refign at Test-Time

Although designed to refine pseudo-labels during online self-training, Refign can also be applied at test-time to ar-
Table D-5. Performance of Refign vs. DAFormer baseline with a DeepLabv2 model on the ACDC and Dark Zurich validation sets.

<table>
<thead>
<tr>
<th>Method</th>
<th>ACDC</th>
<th>Dark Zurich</th>
</tr>
</thead>
<tbody>
<tr>
<td>DAFormer (DeepLabv2)</td>
<td>56.6</td>
<td>38.0</td>
</tr>
<tr>
<td>Refign-DAFormer (DeepLabv2)</td>
<td>58.5</td>
<td>39.1</td>
</tr>
</tbody>
</table>

Table E-1. Applying Refign only for one refinement iteration at test-time to DAFormer on the ACDC and Dark Zurich validation sets.

<table>
<thead>
<tr>
<th>Method</th>
<th>ACDC</th>
<th>Dark Zurich</th>
</tr>
</thead>
<tbody>
<tr>
<td>DAFormer [7]</td>
<td>55.6</td>
<td>34.1</td>
</tr>
<tr>
<td>DAFormer + Test-Time Refign</td>
<td>56.8</td>
<td>38.0</td>
</tr>
</tbody>
</table>

We present a method to adapt existing semantic segmentation models to new domains, even though we restrict ourselves to adverse-condition autonomous driving in this paper, our algorithm could potentially be used in more undesired applications, such as surveillance or military. This risk of potential misuse exists for all semantic segmentation algorithms, and could be mitigated through appropriate legislation.

F. Qualitative Results

We show more qualitative results in this section. Fig. F-1 shows the warps and corresponding confidence maps for randomly selected ACDC samples. In Fig. F-2, we show some warp failures. Importantly, the confidence map correctly blends out the inaccurate warps. Finally, Fig. F-3 shows more qualitative segmentation results for randomly selected ACDC validation samples.

G. Potential Negative Societal Impact

We present a method to adapt existing semantic segmentation models to new domains. Even though we restrict ourselves to adverse-condition autonomous driving in this paper, our algorithm could potentially be used in more undesired applications, such as surveillance or military. This risk of potential misuse exists for all semantic segmentation algorithms, and could be mitigated through appropriate legislation.

References


Figure F-1. Example visualizations of warped reference images and the corresponding confidence maps from ACDC.
Figure F-2. Warp failure examples on ACDC.


|-------|----------|---------------------|--------------|-----------------|--------------|

Figure F-3. Prediction samples of the ACDC validation set.