Supplementary material for Learning to Adapt for Stereo

Alessio Tonioni*[1], Oscar Rahnama†[2,4], Thomas Joy†[2], Luigi Di Stefano[1], Thalaiyasingam Ajanthan*[3], and Philip H. S. Torr[2]

[1]University of Bologna [2]University of Oxford [3]Australian National University [4]FiveAI

In Sec. 1 we provide additional implementation details concerning the loss used for unsupervised online adaptation, the confidence network and the training procedure of the disparity estimation network. Later in Sec. 2 we give more experimental results on various components of our method that further confirm the findings of the main paper.

1. Implementation Details

1.1. Unsupervised Online Adaptation Loss

To perform unsupervised online adaptation we use a modified version of the re-projection loss described in [3]. Denoting with (i^l, i^r) an RGB stereo image pair with resolution $W \times H$, which is also normalized such that each pixel lies in the range $[0, 1]$. We use this pair as an input for the disparity estimation network which produces a disparity map y^l aligned with i^l as an output.

To compute the unsupervised adaptation loss, first, we use the scaled right input image (i^r) and the disparity map (y^l) to generate a re-projected left input image $(\tilde{i}^l)$ through differentiable bilinear sampling [3]. The error ε_{ij} computed for any given pixel at coordinates (i, j) is measured as a weighted sum of SSIM [7] (measured over 3×3 patches centred on (i, j)) together with an element-wise L_1 distance between real and reprojected pixels. Thus, the pixel error ε_{ij} can be formally expressed as:

$$\varepsilon_{ij} = \alpha \frac{1 - \text{SSIM}\left(i^l_{ij}, \tilde{i}^l_{ij}\right)}{2} + (1 - \alpha)\left\|i^l_{ij} - \tilde{i}^l_{ij}\right\|. \quad (6)$$

We denote with $\alpha = 0.85$ the weight of the linear combination of the two losses. The final loss optimized to perform online adaptation is the mean re-projection error across all the pixels of the whole image.

1.2. Confidence Network Architecture

We implement the confidence estimation network as a three layer 2D convolutional neural network without pooling. Each of these layers uses 3×3 kernels and a stride of 1. The first and second layers have 32 and 64 filters, respectively, and both make use of leaky ReLU activations as well as batch normalization. The final layer has a single channel output and uses the sigmoid function as activation.

We use the re-projection errors (estimated according to Eq. 6) as an input to our confidence network. Before being fed into the confidence network, this re-projected error map is first downsampled to a quarter of its original size to reduce memory consumption and increase inference speed. With this, the network then outputs a confidence map with the same reduced dimensions which we then upscale back to full resolution using bilinear interpolation.

1.3. Training Details

All our systems are entirely implemented in TensorFlow. The pre-training of Dispnet is performed for 1200K steps on *F3D* from random initialization using Adam optimizer with an initial learning rate of 0.0001. The learning rate following the authors' guidelines is halved at 600K steps and then again at 1000K steps. The network is trained by minimizing a weighted sum of the multiple losses computed at different output resolutions. We refer the interested reader to [4] for additional details on the scheduling of the weights associated with loss at different resolutions across the training process.

Fine tuning of both the baselines and our methods is performed for 40K steps on the training dataset with the hyperparameters that are detailed in the main paper. For these fine tunings the loss functions are computed only for prediction at full resolution.

2. Additional Results

2.1. Synthetic to Synthetic Experiments

We extend the experiment performed in Sec. 5.2.2 in the main paper by swapping the role of the two synthetic datasets and report the results in Tab. 3. Here we perform training of different methods on Synthia [5] and test the

*Work done while at University of Oxford.
†Second two authors contributed equally.

	Method	Training Set	Ad. Unsupervised D1-all (%)	Ad. Unsupervised EPE	Ad. Supervised D1-all (%)	Ad. Supervised EPE
(a)	**SL+Ad**	-	16.63	2.56	10.56	1.50
(b)	**SL+Ad**	Synthia	12.53	1.71	5.36	0.87
(c)	**L2A+Ad**	Synthia	10.79	1.48	**5.09**	**0.85**
(d)	**L2A+WAd**	Synthia	**9.98**	**1.45**	✗	✗

Table 3. Comparison of the different methods when training on Synthia and evaluating on sequences from Carla. Similar to our previous results in Tab. 2, the best performing training method is **L2A+WAd**. We also provide results when a L_1 based supervised adaptation loss is used at test time. Best results are in bold.

