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

Abstract

Real world applications of stereo depth estimation require models that are robust to dynamic variations in the environment. Even though deep learning based stereo methods are successful, they often fail to generalize to unseen variations in the environment, making them less suitable for practical applications such as autonomous driving. In this work, we introduce a "learning-to-adapt" framework that enables deep stereo methods to continuously adapt to new target domains in an unsupervised manner. Specifically, our approach incorporates the adaptation procedure into the learning objective to obtain a base set of parameters that are better suited for unsupervised online adaptation. To further improve the quality of the adaptation, we learn a confidence measure that effectively masks the errors introduced during the unsupervised adaptation. We evaluate our method on synthetic and real-world stereo datasets and our experiments evidence that learning-to-adapt is, indeed beneficial for online adaptation on vastly different domains.

1. Introduction

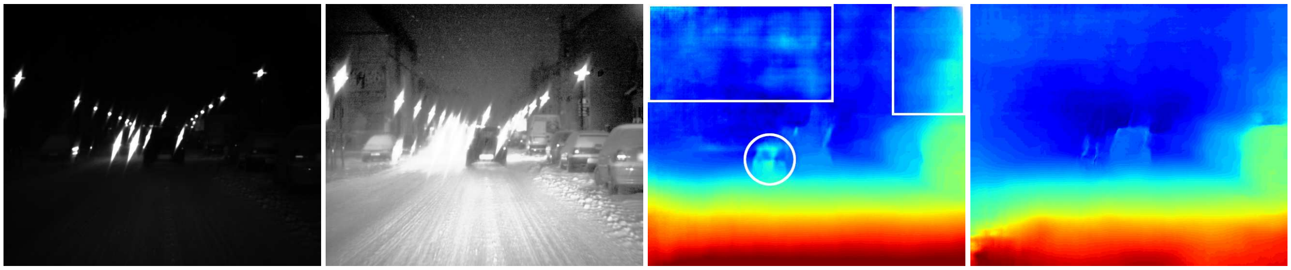
Stereo correspondence estimation is one of the standard methods for predicting the depth of a scene. State-of-the-art algorithms treat stereo as a supervised learning problem and employ deep convolutional neural networks (CNNs) to directly predict the disparity values [15]. However, the inability of deep stereo methods to generalize to new domains [21, 29] presents a serious problem in applications where stereo vision is most useful. Consider an autonomous car driving along the twisting turns, endless meanders and through the frequent tunnels around Lake Como. With few or ineffective barriers offering safety from the shear cliffs, it is imperative that the autonomous car performs flawlessly. Moreover, when passing through frequent tunnels where the lighting conditions change dramatically, a learned stereo system might fail to perform in the expected manner, potentially leading to fatal consequences.

Fine tuning a learned model on the target environment may help to achieve good performance. However, acquiring real dense ground truth data for stereo is extremely challenging, even with expensive equipment and human effort [17]. Moreover, considering the above example, one cannot expect to collect ground truth data for all possible seasons, times of the day, weather conditions, *etc*. To address this issue, we propose to investigate the use of synthetic data to learn a model offline, which, when deployed, can quickly adapt to any unseen target domain online in an unsupervised manner, eliminating the need for expensive data collection.

We formulate this *learning-to-adapt* problem using a meta-learning scheme for continuous adaptation. Specifically, we rely on a model agnostic meta-learning framework [5] due to its theoretical foundation [6], ease of use, and its successful application on various domains [5, 3, 1]. Our goal is to learn a model offline using synthetic data, which can continuously adapt to unseen real video sequences in an unsupervised manner at test time. This means our model is always in training mode and its parameters are automatically tuned to the current environment online without the need for supervision. Such an online adaptation scenario has been considered previously in the literature [30, 34]. However, in this work, we explicitly *learn-to-adapt* which allows us to achieve superior performance.

Our meta-learning approach directly incorporates the online adaptation step into the learning objective, thus allowing us to obtain a base set of weights that are better suited for unsupervised online adaptation. However, since the adaptation is performed in an unsupervised manner (*e.g.*, based on re-projection loss [7, 10]), it is inherently noisy, causing an adverse effect on the overall algorithm. To alleviate this deficiency, we learn a confidence measure on the unsupervised loss and use the confidence weighted loss to update the network parameters. This effectively masks

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



(a) Left input frame (b) Left frame equalized (c) KITTI tuned (d) Ours

Figure 1. We demonstrate the effectiveness of continuous adaptation on a challenging video sequence from [16]. **(a)** Left input frame, **(b)** histogram equalized frame for visualization purpose, **(c)** disparity map produced by a Dispnet-Corr1D [15] trained on annotated real data from KITTI, **(d)** disparity map produced by a Dispnet-Corr1D [15] trained on synthetic data using our learning-to-adapt framework and continuously adapted on this video sequence. The prediction of our method does not suffer from the same artifacts as **(c)** (highlighted in white), thus illustrating the advantage of continuous unsupervised adaptation.

the noise in the adaptation step, preventing detrimental parameter updates. In our case, the confidence measures are predicted using a small CNN which is incorporated into the meta-learning framework, allowing the network to be trained end-to-end with no additional supervision.

