Federated Online Adaptation for Deep Stereo

Supplementary Material

This document supplements the CVPR 2024 paper "Federated Online Adaptation for Deep Stereo". It provides additional implementation details and deeper insights into the results reported in the main paper.

6. Implementation Details

6.1. MADNet 2 Architecture

MADNet 2 is implemented on top of MADNet [55]. Specifically, it is made of two, shared feature extractors and a set of five shallow disparity decoders. These modules are assembled to implement coarse-to-fine processing, as shown in Fig. 5.

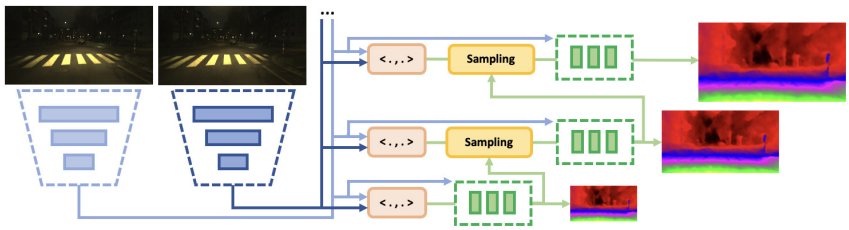

Figure 5. **MADNet 2 architecture.** Given a stereo pair, a set of multi-scale features is extracted by means of two feature extractors with shared weights. Starting from the lowest resolution – *i.e.*, $\frac{1}{64}$ – correlation scores are computed and sampled by means of the all-pair correlation module and lookup operator from [30]. Sampled scores and image features are processed by a disparity decoder, which predicts an initial disparity map at $\frac{1}{64}$ resolution. This latter is upsampled and used by the look operator working on the correlation volume at $\frac{1}{32}$ resolution, then a second decoder predicts a refined disparity map at $\frac{1}{32}$ resolution. This process is repeated up to $\frac{1}{4}$ resolution. There, the final prediction is bilinearly upsampled to the original resolution.

Feature Extractors. These sub-networks are implemented as a sequence of twelve 3×3 convolutional layers, each followed by Leaky ReLU activations with $\alpha = 0.2$. These layers have respectively [16, 16, 32, 32, 64, 64, 96, 96, 128, 128, 192, 192] output channels and [2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1] stride factors. Features extracted from layers having stride equal to 1 are respectively at $\frac{1}{2}$, $\frac{1}{4}$, $\frac{1}{8}$, $\frac{1}{16}$, $\frac{1}{32}$, and $\frac{1}{64}$ of the input resolution and are used to compute coarse-to-fine cost volumes.

All-pairs Correlation Volume [30]. Features obtained from the two extractors are used to build a pyramid of cost volumes, by computing all-pairs correlation scores [30]. Given feature maps $\mathbf{f}, \mathbf{g} \in \mathbb{R}^{H \times W \times F}$, a 3D correlation volume can be computed by computing the inner product between features on the same horizontal line:

$$\mathbf{C}_{ijk} = \sum_{h} \mathbf{f}_{ijh} \cdot \mathbf{g}_{ikh}, \quad \mathbf{C} \in \mathbb{R}^{H \times W \times W} \tag{3}$$

Conversely to the correlation layer used originally in [55], this operation is not bound to a specific search range, thus allowing the network to compute matching scores for all possible candidates along the epipolar line. As the correlation volume produces $\mathbf{C} \in \mathbb{R}^{H \times W \times W}$, this makes the cost volume channels dimension dependent on the resolution of the input image. However, by exploiting the lookup operator from RAFT-Stereo [30] we can sample a fixed number of correlation scores along the channel dimension and build a fixed-size cost volume to be processed, subsequently, by a disparity decoder. Sampling is performed in a 1D neighborhood with a radius 2, selecting $H \times W \times 5$ correlation features.

Disparity Decoders. These modules process correlation scores sampled by the lookup operator, the features produced by the feature extractor from the left image, and the disparity map estimated at the previous stage, in order to predict a refined disparity map at the current resolution. Each decoder is made of five 3×3 convolutional layers, with stride 1 and [128, 96, 48, 32, 1] output channels. Any layer is followed by Leaky ReLU activations with $\alpha = 0.2$, except the last one. Following [35, 55], predicted disparity maps are scaled by a factor $\frac{1}{20}$.

Training. Following [55], we train MADNet2 with a weighted sum of L1 losses computed on each disparity map. Specifically, the pixel-wise L1 between predicted and downsampled ground truth disparity is summed over the entire image. The four terms are summed with weights [0.08, 0.02, 0.01, 0.005] from lower to higher resolution. We use Adam optimizer, batch size 8, and an initial learning rate of $1\mathrm{e}-4$, halved after 150 epochs. We use color and spatial augmentations from [2].

Adaptation. We use Adam optimizer and learning rate $1\mathrm{e}-5$ when adapting MADNet 2 and any other model.

6.2. Pre-trained Models and Proxy Labels

We now report which specific weights have been used for any stereo network involved in our experiments.

RAFT-Stereo [30], CREStereo [24] – We use pre-trained weights provided by [58], as i) they showed slightly better generalization for RAFT-Stereo [30], and ii) official weights pre-trained on SceneFlow are not available for CREStereo [24].

