Warp-Refine Propagation: Semi-Supervised Auto-labeling via Cycle-consistency – Supplementary Material –

Aditya Ganeshan[1*] Alexis Vallet[2] Yasunori Kudo[2] Shin-ichi Maeda[2] Tommi Kerola[2]
Rareș Ambruș[3] Dennis Park[3] Adrien Gaidon[3]
[1]Brown University [2]Preferred Networks, Inc. [3]Toyota Research Institute (TRI)

1. Introduction

In this document, we provide additional details and experimental results to help further understand and reproduce our proposed method, i.e. *Warp-Refine Propagation*. The supplementary material is divided into the following sections:

- **Section 2 - Additional experimental studies**: We explore different aspects of training with propagated labels, such as training with different architectures, and training in data scarce setting.
- **Section 3 - Additional details**: We provide details for our training setup, as well as details of ApolloScape dataset [10] usage.
- **Section 4 - Qualitative results**: We provide qualitative examples visualizing the results of different aspects of our method.

2. Additional experimental studies

2.1. Propagation for different architectures

We evaluate the benefit of the propagated labels on different segmentation models. This result is summarized in Table 1. We see that the propagated labels are significantly beneficial for smaller architectures, which have lower performance. However, in the case of motion-only propagated labels, we see that the performance is unaffected or sometimes deteriorated. Note that these results do not use the 20000 additional coarse labels, nor Mapillary Vistas [5] pre-training.

2.2. Motion estimation model ablation

A simple way to use geometric cues is to simply warp the labels between consecutive frames based on the Optical flow. We tried different optical flow methods including RAFT [9] for warping, but found them to be unsuitable. Apart from drifting errors, directly warping with optical flow also causes content duplication on de-occluded regions [14]. Therefore, for warping labels between consecutive frames, we found video prediction to work the best for us.

2.3. Ablative analysis with motion-only labels

As indicated in the main paper, we do not train using the Relaxed Label Loss (RLL) proposed by Zhu et al. [16], and also use a fixed epoch-size. In this section, we provide additional ablative experiments, validating our choices. Our results are summarized in Table 2, along with the numbers reported by Zhu et al. [16] under similar training conditions. Note that we add the *motion-only* propagated labels at time step $t{\pm}3$ as represented by D^m_3. We perform the experiments under different training settings, namely considering the usage of coarse-labels and Mapillary Vistas pre-training. We report the mean and the standard deviation by conducting three runs with different random number generator seeds for each result. We note a significant improvement between our baseline when training with the Cityscapes coarse-labels and with Mapillary Vistas pre-training (80.94 *mIoU*) and the baseline reported in [16] (79.46 *mIoU*) which we attribute to a longer training schedule of our baseline and a modified learning rate schedule:

1. *Longer training of the baseline*: When training with propagated labels, our dataset size for $\mathtt{B} + D^m_3$ is increased. This leads to more training iterations for $\mathtt{B} + D^m_3$ with respect to the baseline model ($\mathtt{B}$ is trained for only one-third the iterations of $\mathtt{B} + D^m_3$). We therefore modify the training such that $\mathtt{B}$ is trained for the same number of iterations as $\mathtt{B} + D^m_3$.
2. *Higher learning rate for the baseline*: We increase the learning rate by a factor of 8 for the baseline $\mathtt{B}$ (we observed that the scale of cross-entropy loss is much smaller than the scale of Relaxed Label Loss).

With the updated baseline, we find RLL as well as training with *motion-only* labels to be ineffective. Further, to avoid the pitfall of under-training the baseline, we fix the epoch-size for all the models we compare. This ensure that the

*Work done while A. Ganeshan was at Preferred Networks, Inc.

Table 1: Training with different labelling policies on Cityscapes [3] val-split: We evaluate the benefit from warp-refine propagation across different segmentation models. Due to the lack of semantic complexity in the dataset (only 19 classes), and the high performance (*mIoU* = 83.35) of the semantic labelling network, we find the *semantic-only* labels to give significant benefits as well. (Note that for *motion-only* we utilize only time-frames $\pm$[2] as recommended by the authors Zhu et al. [16]). We report the average of three independent runs with different random seeds. Note that we do not use any additional data (coarse labels and Mapillary Vistas [5] pretraining) for this ablative analysis.

