Semi-supervised Multi-task Learning for Semantics and Depth Supplementary Material

In this supplementary material, we present the training procedure of the baseline algorithm and additional implementation details in the experiments. We also provide the quantitative analysis of our method and more qualitative results one the street-view and remote sensing datasets.

1. Joint Training Baseline

We describe the joint training baseline (JTL) [4] in details, which proposes to update network parameters after observing sufficient task-specific samples with ground-truths. Since we have two dense labeling tasks, *i.e.*, semantic segmentation and depth estimation, we extract the annotated mini-batches for each task to compute the per-task loss and back-propagate the gradients. After this interleaving forward and backward process, we update both the task parameters for task-specific decoders and the cumulative parameters for the shared encoder. The iterative joint training method is illustrated in Algorithm 1, where $\mathcal{L}_{gt}^{t}$ denotes the supervised loss for each task t and w_t weighs the importance of each task-specific loss.

Algorithm 1 Procedure of the joint training baseline.

for iteration $i = 1$ to N **do**
  for dataset $k \in \{1, \ldots, K\}$ **do**
    {construct mini-batch}
    for task $t = 1, \ldots, |\mathcal{T}|$ with ground-truth **do**
      {gradients for E and F_t}
      $\mathbf{L}_G^t \leftarrow w_t \mathcal{L}_{gt}^t$
    end for
    {Back-propagate all gradients with}
    $\mathbf{L}_G = \sum_{t=1}^{|\mathcal{T}|} \mathbf{L}_G^t$
    {Update the overall parameters}
  end for
end for

2. Implementation Details

For the experiments on the cross-city setting with Cityscapes and Cityscapes-depth datasets [2], we train the networks for 120 epochs using 256×256 crops with a batch size of 32. Since there are different numbers of annotated training images in each dataset (2975 in Cityscapes and 20K in Cityscapes-depth), we iterate over all samples in Cityscapes while randomly selecting the same number of samples in Cityscapes-depth to extract the interleaving mini-batches during training. For the experiments on the cross-domain setting with the Cityscapes and Synscapes [5] datasets, we train for 120 epochs using a larger crop size of 384×384 with a batch size of 24. While for the cross-dataset experiments, the two remote sensing datasets, Potsdam and Vaihingen [1], contain very high-resolution true orthophoto (TOP) tiles with different sizes. In particular, the Potsdam dataset consists of 38 TOP tiles of resolution 6000×6000. Regarding the Vaihingen dataset, there are 33 image tiles, each of which has an average of 2500×2000 pixels. As such, we extract patches of size 512×512 from the raw large images using a 50% overlapped sliding window along both the rows and columns for training and testing. We then train the network for 120 epochs using a crop size of 384×384 and a batch size of 24. The same iteration scheme is adopted for the last two settings as the one in the cross-city setting.

The Cityscapes dataset provides the disparity ground-truth pre-computed by the SGM [3] algorithm, which may contain invalid points. We then mask these points during both the training and evaluating process. We also ignore the left 5% and bottom 15% areas of the image where there are noisy disparity values. As the Synscapes dataset provides accurate depth ground-truth, we transform depth into inverse depth to represent points with infinite distance as zero. For the remote sensing datasets, we simply the normalize the nDSM data to the range of [0,1]. We then randomly flip and scale the image between $[0.75, 1.5]$ and rotate the image with a degree between $[-10, 10]$ for the street-view datasets and $[-180, 180]$ for the top-view remote sensing datasets. Finally we crop the image into the fixed size for training.

3. Model Analysis

Effect of task loss weights. We optimize our SemiMTL model with a naive weighted summation of multiple loss functions, and thus the choice of the loss weight for each task is an important factor for the model. We explore the effect of different task loss weights on our framework under the

Table 1. **Sensitivity analysis of loss weights for depth.** We evaluate our method with different loss weights for depth estimation task, which measures the relative importance between segmentation and depth tasks.