IGEV-Stereo [63], UniMatch [65] – We use the pre-trained weights provided by the authors – respectively, `sceneflow.pth` and `gmstereo-scale2-regrefine3-resumeflowthings-sceneflow-f724fee6.pth`.

Real-time models (TemporalStereo [72], HITNet [52], CoEX [3]) – As the weights available online proved poor generalization from synthetic to real images, we re-trained these three models from scratch on FlyingThings3D [35], using the original losses presented in the respective papers, following the same training protocol and using the same hyper-parameters.

Proxy Labels. Disparity maps used for FULL++/MAD++ are obtained following [41], *i.e.*, by running rSGM[2] and then post-processing the results with left-right consistency check and a speckle filter.

7. Additional Experiments

We now report some additional experiments not fitting the 8-page limit of the main paper.

7.1. MADNet vs MADNet 2

We start by discussing the results achieved by our improved version of the original MADNet [41, 55]. Tab. 6 shows the error rates achieved by different flavors of MADNet without adaptation. Our PyTorch re-implementation already improves over the original source code, with stronger data augmentation [58] further improving generalization from synthetic to real images. Eventually, the all-pairs correlation volume allows to decimate the errors with respect to the original model.

		City		Residential		Campus$^{(\times 2)}$		Road		All	
Model	Impl. details	D1-all (%)	EPE (px)	D1-all (%)	EPE (px)	D1-all (%)	EPE (px)	D1-all (%)	EPE (px)	D1-all (%)	EPE (px)
MADNet	[41, 55]	37.42	9.96	37.04	11.34	51.98	11.94	47.45	15.71	38.84	11.68
	ours (PyTorch)	28.26	4.61	26.10	4.79	34.68	5.05	34.27	6.25	27.82	4.96
	+ augment.	11.51	1.75	9.25	1.63	10.63	1.88	15.47	1.90	10.53	1.69
MADNet 2 **(ours)**	+ cost volume	4.04	1.10	4.05	1.03	6.07	1.29	4.01	1.08	4.21	1.09

Table 6. **MADNet [55] vs MADNet 2.** Results on the *City*, *Residential*, *Campus*, and *Road* sequences from KITTI [17] as defined in [55].

7.2. Federated Adaptation – Listening Client on Low-Powered Hardware

All of the federated experiments carried out in the main paper are performed by running both active and listening clients on 3090 GPUs for simplicity. Accordingly, the listening client runs at a much faster inference speed with respect to the adapting clients, and thus the relative frequency of the updates received from the server will be much lower.

This translates into a lower improvement achieved by the listening client with respect to what would happen in a real use-case, *i.e.*, when it runs on a low-powered platform and is not capable of adapting on its own. In such a case, its processing speed would also be much lower, with a consequent increase of the relative frequency of updates it receives.

To confirm this hypothesis, we run an additional experiment on KITTI, by constraining the listening client to run at lower speed (about 10 FPS). Tab. 7 recalls, on top (a), the results obtained in the main paper with our federated adaptation framework (Tab. 2). At the bottom (b), we report the results achieved in this latter experiment. We can notice how the error rates achieved on most sequences are lower when the listening client runs at a lower speed, confirming our hypothesis.

		City		Residential		Campus$^{(\times 2)}$		Road	
Model	Adapt. mode	D1-all (%)	EPE (px)	D1-all (%)	EPE (px)	D1-all (%)	EPE (px)	D1-all (%)	EPE (px)
MADNet 2	FedFULL	1.42	0.89	1.22	0.80	3.93	1.14	1.12	0.80
	FedMAD	1.48	0.90	1.29	0.81	4.05	1.17	1.16	0.82
	FedFULL++	1.38	0.94	1.12	0.81	3.45	1.10	1.11	0.85
	FedMAD++	1.46	0.95	1.20	0.83	3.55	1.11	1.19	0.87
(a) Listening Client on high-end hardware									
MADNet 2	FedFULL	1.29	0.87	1.21	0.79	3.46	1.25	1.04	0.80
	FedMAD	1.39	0.90	1.35	0.82	3.51	1.27	1.10	0.83
	FedFULL++	1.25	0.90	1.09	0.78	2.99	1.06	0.99	0.82
	FedMAD++	1.32	0.92	1.22	0.82	3.10	1.08	1.04	0.84
(b) Listening Client on low-powered hardware									

Table 7. **Federated Adaptation – Impact of speed by the listening client.** Results on the *City*, *Residential*, *Campus*, and *Road* sequences from KITTI [17]. Performance achieved by a listening client running either on (a) high-end hardware or (b) low-powered platforms.

[2] https://github.com/ivankreso/stereo-vision/tree/master/reconstruction/base/rSGM

7.3. Impact of Randomness on Federated Adaptation

The asynchronous nature of our federated framework makes it susceptible to different, random factors, some of them even out of our control. Indeed, while we can constrain some of these by fixing the random seed in our experiments – *e.g.*, the sequences sampled by the active clients, the blocks sampled by MAD and FedMAD heuristics, etc. – we cannot enforce the very same concurrent behavior for the threads running independently in our experiments.

