Domain Decluttering: Simplifying Images to Mitigate Synthetic-Real Domain Shift and Improve Depth Estimation *–Supplementary Material–*

Yunhan Zhao[1] Shu Kong[2] Daeyun Shin[1] Charless Fowlkes[1]
[1]UC Irvine [2]Carnegie Mellon University
{yunhaz5, daeyuns, fowlkes}@ics.uci.edu shuk@andrew.cmu.edu

In the supplementary material, we first provide qualitatively visualizations of Kitti dataset, including analyses of improving and failure cases. Second, we plot the curve of sparsity level ρ vs. absolute relative difference (Abs Rel) on Kitti dataset. Third, we present detailed training diagrams of how we initialize/pretrain the inpainting module $\mathcal{I}$ and attention module $\mathcal{A}$ in our modular coordinate descent algorithm. Lastly, we provide an additional figure to study per-sample improvement with sorted RMSE-log error. Some additionally qualitatively visualizations of NYUv2 dataset are shown at the end.

1. Qualitatively Visualizations of Kitti

We present qualitative results of ARC on Kitti dataset. Similar to Fig. 5 in the paper, we show examples over which our ARC improves the depth prediction or makes worse predictions (as failure case).

From Fig. 1, we observe that the attention module $\mathcal{A}$ attempts to mask out the sky, pavements, and overexposured areas in both improvements and failure cases. In failed cases, we find a pattern in some images that the sky and pavements are connected (*e.g.*, due to overexposure). Under such condition, the attention module is very likely to remove them together and the original *vanishing point* cannot be reliably inferred, we believe important to estimate the depth in Kitti images. Recall that Kitti images have similar structures as the car is moving forward and the vanishing point is around the image center in most images.

It is worth noting that in real training images of Kitti dataset, the depth annotations are very sparse (due to LiDAR sensor) or missing in sky regions. So it is reasonable that the model learns to remove sky regions in a lazy way as there is no penalty from the depth loss on the sky region. Moreover, interestingly, the ARC model learns to paste "green trees" to the removed regions. We conjecture that the green trees are large regions besides the road and reliable cues to estimate vanishing point and thus better depth prediction.

2. (*Cont'd*) Study of Sparsity Level ρ

Here, We show the curve of how sparsity level ρ affects the performance of ARC on Kitti dataset. As shown in Fig. 2, the trend of the curve on Kitti dataset is similar to the curve plotted on NYUv2 dataset, but the slope is quite different. The performance changes very slightly with different sparsity level ρ. Considering the LiDAR depth map is sparse and there is no depth annotation in the sky regions of Kitti images, we believe this behavior matches our expectations. As shown in the previous section, the attention module $\mathcal{A}$ is likely to focus on the sky or pavement. As there is little supervision (no depth in sky regions) and large plain regions (*e.g.*, road), removing pixels from these regions to different extents does not significantly affect the overall depth prediction.

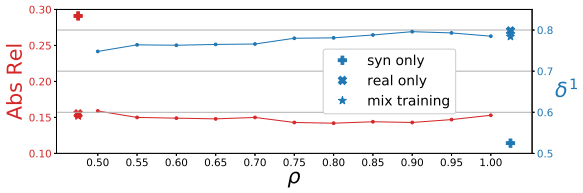

Figure 2: Study of the sparsity factor ρ of the ARC model on Kitti dataset.

With the comprehensive study of sparsity hyper-parameter ρ from both Fig. 3 in the paper and Fig. 2, we see that the performance drops when the sparsity level ρ increases from 0.95 to 1.0 (strictly speaking, $\rho = 0.99999$[1]). Decrements in performance show that learning to remove a reasonably portion of pixels, *e.g.*, $\rho = 0.90$ or $\rho = 0.95$, indeed helps improve depth prediction. Note that although $\rho = 1.0$ expects no sparse attention map output from the attention module, we observe training with $\rho = 1.0$ achieves better performance than simply training without attention

[1]One cannot set $\rho = 1.0$ exactly due to the KL loss.