Method	Train on Carla D1-All (%)	Train on Carla EPE	Train on Synthia D1-All (%)	Train on Synthia EPE
L2A+Ad	22.69	3.08	10.79	1.48
FOL2A+Ad	23.14	3.12	10.81	1.49

Table 4. Comparison between our full framework and a first-order approximation of it. All the models are trained on one of the two synthetic datasets and tested on the other, by performing unsupervised online adaptation.

models performing online adaptation on sequences from Carla. We report results for both unsupervised and supervised online adaptation for the base model trained on *F3D* (row (a)), the model fine-tuned according to an L_1 regression loss (row (b)), and our learning to adapt framework with and without the confidence network (row (d) and (c) respectively).

As in the main paper, our **L2A+Ad** formulation provides an increase in performance over **SL+Ad** for both unsupervised and supervised adaptation. Moreover, the introduction of our confidence weighted adaptation (row (d)) provides an additional small increase in performance for unsupervised adaptation. In this scenario, however, the gap between the different methods appears to be smaller.

2.2. First-Order Approximation

In MAML [2], the use of two nested optimization loops introduces the need for second-order derivatives during the back-propagation phase through gradient computations of the inner loop. Naturally, the computation of second-order derivative comes at a significant computational cost that slows down the training process considerably. To alleviate this issue, the authors of MAML [2] propose a first-order approximation that ignores the costly back-propagation through the gradient computation steps and leads to similar performance at a more affordable computational cost.

Our learning to adapt formulation, described in Sec. 3.1 of the main paper, also uses two nested optimizations, and therefore may benefit from the same kind of approximation. This approximated version can be easily implemented in our framework exactly as in MAML by omitting the computation of the costly second order derivatives during the back propagation. This approximated method is referred to as **FOL2A**.

To measure the impact of the first order approximation on the final performance of the network we trained two additional Dispnet using **FOL2A** on a synthetic dataset (Carla or Synthia) and measured the performance on the other (*i.e.*, we followed the paradigm used for the synthetic to synthetic tests) . In Tab. 4 we compare the performance achieved by the networks trained with **FOL2A** and the corresponding models trained without approximations (*i.e.*, using **L2A**). On both training datasets, the approximated methods perform similarly to the complete ones, with performance almost equal when trained on Synthia. Hence, if one wishes to save on computational resources, this first order approximation approach may be employed.

Unfortunately, weighted adaptation (using the confidence network) is not possible with the first-order approximation as the confidence network is trained using the gradients computed through the adaptation step (inner loop) and these gradients are ignored in the first-order approximation. Therefore, for the sake of fairness, when comparing between the different variants of our method across all the tests, we have always used the version that relies on the computation of the second-order derivatives.

2.3. Qualitative Results on the Confidence Network

In Fig. 6, we show additional qualitative examples of our predicted confidence masks on a challenging sequence from the KITTI dataset. We wish to point out that, in our learning-to-adapt framework, despite the confidence network only being trained on synthetic data, it demonstrates good generalization ability to real images. In almost all examples, the confidence network is able to mask out reprojection errors due to occlusions (usually on the left side of objects in the foreground) or reflective surfaces (*e.g.*, the car on the left in the second row). In some examples, we note that certain points in the background are masked. This slightly unexpected behavior may hint to the presence of noise within the confidence prediction process. Future work

will be dedicated to improve the quality of these confidence estimations by incorporating geometric constraints.

2.4. Qualitative Results on Cityscapes

We show on Fig. 7 some additional qualitative results obtained on the Cityscapes[1] dataset. We show three different disparity prediction obtained using Dispnet as disparity estimation network and three different adaptation strategies; from left to right: no adaptation (**SL**), online adaptation as in [6] (**SL+Ad**) and our proposed framework (**L2A+WAd**). The number of adaptation steps performed increase by roughly 25 together with the row considered. The visualization clearly shows how both adaptation strategies (+Ad. methods) are able to address most of the nuisances in the prediction of **SL**, *i.e*. those produced by a method without adaptation. The differences between **SL+Ad** and **L2A+WAd** are quite evident in the first row, *i.e*. after few step of adaptation, but became quite subtle by the last row. Nevertheless, **L2A+WAd** is always able to obtain sharper and cleaner prediction.