We measure the impact of these factors in Tabs. 8 to 10, respectively on KITTI, DrivingStereo, and DSEC datasets. Each table reports two distinct experiments, consisting in (a) running our federated framework five times with different random seeds, and (b) running it five times by fixing the very same seed. We report, for each sequence, the margin in terms of D1-all and EPE between the maximum and the minimum measured over the five runs, with ↓ referring to margins lower than 0.01.

Starting from Tab. 8, we can notice how the fluctuations on D1-all are lower than 0.1 in most sequences, even when changing the seed (a). The only exception is the Campus sequence which is, unsurprisingly, the shortest and hardest. When fixing the random seed over multiple runs (b), we can still observe some lower fluctuations, confirming the influence of some factors over which we have no control – such as threads scheduling, initialization, and concurrence to access resources.

The same trend can be observed on Tabs. 9 and 10. In particular, as the sequences from DrivingStereo and DSEC are shorter and more challenging, the impact of randomness is slightly higher with respect to what observed on KITTI.

		City		Residential		Campus$^{(\times 2)}$		Road	
Model	Adapt. mode	D1-all (%)	EPE (px)	D1-all (%)	EPE (px)	D1-all (%)	EPE (px)	D1-all (%)	EPE (px)
MADNet 2	FedFULL	0.02	0.01	0.01	0.01	0.38	0.04	0.04	0.03
	FedMAD	0.02	0.01	0.02	0.01	0.51	0.09	0.05	0.03
	FedFULL++	0.05	0.02	0.02	0.01	0.17	0.03	0.05	0.02
	FedMAD++	0.07	0.02	0.04	0.02	0.26	0.06	0.04	0.02
(a) Different random seeds									
MADNet 2	FedFULL	↓	↓	↓	↓	0.06	0.02	↓	↓
	FedMAD	0.03	0.01	↓	↓	0.15	0.06	0.01	0.01
	FedFULL++	↓	↓	↓	↓	0.01	↓	0.01	↓
	FedMAD++	0.01	↓	↓	↓	0.21	0.04	0.03	0.01
(b) Same random seeds									

Table 8. **Federated Adaptation – Impact of randomness.** Results on the *City*, *Residential*, *Campus*, and *Road* sequences from KITTI [17]. We report the margin between min and max errors over five runs, either when fixing (a) different random seeds or (b) the same seed.

		Rainy		Cloudy		Dusky	
Model	Adapt. mode	D1-all (%)	EPE (px)	D1-all (%)	EPE (px)	D1-all (%)	EPE (px)
MADNet 2	FedFULL	0.08	0.01	0.27	0.03	0.70	0.03
	FedMAD	0.32	0.05	0.20	0.03	1.00	0.09
	FedFULL++	0.02	↓	0.11	0.08	0.65	0.08
	FedMAD++	0.32	0.07	0.19	0.18	0.79	0.09
(a) Different random seeds							
MADNet 2	FedFULL	0.15	0.04	0.03	↓	0.16	0.02
	FedMAD	0.35	0.07	0.15	0.01	0.83	0.04
	FedFULL++	↓	↓	0.01	0.01	0.01	↓
	FedMAD++	0.05	0.02	0.06	0.02	0.19	0.01
(b) Same random seeds							

Table 9. **Federated Adaptation – Impact of randomness.** Results on the *Rainy*, *Dusky*, and *Cloudy* sequences from DrivingStereo [67]. We report the margin between min and max errors over five runs, either when fixing (a) different random seeds or (b) the same seed.

		Night#1		Night#2		Night#3		Night#4	
Model	Adapt. mode	D1-all (%)	EPE (px)	D1-all (%)	EPE (px)	D1-all (%)	EPE (px)	D1-all (%)	EPE (px)
MADNet 2	FedFULL	0.10	0.02	0.07	0.02	0.01	0.01	0.10	0.01
	FedMAD	0.26	0.03	0.21	0.03	0.15	0.01	0.05	0.01
	FedFULL++	0.01	↓	0.01	↓	↓	↓	0.01	↓
	FedMAD++	0.04	0.01	0.06	0.01	0.03	↓	0.06	0.01
(a) Different random seeds									
MADNet 2	FedFULL	0.05	0.01	0.07	0.01	0.06	0.01	0.04	0.01
	FedMAD	0.16	0.01	0.13	0.02	0.14	0.01	0.17	0.01
	FedFULL++	0.01	↓	↓	↓	↓	0.01	↓	↓
	FedMAD++	0.05	0.01	0.06	0.01	0.03	↓	0.05	↓
(b) Same random seeds									

Table 10. **Federated Adaptation – Impact of randomness.** Results on the *Night#1*, *Night#2*, *Night#3*, and *Night#4* sequences from DSEC [14]. We report the margin between min and max errors over five runs, either when fixing (a) different random seeds or (b) the same seed.

7.4. Federated Adaptation with Other Real-Time Networks

In the main paper, we showcased in Tab. 3 how federated adaptation – and online adaptation, in general – can be implemented with other, real-time networks such as TemporalStereo [72], HITNet [52] and CoEX [3], reporting results with photometric loss [55] only due to the lack of space. For completeness, we complement those results here. Tab. 11 completes Tab. 3 with the results achieved by using proxy labels [41] on the KITTI dataset, confirming what already discussed in the main paper.