Model	Backbone	Baseline	*motion-only*	*semantic-only*	*warp-refine*
DeepLab V3 [2]	ResNeXt-50 [12]	79.26	79.01	80.45	**80.68**
OCR Net [13]	ResNeXt-50 [12]	79.55	79.60	**80.89**	80.80
MSA-HRNet-OCR [8]	HRNet-W48 [7]	83.35	83.00	83.91	**84.07**

Table 2: Results of training a segmentation model with labels generated by video propagation [16] (D_3^m) and relaxed label loss (RLL), on the Cityscapes validation split. These experiments are conducted under different training settings (as shwon by the top two rows) . We also compare the mean IoU to those reported in previous work [16]. We conduct three runs with different random seeds.

Coarse Labels / Map. Pre-train	✗ / ✗		✓ / ✗		✓ / ✓		Training iterations
	avg. mIoU	std.	avg. mIoU	std.	avg. mIoU	std.	#
[16] baseline B	-	-	-	-	79.46	-	175×2975
[16] B + RLL	-	-	-	-	80.85	-	175×2925
[16] B + RLL + D_3^m	-	-	-	-	81.35	-	175×8925
Baseline B	77.66	0.27	79.15	0.23	80.94	0.10	175×8925
B+ RLL	77.50	0.12	-	-	80.76	0.18	175×8925
B + RLL + D_3^m	77.41	0.22	78.69	0.20	80.8	0.11	175×8925

improvement by using additional labels is not conflated with improvement by longer training.

3. Additional details

3.1. Training details

We use an SGD optimizer and employ a polynomial learning rate policy, where the initial learning rate is multiplied by $(1-\frac{\texttt{epoch}}{\texttt{max epoch}})^{\texttt{power}}$. The learning rate is varied for different datasets: for KITTI [1] we utilize a learning rate of 0.0005, for Cityscapes we utilize 0.01 and for NYU-V2 [4] we utilize 0.001. Momentum and weight decay are set to 0.9 and 0.0001 respectively. We use synchronized batch normalization (batch statistics synchronized across all GPUs) with the batch distributed over 8 V100 GPUs. For data augmentation, we randomly scale the input images (from 0.5 to 2.0), and apply horizontal flipping, Gaussian blur and color jittering during training. Further, we utilize uniform sampling [16] across semantic classes with 50% of each epoch.

We introduce two changes from the training configuration outlined by Zhu et al. [16]:

- As our approach generates additional training data, the epoch size varies greatly depending on training settings. This can lead to a situation where the observed improvement in performance can be due to longer training rather than generated data (As shown in Section 2.3). To avoid such mis-attribution of the reason for improvement, we ensure that the training regime for all compared experiments is equivalent. To achieve that, we define an epoch to have a fixed size (roughly $3\times$ the size of the normal dataset). With this definition, we train for 175 epochs.
- We adjust our data sampling such that in each epoch, 30% samples are drawn from the manually annotated dataset, and 70% data is drawn from the generated dataset (through label propagation). Hence, the number of pseudo-labels considered per epoch remains consistent independent of the amount of generated labels (In the presence of Coarse labelled data, we reduce sampling from the generated dataset to 30%).

For models evaluated on the test set, we use the same training validation split used by Zhu et al. [16] (`cv2` split). The cities Mönchengladbach, Strasbourg and Stuttgart are used as validation set while all the others are used as training data.

3.2. ApolloScape partitioning

The ApolloScape dataset [10] contains pixel-level annotations for sequentially recorded images, divided as 40960 training and 8327 validation images. These images are further broken into the subsets based on the road on which they were recorded, and the Record-ID. Each Record-ID consists of variable length sequentially annotated frames. We break these sequentially annotated frames into partitions each consisting of 21 consecutive frame. The images which are not a part of any such 21-frame partition (for example when a Record-ID contains less than 21 frames) are discarded.

Now, from each partition, we utilize the central frame as a training data point (i.e. with manual annotation) and all the other frames are treated as frames where labels have to be generated via propagation. This allows us to create a dataset with ground-truth labels containing 2005 frames, and additional 40100 sequential images (we only use the provided ground truth for these images for evaluation purposes).