References

[1] 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 *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016. 3

[2] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 1126–1135, International Convention Centre, Sydney, Australia, 06–11 Aug 2017. PMLR. 2

[3] Clément Godard, Oisin Mac Aodha, and Gabriel J Brostow. Unsupervised monocular depth estimation with left-right consistency. In *CVPR*, volume 2, page 7, 2017. 1

[4] Nikolaus Mayer, Eddy Ilg, Philip Hausser, Philipp Fischer, Daniel Cremers, Alexey Dosovitskiy, and Thomas Brox. A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016. 1

[5] German Ros, Laura Sellart, Joanna Materzynska, David Vazquez, and Antonio M. Lopez. The synthia dataset: A large collection of synthetic images for semantic segmentation of urban scenes. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016. 1

[6] Alessio Tonioni, Fabio Tosi, Matteo Poggi, Stefano Mattoccia, and Luigi Di Stefano. Real-time self-adaptive deep stereo. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 3

[7] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural sunsupervised learning ofimilarity. *IEEE transactions on image processing*, 13(4):600–612, 2004. 1

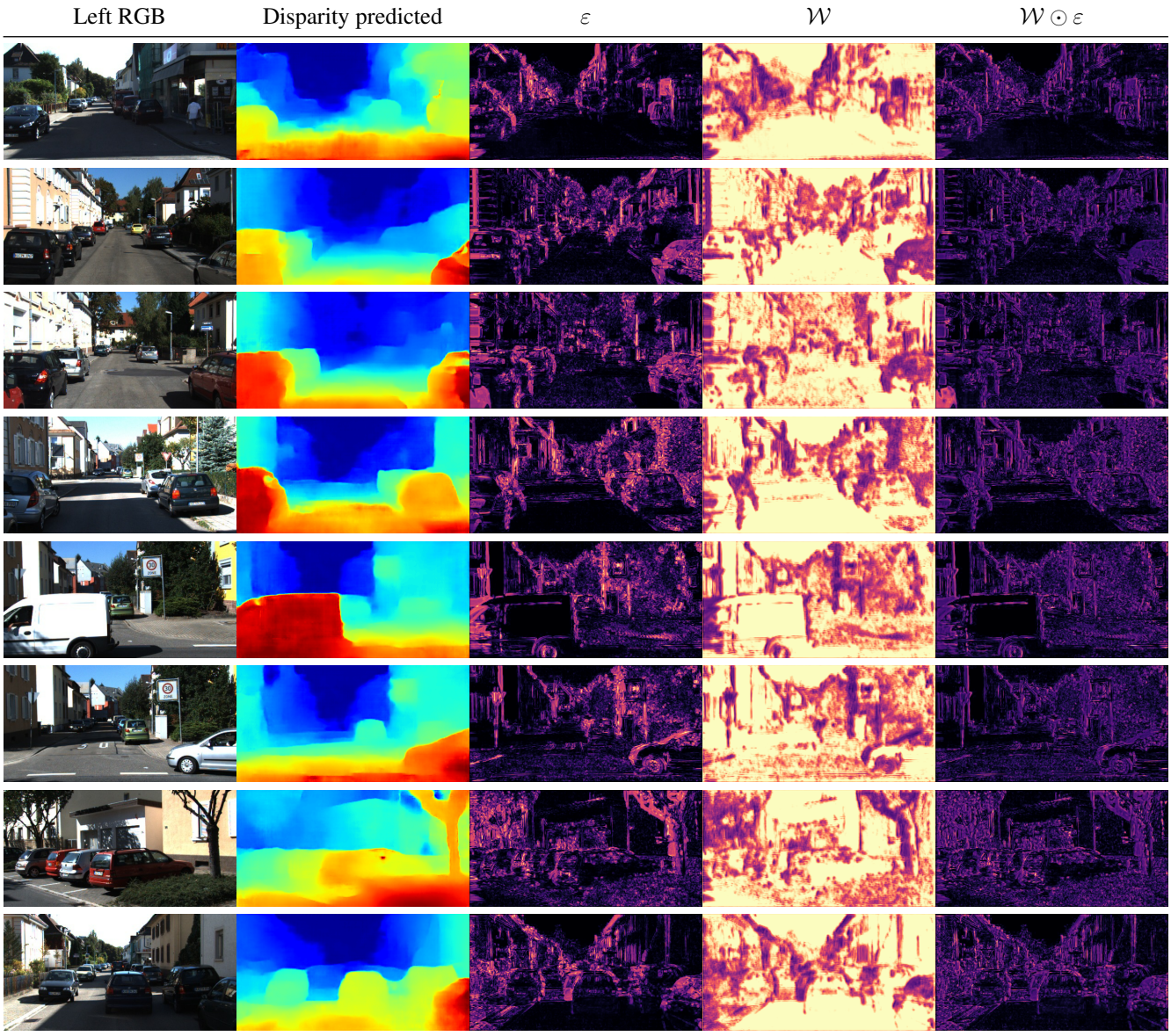

Figure 6. Visualization of our confidence estimations. From left to right: left rgb frame from a stereo pair, disparity predicted by Dispnet, reprojection error obtained as described in Sec. 1.1, confidence mask predicted by our network, weighted re-projection error used for online adaptation. On the last three columns bright colors indicate high values.