		City		Residential		Campus$^{(\times 2)}$		Road		Data Traffic		Runtime	
Model	Adapt. mode	D1-all (%)	EPE (px)	D1-all (%)	EPE (px)	D1-all (%)	EPE (px)	D1-all (%)	EPE (px)	To Server (MB/s)	To Client (MB/s)	3090 (ms)	AGX (ms)
CoEX [3]	No Adapt.	2.57	1.04	2.51	0.96	3.97	1.25	2.98	1.02	-	-	19	177
	FULL++	1.00	0.86	0.85	0.78	1.73	0.85	0.94	0.82	-	-	75	1197
	FedFULL++	1.16	0.88	0.96	0.78	2.40	0.98	1.16	0.84	8.4	2.4	19	177
HITNet [52]	No Adapt.	1.99	1.00	2.15	0.93	3.11	1.06	2.07	0.95	-	-	36	404
	FULL++	0.89	0.85	0.83	0.76	1.80	0.83	0.97	0.82	-	-	105	1535
	FedFULL++	1.04	0.89	1.05	0.80	2.24	0.89	1.17	0.85	2.3	0.6	36	404
TemporalStereo [72]	No Adapt.	4.33	1.26	3.47	1.10	3.80	1.19	4.67	1.21	-	-	42	✗
	FULL++	1.04	0.86	0.90	0.77	1.86	0.85	0.88	0.81	-	-	150	✗
	FedFULL++	1.22	0.88	1.01	0.79	2.32	0.95	1.11	0.83	33.5	9.5	42	✗

(b) Single-agent vs Federated Adaptation – proxy labels [41]

Table 11. **Online adaptation by fast networks (TemporalStereo [72], HITNet [52], CoEX [3]) within a single domain – single agent vs federated adaptation.** Results on the *City*, *Residential*, *Campus*, and *Road* sequences from KITTI [17].

Furthermore, Tabs. 12 and 13 complete this evaluation by extending it to DrivingStereo [67] and DSEC [14] datasets. In general, adapting any network through FULL/FULL++ often allows for improving their accuracy and achieving error rates even lower compared to MADNet 2. Nonetheless, none of the three models can stand with this latter in efficiency.

		Rainy		Dusky		Cloudy		Data Traffic		Runtime	
Model	Adapt. mode	D1-all (%)	EPE (px)	D1-all (%)	EPE (px)	D1-all (%)	EPE (px)	To Server (MB/s)	To Client (MB/s)	3090 (ms)	AGX (ms)
CoEX [3]	No Adapt.	13.48	2.53	11.00	1.58	4.46	1.16	-	-	16	130
	FULL	8.33	1.81	9.11	1.41	4.91	1.19	-	-	65	1278
	FedFULL	10.70	2.30	8.32	1.32	4.06	1.09	10.2	3.4	16	130
HITNet [52]	No Adapt.	14.08	2.74	8.88	1.37	4.17	1.14	-	-	29	311
	FULL	10.42	2.00	9.12	1.36	5.56	1.22	-	-	88	1720
	FedFULL	10.05	2.00	6.20	1.15	4.16	1.09	3.3	1.1	29	311
TemporalStereo [72]	No Adapt.	18.53	3.94	13.61	1.80	6.02	1.31	-	-	33	✗
	FULL	11.51	1.95	9.15	1.39	5.98	1.24	-	-	140	✗
	FedFULL	13.86	3.06	7.81	1.32	4.30	1.06	43.7	14.5	33	✗

(a) Single-agent vs Federated Adaptation – photometric loss [55]

CoEX [3]	No Adapt.	13.48	2.53	11.00	1.58	4.46	1.16	-	-	16	130
	FULL++	9.96	2.48	4.80	1.05	3.34	1.14	-	-	60	1192
	FedFULL++	8.52	1.77	6.18	1.15	2.78	0.93	11.4	3.8	16	130
HITNet [52]	No Adapt.	14.08	2.74	8.88	1.37	4.17	1.14	-	-	29	311
	FULL++	10.27	2.33	3.62	0.96	4.99	1.59	-	-	84	1623
	FedFULL++	8.00	1.79	3.57	0.94	3.57	1.06	3.7	1.2	29	311
TemporalStereo [72]	No Adapt.	18.53	3.94	13.61	1.80	6.02	1.31	-	-	33	✗
	FULL++	10.36	2.16	4.88	1.06	4.51	1.23	-	-	130	✗
	FedFULL++	12.94	2.88	6.49	1.20	3.82	1.00	47.8	10.9	33	✗

(b) Single-agent vs Federated Adaptation – proxy labels [41]

Table 12. **Online adaptation by fast networks (TemporalStereo [72], HITNet [52], CoEX [3]) on DrivingStereo [67] – single agent vs federated adaptation.** Results on the *Rainy*, *Dusky* and *Cloudy* sequences as selected in [41].