Note that to ensure that training and validation data do not have any overlap (which could happen if any partition of 21 frames contains validation samples), we combine the training and validation subset, and re-divide it at a Record-ID level (randomly). This ensures that none of our train-sequences have any overlap with the validation data. Due to this our training and validation split are different from the one provided with the dataset. To encourage and facilitate comparisons with our work, we will release our training and validation splits to the community.

3.3. Denoising module

Our denoising module Ω_λ is inspired from semantic-to-real models [11, 6]. We show our architecture in Figure 2. Our network takes the warp-inpainted labels L_t^w, along with auxiliary inputs: the warped image I_t^m, and the image at time $t+1$ I_{t+1} to generate refined labels L_{t+1}^R:

$$L_{t+1}^R = \Omega_\lambda(I_{t+1}, I_t^w, L_t^w) \tag{1}$$

The warped labels l_t^w are used as one-hot vectors per pixel. All the inputs are concatenated along the "channel" dimension and provided to the encoder network $N_{encoder}$. The generated encoding is then concatenated with OCR-features [13] of the image I_{t+1} (extracted using the baseline model g_ψ trained with only manually annotated images). This is done to provide rich semantic cues for regions with new objects. Finally the concatenated encoding is passed through the decoder network $N_{decoder}$ to generate the refined labels L_{t+1}^R. The complete pipeline is visualized in Figure 2.

Our network is trained with the same optimization setting as detailed in Section 3.1. The RMI loss [15] is used to compute the cycle-consistency loss $\mathcal{L}(L_t, L_t^R)$.

4. Qualitative results

In Figure 1, we show examples of cyclic warped labels l° (cf. Section 3.2 in the main paper) for different cycle lengths. As shown, by using different cycle lengths we are able to expose the denoising module Ω_λ to a larger variety of label noise created due to warp-inpaint propagation. Figure 3 compares the output of model trained with and without *warp-refine* labels on KITTI [1] test-split (and nearby images using scene-flow test-split). We observe that on training with *warp-refine* labels, improves the networks performance on confusing classes such as (i) bus-truck, (ii) truck-car, (iii) rider-pedestrian, and (iv) fence-wall.

Finally, Figure 4 shows additional qualitative comparisons between our propagation method and established baselines: i) *motion-only* labels [16], and ii) *semantic-only* labels [8]. (a)-(d) show cases where our approach surpasses the other methods significantly. We also highlight the errors we observe in our method: 1) Our labels are weak for fine edges, 2) Our labels still appear to show some warping noise (as shown in example (f)) and 3) Our labels can sometimes mislabel some classes (as shown in example (e)). Note that examples in Figure 4 are generated with DeepLabv3 (ResNeXt-50) [2, 12] architecture for g_ψ.

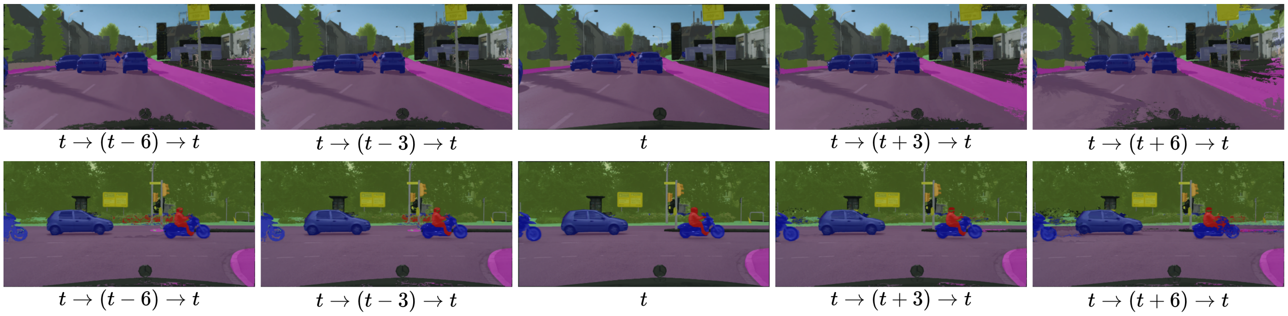


Figure 1: We show examples of cyclic warped labels, generated to train the denoising network Ω_λ. The network is trained to map the samples $(t \rightarrow t+p \rightarrow t)$ to the ground truth label (t). Using longer cycle of propagation (higher p) allows us to expose the network Ω_λ to larger amount of warping noise.