Method		Segmentation		Depth					MTL
		pAcc	mIoU	AbR	RMSE	δ_1	δ_2	δ_3	$\Delta M(\%)$
0.01	JTL [4]	94.8	71.4	0.329	5.469	76.6	91.2	95.7	+9.4
	SemiMTL	**94.9**	**71.9**	**0.287**	**5.234**	**79.3**	**92.6**	**96.3**	**+11.5**
0.001	JTL [4]	**94.9**	71.9	0.302	5.479	75.3	90.7	95.5	+9.7
	SemiMTL	**94.9**	**72.7**	**0.269**	**5.352**	**75.5**	**91.1**	**95.9**	**+11.2**

Table 2. **Sensitivity analysis of adversarial loss weights.** We validate different weights to balance the importance between the supervised and semi-supervised losses.

Method	Segmentation		Depth					MTL
	pAcc	mIoU	AbR	RMSE	δ_1	δ_2	δ_3	$\Delta M(\%)$
SemiMTL (S_1)	95.5	71.9	0.280	5.271	78.4	92.2	96.3	+11.3
SemiMTL (S_2)	95.7	73.3	0.308	5.220	79.0	92.4	96.1	+12.6
SemiMTL (S_3)	95.4	72.1	0.309	5.370	77.2	91.1	95.6	+10.7

cross-city setting. Since there are two dense labeling tasks, we directly set the weight for segmentation task as 1.0 and change the depth weight to adjust their relative importance. Table 1 shows that our proposed method performs favorably against the JTL baseline with all weighting parameters. It is worth noting that a smaller depth weight may decrease the performance of depth task, while facilitating that of the segmentation task. We choose the depth weight parameter as 0.01 to obtain a better trade-off for the two tasks.

Effect of adversarial loss weights. During optimizing our SemiMTL framework, another essential factor is to balance the weight between the supervised and semi-supervised losses. In our framework, the semi-supervised loss is represented by the adversarial losses ($\mathcal{L}_{intra}$ and $\mathcal{L}_{inter}$) of the predictions propagated from the discriminator network. We investigate the weight parameters ($\lambda_{intra}, \lambda_{inter}$) with three different sets: $S_1 = (0.001, 0.001)$, $S_2 = (0.001, 0.0001)$, and $S_3 = (0.0001, 0.0001)$, where the first and second parameters denote the loss weights for predictions from labeled and unlabeled datasets, respectively. Table 2 illustrates that a too small or too large semi-supervised loss weight may not facilitate the overall SemiMTL system significantly. We empirically choose the S_2 set of parameters in our experiments.

4. Qualitative Results

We present more qualitative results on the cross-city, cross-dataset, and cross-domain settings for semantic segmentation and depth/height estimation tasks, see Figure 1, 2, and 3. We also provide additional examples to compare the proposed SemiMTL scheme with the baseline methods and demonstrate the effectiveness of our method on both tasks, as shown in Figure 4.

References

[1] 2d semantic labeling contest. `http://www2.isprs.org/commissions/comm3/wg4/semantic-labeling.html`. Accessed July 1, 2020.

[2] 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 *CVPR*, 2016.

[3] Heiko Hirschmuller. Stereo processing by semiglobal matching and mutual information. *IEEE Trans. Pattern Anal. Mach. Intell.*, 30(2):328–341, 2007.

[4] Iasonas Kokkinos. Ubernet: Training a universal convolutional neural network for low-, mid-, and high-level vision using diverse datasets and limited memory. In *CVPR*, 2017.

[5] Magnus Wrenninge and Jonas Unger. Synscapes: A photorealistic synthetic dataset for street scene parsing. *arXiv:1810.08705*, 2018.