		Night #1		Night #2		Night #3		Night #4		Data Traffic		Runtime	
Model	Adapt. mode	D1-all (%)	EPE (px)	D1-all (%)	EPE (px)	D1-all (%)	EPE (px)	D1-all (%)	EPE (px)	To Server (MB/s)	To Client (MB/s)	3090 (ms)	AGX (ms)
CoEX [3]	No Adapt.	6.26	1.72	10.81	1.87	8.60	1.64	8.31	1.53	-	-	53	539
	FULL	4.82	1.49	6.28	1.34	5.60	1.27	5.31	1.19	-	-	225	2842
	FedFULL	5.32	1.57	7.57	1.49	6.16	1.33	5.79	1.21	9.6	3.1	53	539
HITNet [52]	Any Adapt.	Out of Memory											
TemporalStereo [72]	No Adapt.	7.17	1.68	10.22	1.92	8.66	1.62	8.40	1.49	-	-	118	✗
	FULL	5.77	1.39	9.74	1.57	8.68	1.45	9.38	1.44	-	-	425	✗
	FedFULL	5.27	1.42	7.19	1.50	6.46	1.32	6.46	1.24	41.3	13.7	118	✗

(a) Single-agent vs Federated Adaptation – photometric loss [55]

CoEX [3]	No Adapt.	6.26	1.72	10.81	1.87	8.60	1.64	8.31	1.53	-	-	53	539
	FULL++	4.06	1.20	5.79	1.28	4.96	1.19	4.93	1.14	-	-	205	2776
	FedFULL++	4.92	1.39	7.01	1.41	5.65	1.28	5.45	1.19	10.9	3.5	53	539
HITNet [52]	Any Adapt.	Out of Memory											
TemporalStereo [72]	No Adapt.	7.17	1.68	10.22	1.92	8.66	1.62	8.40	1.49	-	-	118	✗
	FULL++	4.90	1.25	6.95	1.36	5.89	1.25	5.67	1.18	-	-	380	✗
	FedFULL++	5.38	1.38	7.05	1.42	6.05	1.30	5.96	1.22	46.1	15.1	118	✗

(d) Single-agent vs Federated Adaptation – proxy labels [41]

Table 13. **Online adaptation by fast networks (TemporalStereo [72], HITNet [52], CoEX [3]) on DSEC [14] – single agent vs federated adaptation.** Results on the *Night#1*, *Night#2*, *Night#3* and *Night#4* sequences.

This becomes particularly evident on the AGX board: when adapting on DrivingStereo with FULL/FULL++ (Tab. 12), CoEX and HITNet cannot reach 1 FPS, while MADNet 2 can still run at 2 FPS, or even faster with MAD/MAD++. Consequently, leveraging federated adaptation is the only way for CoEX and HITNet to keep a decent frame rate, respectively about 9 and 3 FPS. Yet, MADNet 2 maintains its supremacy by running at more than 20 FPS in the same setting.

On DSEC (Tab. 13), this gap becomes even larger, with CoEX not even running at 2 FPS and HITNet running out-of-memory when trying to carry out adaptation, whereas MADNet still reaches nearly 10 FPS.

8. Qualitative Results

We conclude with some qualitative examples of disparity maps predicted by the several models involved in our experiments.

Figs. 6 and 7 reports two examples respectively from the *Road* and *Residential* sequences of the KITTI dataset [17]. On each, we report different disparity maps and the corresponding error maps (these latter are dilated to ease visualization). We can appreciate how state-of-the-art models [24, 30, 63, 65] already predict very accurate results, yet with the high runtime highlighted in Tab. 1. Real-time models start from slightly higher error rates when not performing adaptation. Nonetheless, they can reach (and even surpass) the accuracy of state-of-the-art models either by actively adapting over the sequence itself (FULL++) or by leveraging federated optimization performed by other clients running on different sequences (FedFULL++).

Figs. 8 and 9 shows examples from *Cloudy* and *Rainy* sequences in DrivingStereo [67]. There, state-of-the-art models [24, 30, 63, 65] achieve slightly lower accuracy, with real-time networks easily outperforming them through adaptation.