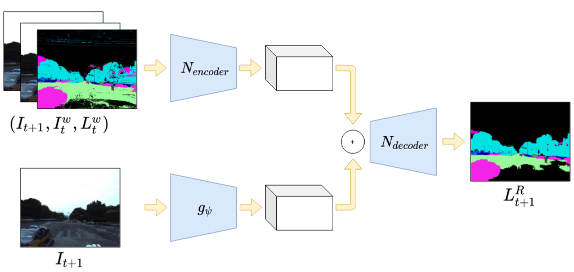
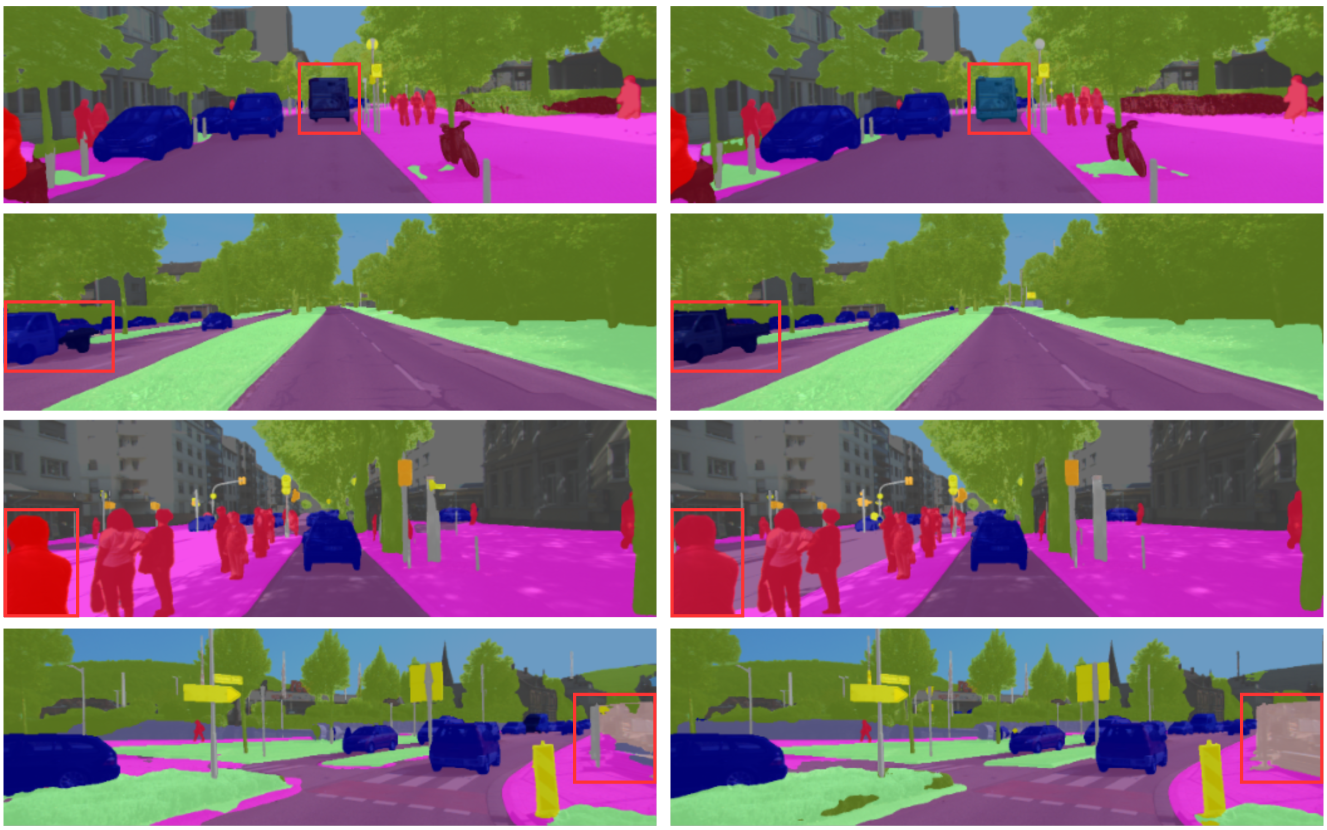


Figure 2: Architecture of the denoiser: The encoder and decoder are based on pix2pix [11]. g_ψ is the baseline model trained only with manually annotated labels. The input to the encoder are concatenated along the channels dimension. Similarly, the input of the decoder is the concatenated output of the encoder, and OCR-features [13] from g_ψ.