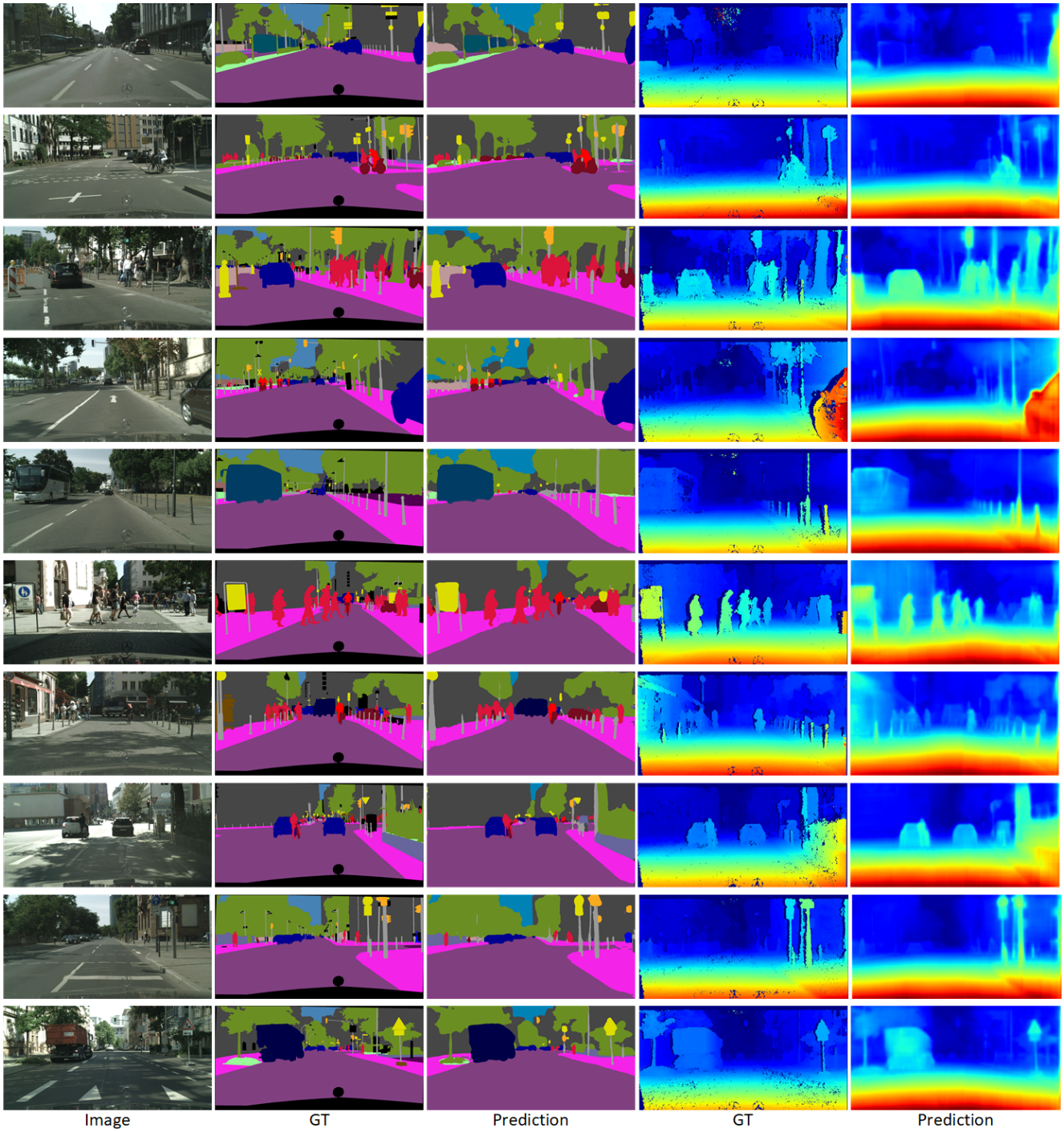

Figure 1. **Qualitative results from the proposed SemiMTL method on the cross-city setting.** The SemiMTL model is trained on the Cityscapes and Cityscapes-depth datasets, and evaluated on the validation set of Cityscapes which provides the ground-truths for both tasks.