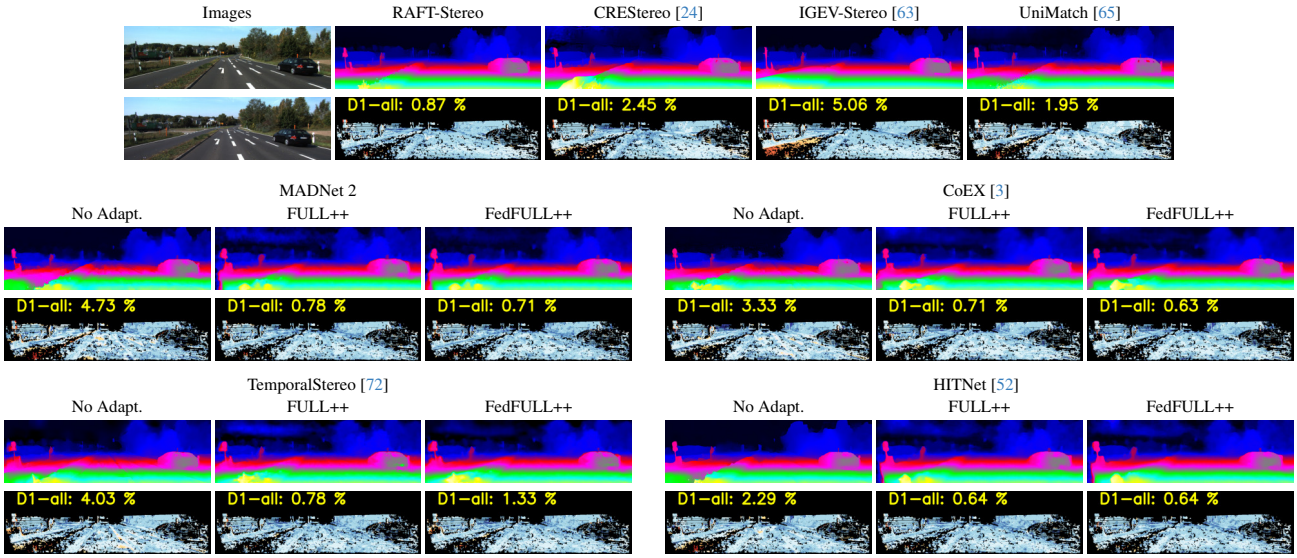

Figure 6. **Qualitative results – KITTI dataset [17], *Road* sequence.**

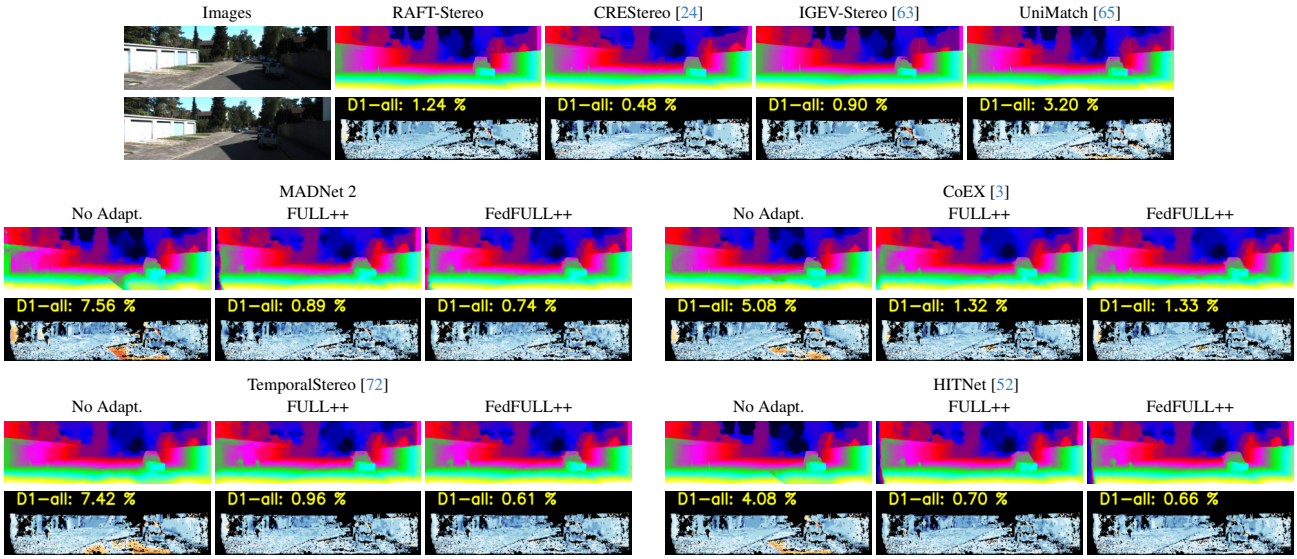

Figure 7. **Qualitative results – KITTI dataset [17], *Residential* sequence.**

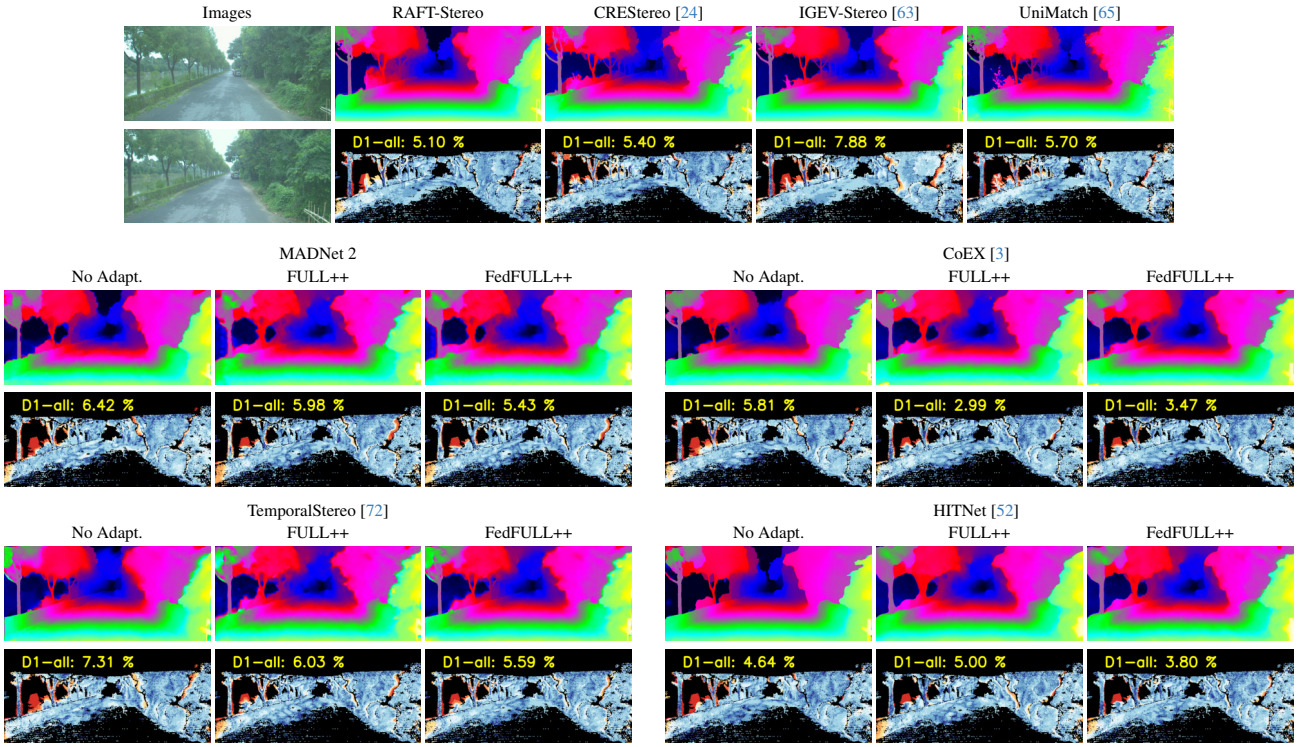

Figure 8. **Qualitative results – DrivingStereo dataset [67], *Cloudy* sequence.**

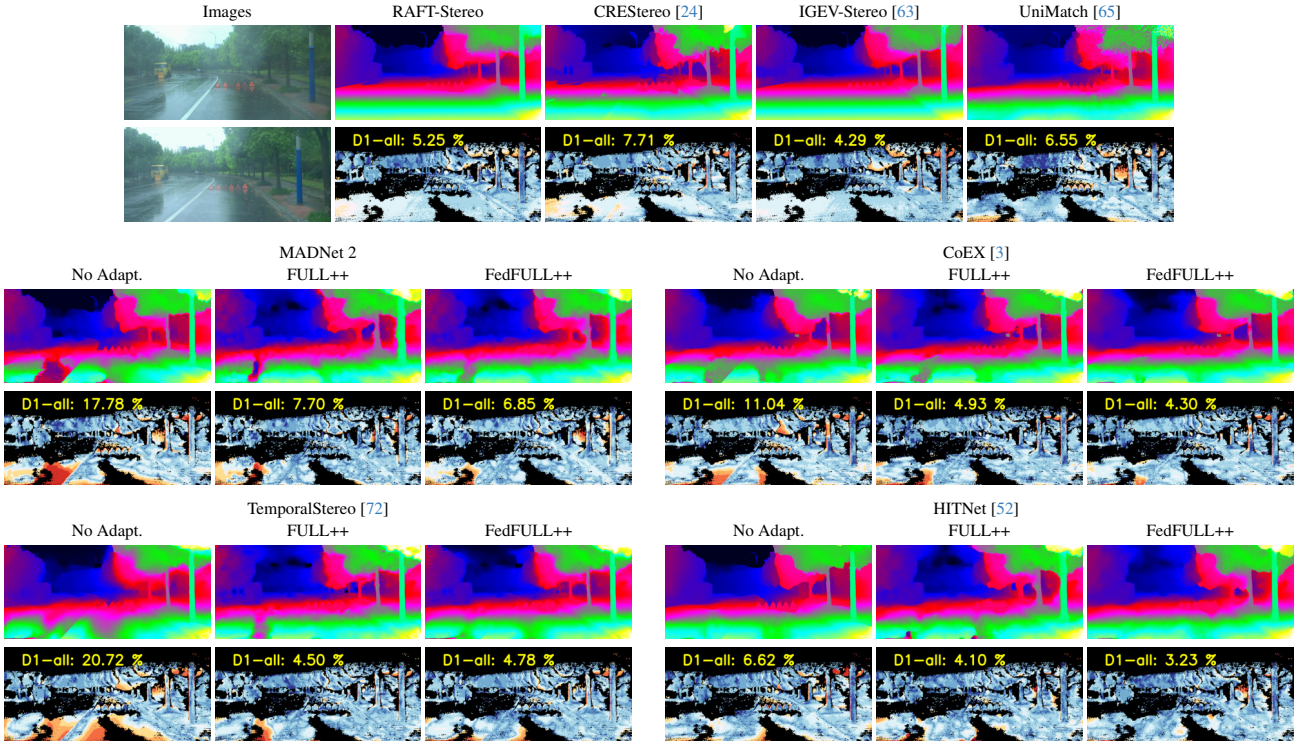

Figure 9. **Qualitative results – DrivingStereo dataset [67], *Rainy* sequence.**

Finally, Figs. 10 and 11 reports qualitative examples from *Night#2* and *Night#4* sequences in DSEC [14]. On this dataset, state-of-the-art models [24, 30, 63, 65] struggle severely – in particular on *Night#2* sequence – because of the sensibly higher noise in the images due to the poor illumination. Again, online adaptation allows real-time models to improve their accuracy on-the-fly and to recover details such as traffic signals that were lost by the original model not performing any adaptation. It is also worthing how the quality of proxy labels used by FULL++ and FedFULL++ is inevitably lower on these scenes, yielding some artifacts to appear in the predictions by the adapted models – *e.g.* at the bottom left.