(a) Trained without warp-refine labels. (b) Trained with warp-refine labels.

Figure 3: Qualitative comparison of model trained with and without warp-refine labels. We see that training with warp-refine labels increase performance for confusing classes: Baseline model mis-predicts (i) 'bus' as 'truck', (ii) 'truck' as 'car', (iii) 'pedestrian' as 'rider', and (iv) 'fence' as 'wall' and 'sidewalk'.

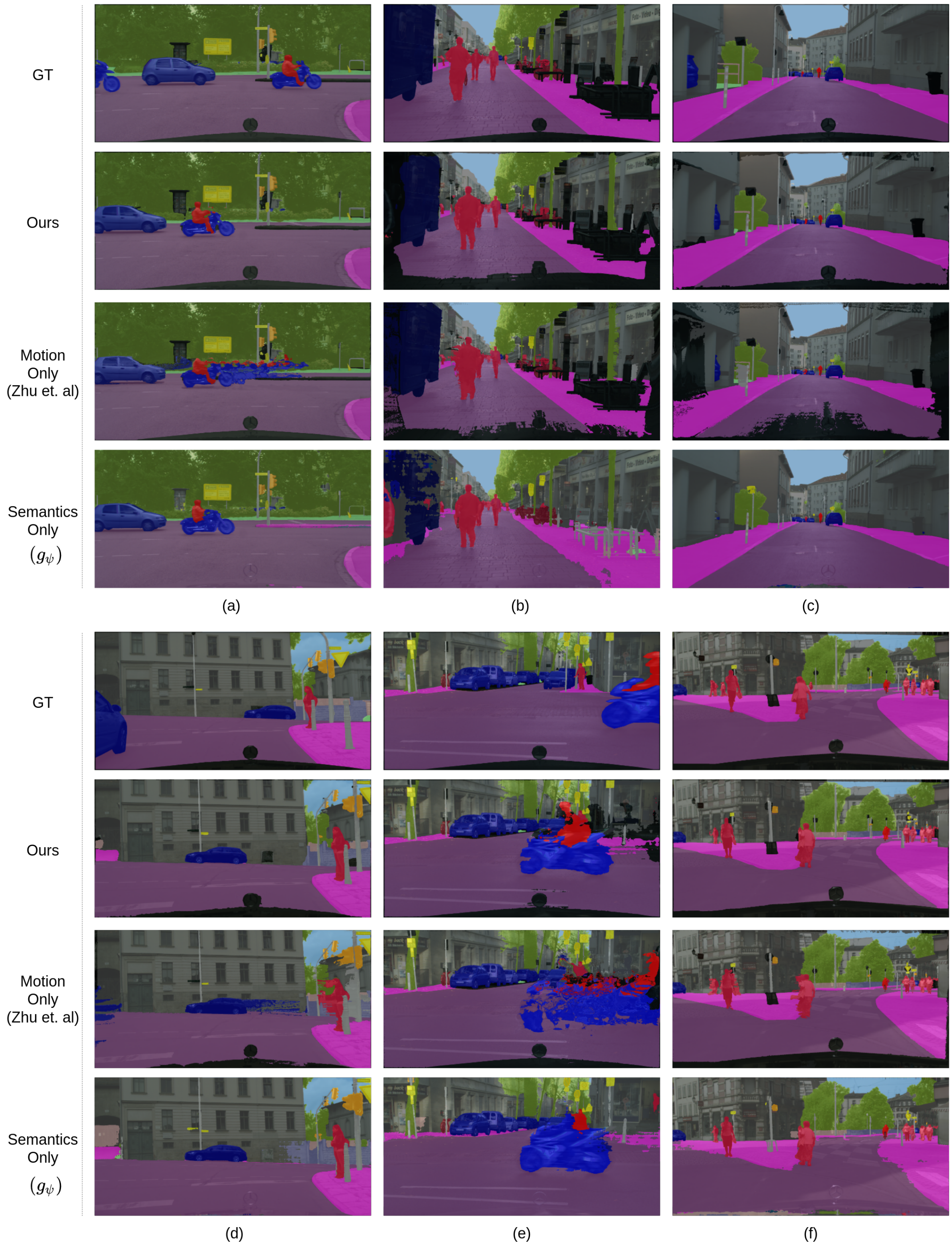


Figure 4: (a) g_ψ mislabels the Rider's legs, and motion only (Zhu et al. [16]) shows heavy drifting. (b) g_ψ mislabels the truck, and Zhu et al. [16] cause drifting near the pedestrian pixels. (c) g_ψ mislabels thin objects like poles (left side) (d) g_ψ mislabels part of the building. (e) We note that when both semantic and motion cues fail, our method fails as well. (f) Our method outputs slightly warped labels when the consecutive frames do not contain any ego-motion (note the warping of the pole in the center).

References

[1] Hassan Alhaija, Siva Mustikovela, Lars Mescheder, Andreas Geiger, and Carsten Rother. Augmented reality meets computer vision: Efficient data generation for urban driving scenes. *International Journal of Computer Vision (IJCV)*, 2018. ii, iii

[2] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *The European Conference on Computer Vision (ECCV)*, September 2018. ii, iii

[3] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. ii

[4] Pushmeet Kohli Nathan Silberman, Derek Hoiem and Rob Fergus. Indoor segmentation and support inference from rgbd images. In *ECCV*, 2012. ii

[5] Gerhard Neuhold, Tobias Ollmann, Samuel Rota Bulò, and Peter Kontschieder. The mapillary vistas dataset for semantic understanding of street scenes. In *International Conference on Computer Vision (ICCV)*, 2017. i, ii

[6] Taesung Park, Ming-Yu Liu, Ting-Chun Wang, and Jun-Yan Zhu. Semantic image synthesis with spatially-adaptive normalization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019. iii