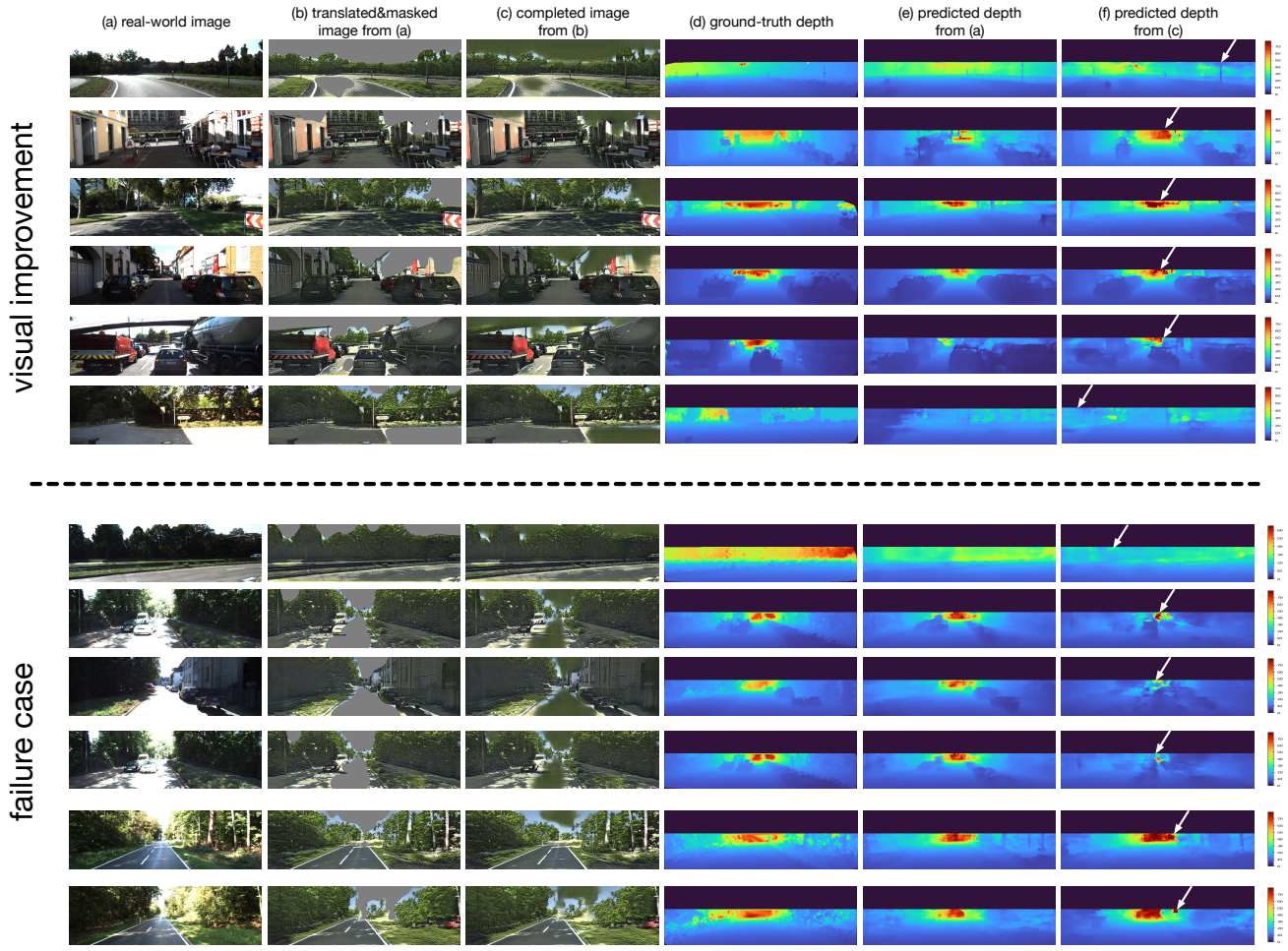

Figure 1: Qualitative visualizations of our ARC on Kitti dataset including improvements as well as failure cases. White arrows in the last column are used to highlight the regions over which the model improves or degrades visibly w.r.t depth prediction. We use the same color bar for the visualizing depth in each row. (Best view in color and zoomed in.)