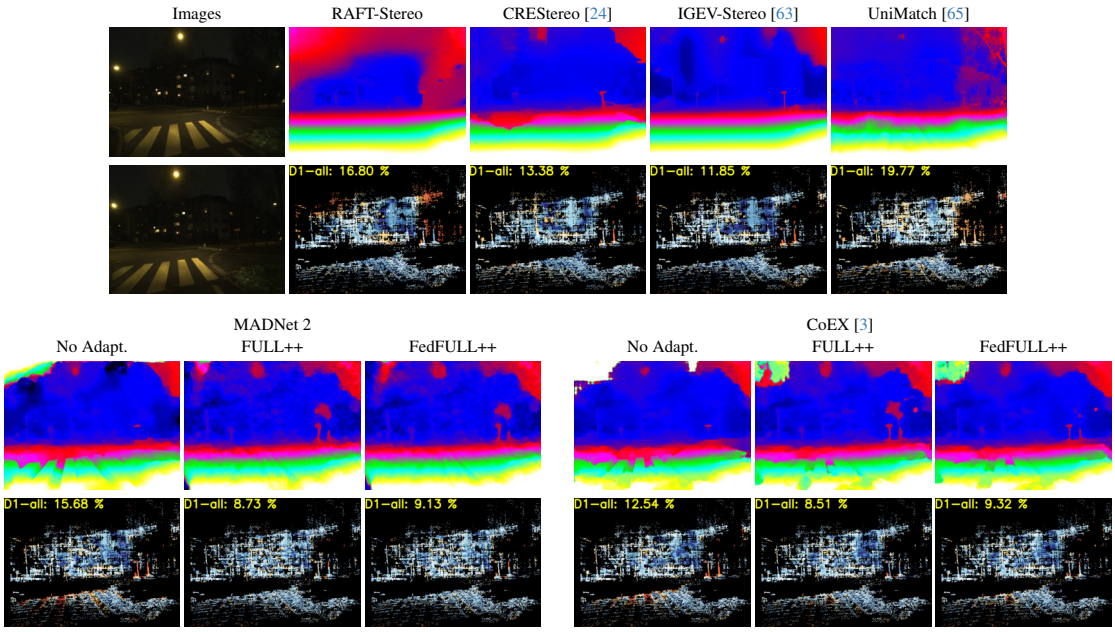

Figure 10. **Qualitative results – DSEC dataset [14], *Night#2* sequence.**

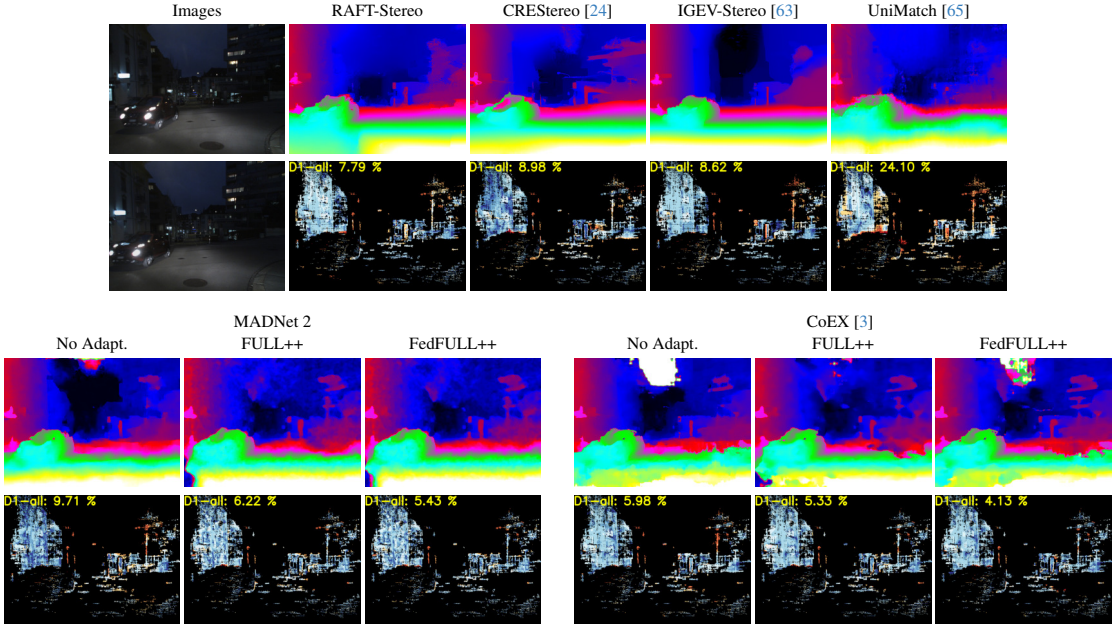

Figure 11. **Qualitative results – DSEC dataset [14], *Night#4* sequence.**