[7] Ke Sun, Yang Zhao, Borui Jiang, Tianheng Cheng, Bin Xiao, Dong Liu, Yadong Mu, Xinggang Wang, Wenyu Liu, and Jingdong Wang. High-resolution representations for labeling pixels and regions, 2019. ii

[8] Andrew Tao, Karan Sapra, and Bryan Catanzaro. Hierarchical multi-scale attention for semantic segmentation, 2020. ii, iii

[9] Zachary Teed and Jia Deng. Raft: Recurrent all-pairs field transforms for optical flow. In *European conference on computer vision*, pages 402–419. Springer, 2020. i

[10] Peng Wang, Xinyu Huang, Xinjing Cheng, Dingfu Zhou, Qichuan Geng, and Ruigang Yang. The apolloscape open dataset for autonomous driving and its application. *IEEE transactions on pattern analysis and machine intelligence*, 2019. i, ii

[11] Ting-Chun Wang, Ming-Yu Liu, Jun-Yan Zhu, Andrew Tao, Jan Kautz, and Bryan Catanzaro. High-resolution image synthesis and semantic manipulation with conditional gans. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018. iii, iv

[12] Saining Xie, Ross Girshick, Piotr Dollar, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017. ii, iii

[13] Yuhui Yuan, Xilin Chen, and Jingdong Wang. Object-contextual representations for semantic segmentation. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *Computer Vision – ECCV 2020*, pages 173–190, Cham, 2020. Springer International Publishing. ii, iii, iv

[14] Shengyu Zhao, Yilun Sheng, Yue Dong, Eric I-Chao Chang, and Yan Xu. Maskflownet: Asymmetric feature matching with learnable occlusion mask. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. i

[15] Shuai Zhao, Yang Wang, Zheng Yang, and Deng Cai. Region mutual information loss for semantic segmentation. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32, pages 11117–11127. Curran Associates, Inc., 2019. iii

[16] Yi Zhu, Karan Sapra, Fitsum A. Reda, Kevin J. Shih, Shawn Newsam, Andrew Tao, and Bryan Catanzaro. Improving semantic segmentation via video propagation and label relaxation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. i, ii, iii, vi