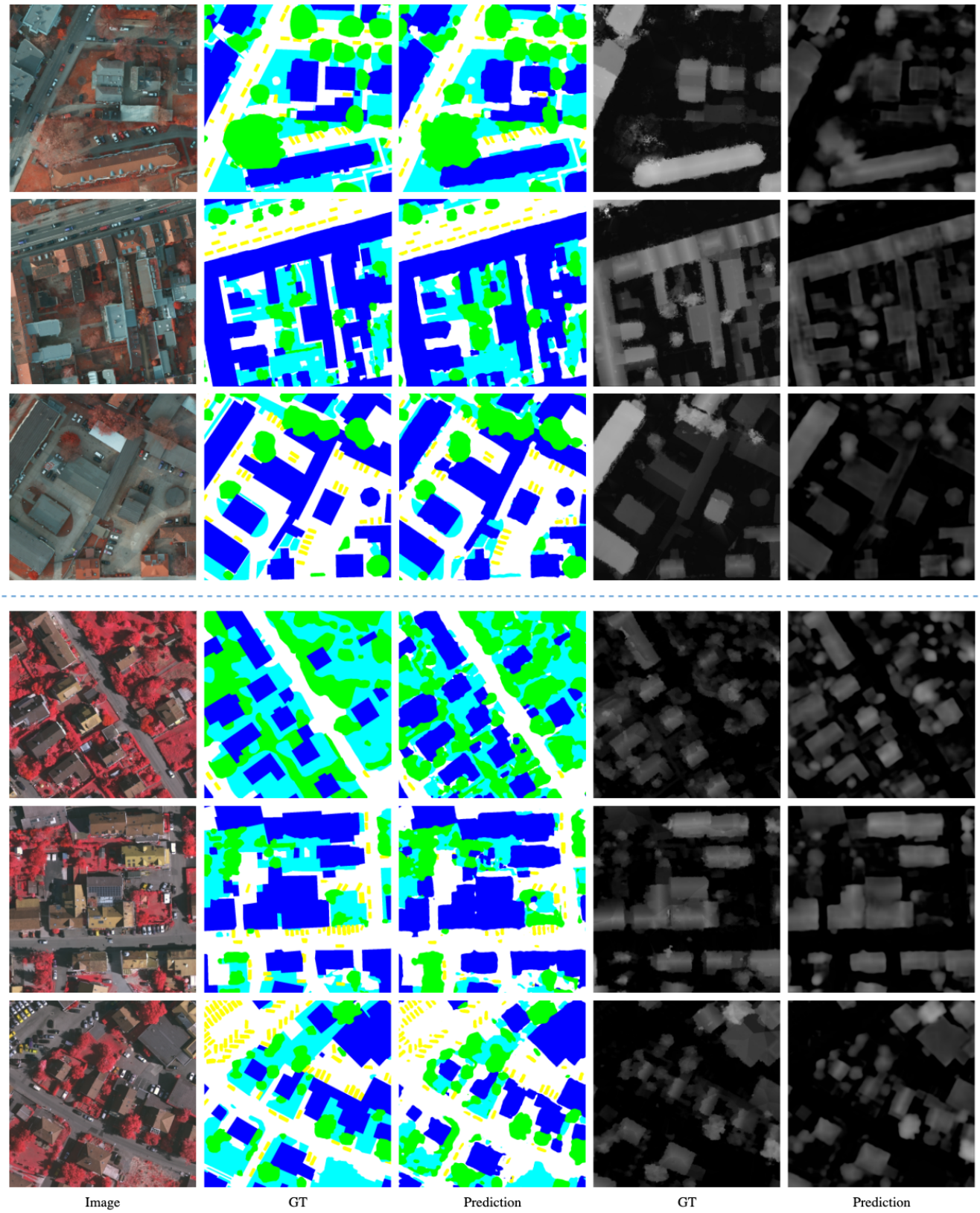

Figure 2. **Qualitative results from the proposed SemiMTL method on the cross-dataset setting.** We train the SemiMTL model with the segmentation annotations from Potsdam and height labels from Vaihingen respectively. We then evaluate the model on their test sets where we have the ground-truths for both tasks. The first and last three rows are the examples drawn from the Potsdam and Vaihingen, respectively.