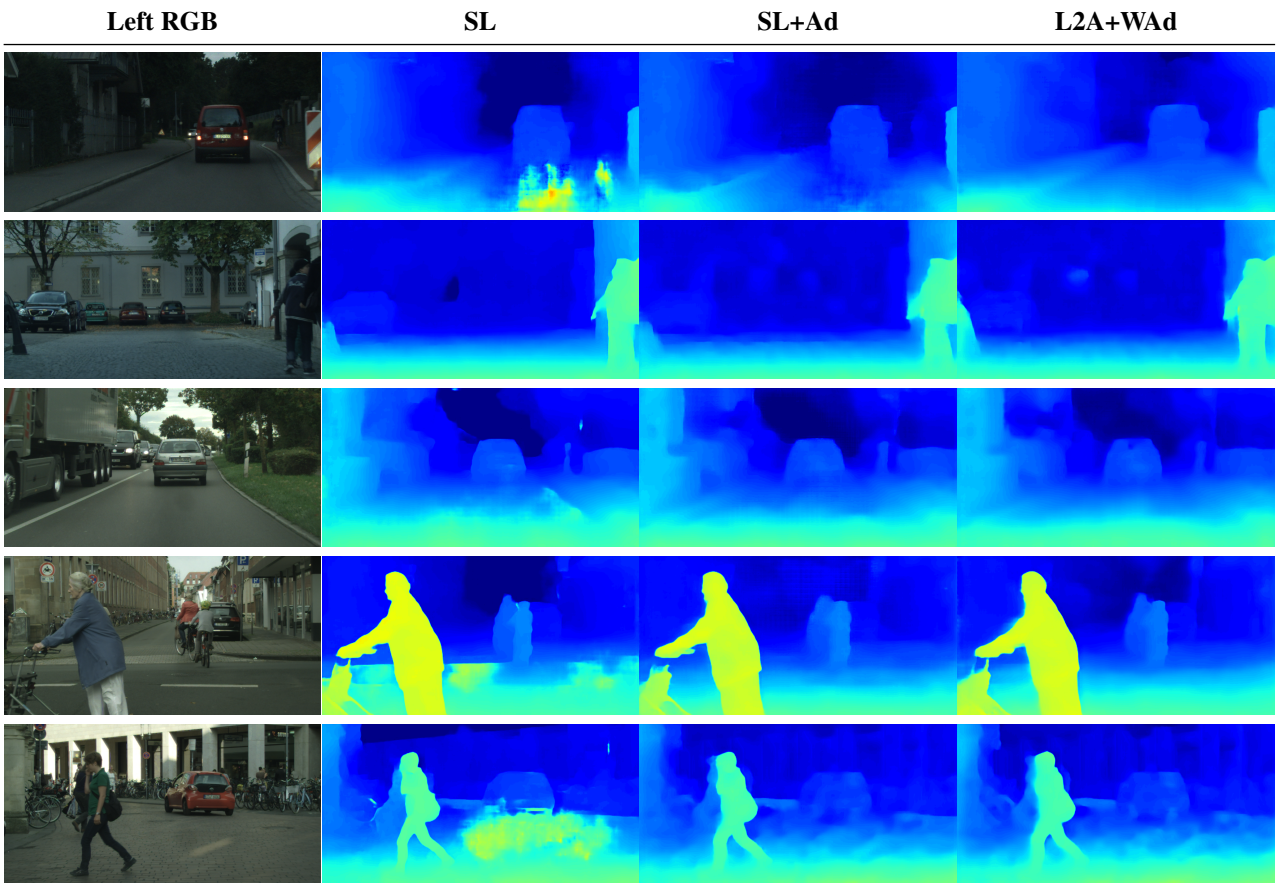

Figure 7. Qualitative results on the Cityscapes dataset. From left to right: left rgb frame from a stereo pair, disparities predicted by Dispnet using as method **SL**, **SL+Ad** and **L2A+WAd** respectivelly. On the last three columns bright colors marks high disparity values. The number of adaptation steps performed increase together with the rows.