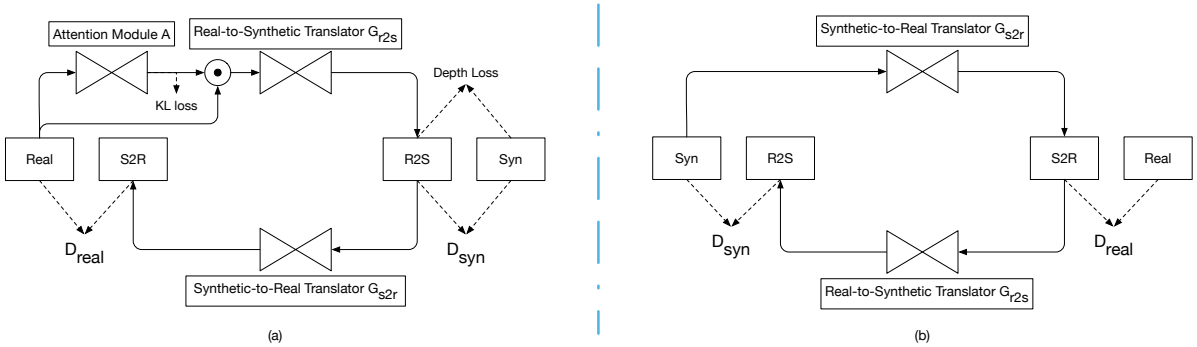

Figure 3: Detailed training diagram of our attention module $\mathcal{A}$. Note that $\mathcal{A}$ only shows in the real-to-synthetic cycle, e.g., the part (a) in the diagram. The intuition behind two asymmetric cycles is that $\mathcal{A}$ should remove clutters in real samples instead of clean synthetic images.

modules. We believe one possible reason behind this observation is that learning $\mathcal{A}$ before its convergence during training still introduce pixel/region removal, which leads to more robust training as studied in literature [1].

3. Detailed Training Diagrams of $\mathcal{A}$ and $\mathcal{I}$

To provide a clear idea of how we (pre)train our modules, we present two diagrams, the attention module $\mathcal{A}$ and inpainting module $\mathcal{I}$. For others, we train the real-to-synthetic style translator $\mathcal{T}$ by simply using the CycleGAN pipeline [2]. To train the depth predictor module $\mathcal{D}$, we train it simply using depth regression loss.

The training diagram of our attention module $\mathcal{A}$ is presented in Fig. 3. Note that $\mathcal{A}$ only appears in the left panel Fig. 3 (a), which means $\mathcal{A}$ only learns where to mask out in real images. We do not apply this to synthetic data, as synthetically rendered images are clean without clutters.

Our detailed training diagram of module $\mathcal{I}$ is shown in Fig. 4. The attention module $\mathcal{A}$ and the style translator $\mathcal{T}$ are pretrained models and we color them in red for the purpose of indication. Note that the output of $\mathcal{I}$ is the intermediate inpainting results and our final reconstructed images still follow Eqn.(3) in the paper.

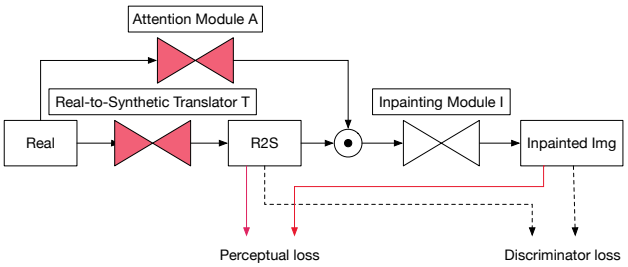

Figure 4: Detailed training diagram of our inpainting module $\mathcal{I}$. Red blocks in the figure indicates that they are pretrained modules, *i.e.*, the attention module $\mathcal{A}$ and style translator $\mathcal{T}$.

4. Additional Per-Sample Improvement Study

We compute the per-image prediction of ARC and the mix training baseline on NYUv2 testing set w.r.t RMS-log and plot the result by sorting the baseline performance. As shown in Fig. 5, ARC reduces errors over a majority of samples as there are more red dots below the blue curve visually. This observation matches the result shown in Fig. 4 in the paper, where experimentally proves ARC reduces error for around 70% of the sample in the dataset. More importantly, ARC reduces error even more when the mix training baseline has larger prediction errors, which demonstrates the effectiveness of removing "hard" pixels.

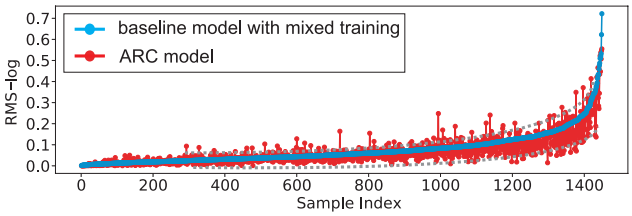

Figure 5: Per-sample improvement of ARC and the mix training baseline on NYUv2 testing set w.r.t RMS-log.