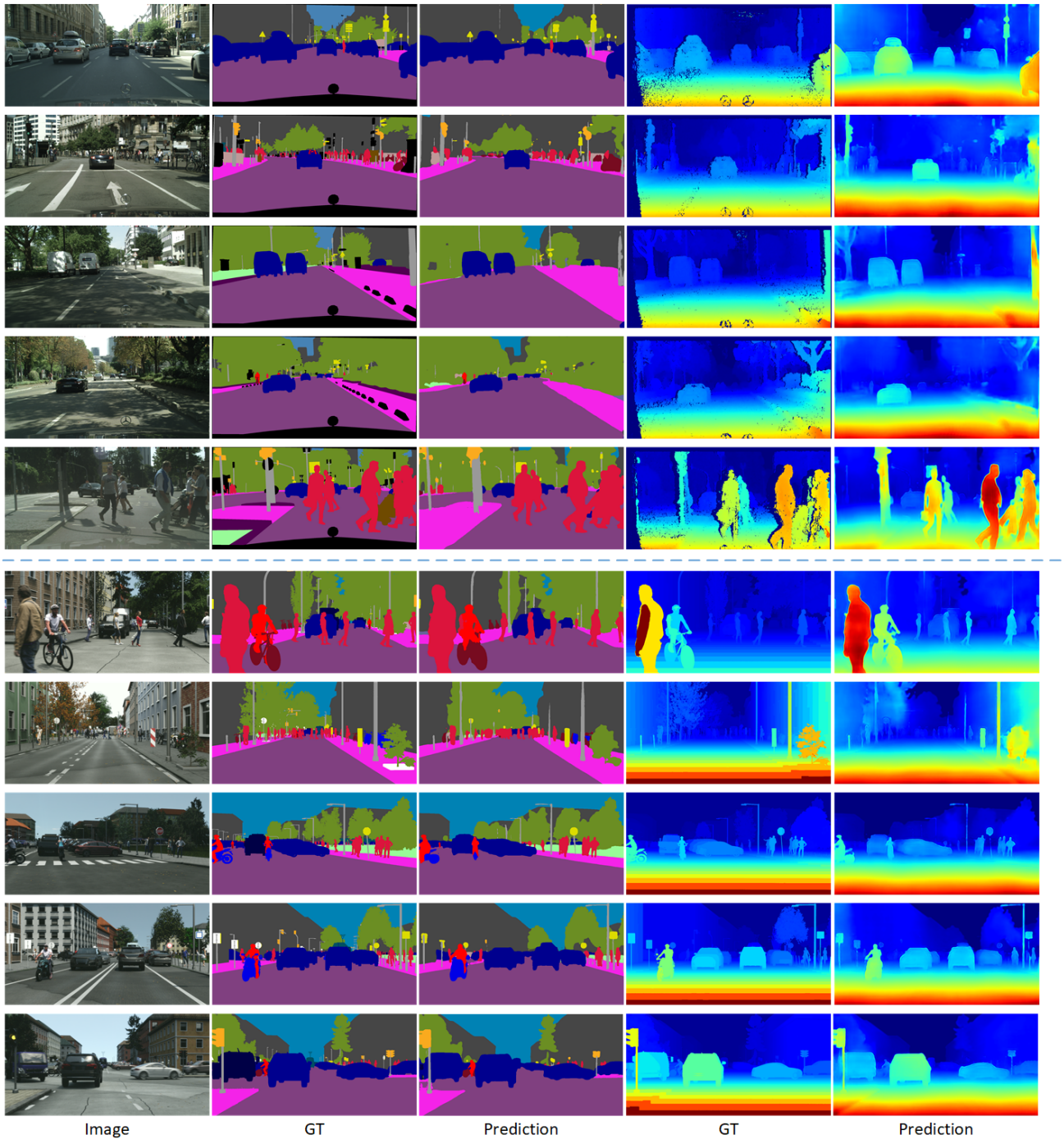

Figure 3. **Qualitative results from the proposed SemiMTL method on the cross-domain setting.** The SemiMTL model is trained on the Cityscapes and Synscapes datasets where the former and latter ones provide the segmentation and depth annotations respectively. We then evaluate the model on the validation set of Cityscapes (upper five examples) and the test set of Synscapes (lower five examples) where we can access to the ground-truths for both tasks.

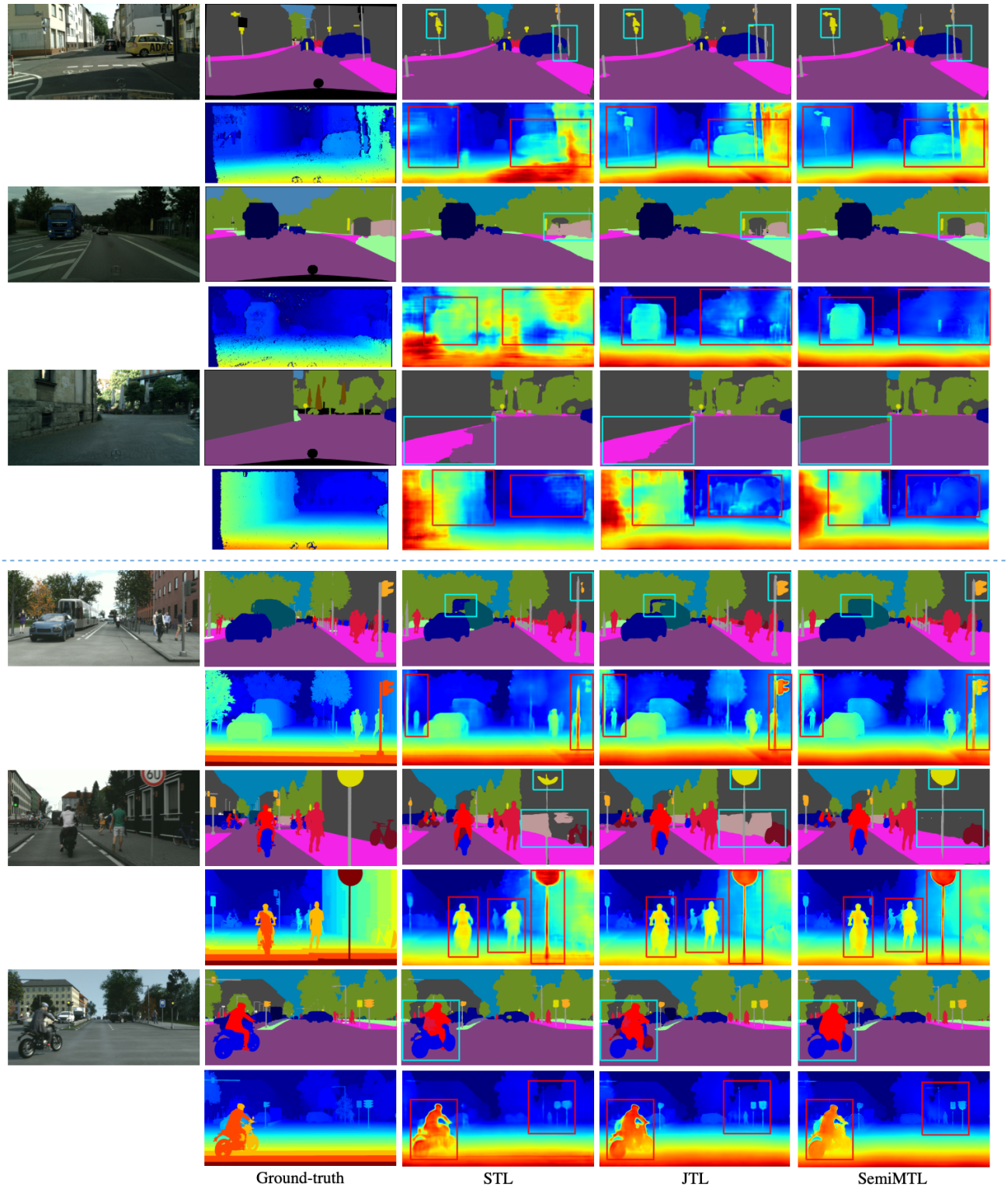

Figure 4. **Qualitative comparison for different methods.** The upper and lower two examples are drawn from the Cityscapes and Synscapes datasets respectively. The visual results show that compared with the baseline methods, the proposed SemiMTL method predicts more accurately for segmentation, and estimates more sharply along boundaries and smoothly within regions for depth. The improvements are highlighted with cyan and red rectangles for segmentation and depth tasks respectively.