References

[1] Golnaz Ghiasi, Tsung-Yi Lin, and Quoc V Le. Dropblock: A regularization method for convolutional networks. In *Advances in Neural Information Processing Systems*, pages 10727–10737, 2018. 3

[2] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2223–2232, 2017. 3

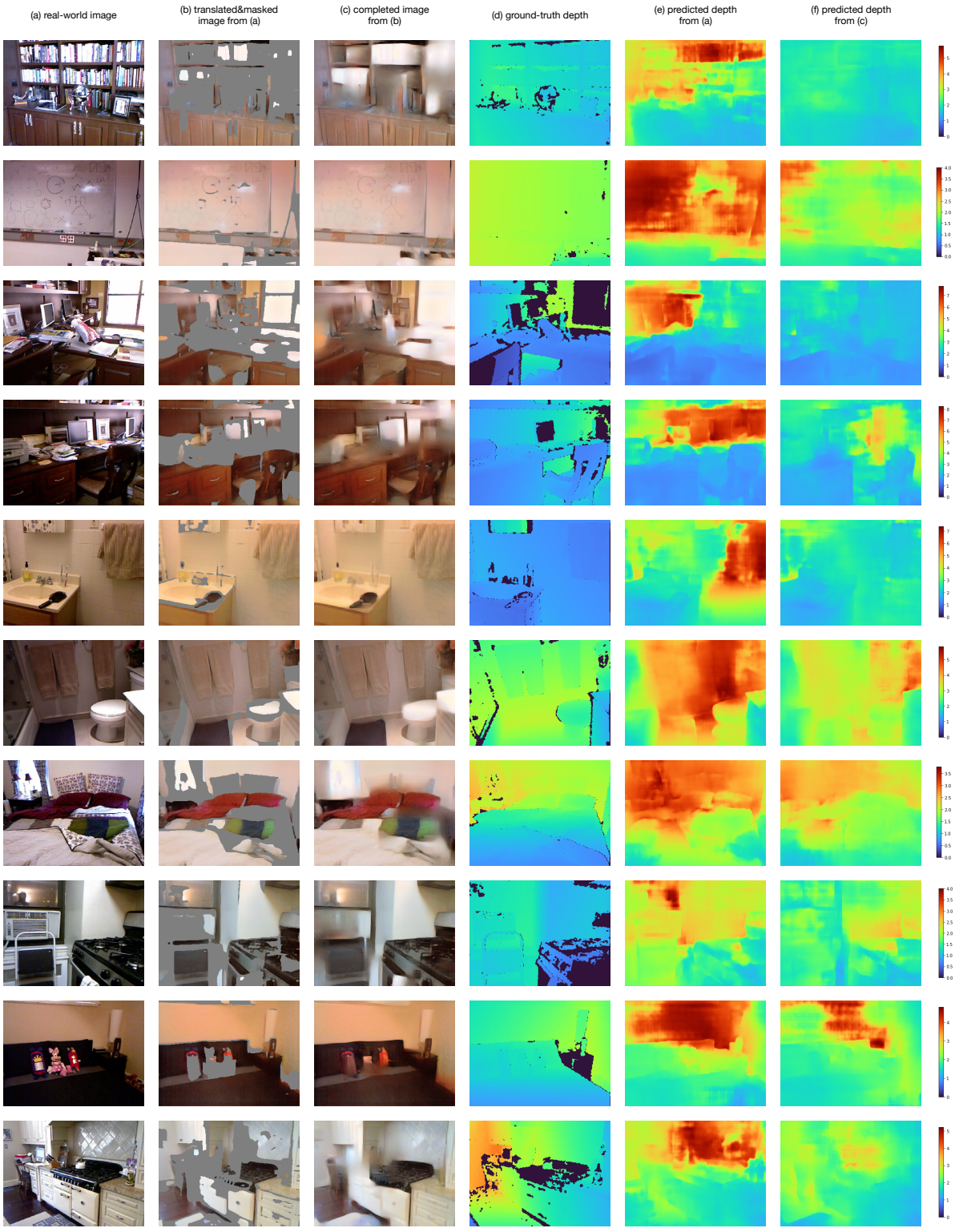

Figure 6: Additional qualitative visualizations of our ARC on NYUv2 dataset. (Best viewed in color and zoomed in.)