In our experiments, we make use of a synthetic stereo dataset (Synthia [25]), a real-world dataset (KITTI-raw [31]), and generate a new synthetic dataset containing multiple sequences of varying weather and lighting conditions using the Carla simulator [4]. We evaluate our algorithm between two pairs of dataset domains: 1) Carla to Synthia; and 2) Carla or Synthia to KITTI-raw. In all experiments, our learning-to-adapt method consistently outperforms previous unsupervised adaptation methods [30], validating our hypothesis that learning-to-adapt provides an effective framework for stereo.

2. Problem Setup and Preliminaries

In this section, first we formalize online adaptation and discuss its advantages. We then briefly review a meta-learning algorithm which we will transpose into our continuous adaptation scenario.

2.1. Online Adaptation for Stereo

Let us denote two datasets of stereo video sequences: $\mathcal{D}_s$ (supervised), with available ground truth, and $\mathcal{D}_u$ (unsupervised), without ground truth. We would like to learn network parameters offline using $\mathcal{D}_s$, and use $\mathcal{D}_u$ as the target (or test) domain. However, in contrast to the standard evaluation setting and following the evaluation protocol of [30, 34], the model is allowed to adapt to the target domain in an unsupervised manner. We follow the online adaptation paradigm proposed in [30], *i.e.*, for each new frame acquired we perform a single gradient descent step to keep the optimization fast and allow better handling of a rapidly changing test environment.

Formally, let the parameters of the base model trained on $\mathcal{D}_s$ be θ. Given an unseen target video sequence $\mathcal{V} \in \mathcal{D}_u$, the adaptation is iteratively performed for each consecutive stereo pair in the sequence, using gradient descent on a predefined unsupervised loss function ($\mathcal{L}_u$). At iteration t, the online adaptation can be written as:

$$\theta_t \leftarrow \theta_{t-1} - \alpha \nabla_\theta \mathcal{L}_u(\theta_{t-1}, i_t) \,, \tag{1}$$

where $\theta_0 = \theta$, $\alpha > 0$ is the learning rate and i_t denotes the stereo pair of t^{th} frame of the sequence $\mathcal{V}$. Note that the network parameters are *continuously* updated for the entire video in a sequential manner. This process is repeated for each video sequence starting from the base model θ.

Motivating Example. To show that deep CNN based stereo networks are highly sensitive to domain-shift and that online adaptation is indeed necessary, we give a motivating example as follows. We select a video sequence from [16] as a test domain, where the environment is similar to that of KITTI but features extreme weather conditions (*e.g.*, night, snow, *etc.*). We compare the predicted disparities of a Dispnet-Corr1D network [15] for two training regimes. The first is fine-tuned on the KITTI training sets [9, 17], and the second is trained on synthetic data using our learning-to-adapt framework and performs unsupervised online adaptation for the target domain. The results are shown in Fig. 1. Here it is evident that fine tuning on real images is not sufficient to obtain reliable performance across all environments as evidenced by the major mistakes in (c) marked in white. As can be seen, (c) behaves worse than the network trained only on synthetic data and adapted online to the target domain in an unsupervised manner by our formulation (d).

2.2. Model Agnostic Meta Learning

Model Agnostic Meta Learning (MAML) [5] is a popular meta-learning algorithm designed for few-shot learning problems. The objective is to learn a base model θ^*, which enables fast adaptation to new tasks when used as initial

Algorithm 1 Adaptation at training time for sequence $\mathcal{V}^\tau$

```
Require: $\theta, \mathcal{V}^\tau = [i_1^\tau, \ldots, i_n^\tau]$
$\theta_0^\tau \leftarrow \theta$    ▷ Parameter initialization
for $t \leftarrow 1, \ldots, n-1$ do
    $\theta_t^\tau \leftarrow \theta_{t-1}^\tau - \alpha \nabla_{\theta_{t-1}^\tau} \mathcal{L}_u \left(\theta_{t-1}^\tau, i_t\right)$    ▷ Adaptation
    $\mathcal{L}_s \left(\theta_t^\tau, i_{t+1}^\tau\right)$    ▷ Supervised evaluation
```

weights. This is achieved by forming a nested optimisation problem, where, in the inner loop, we perform SGD for each task in the standard way. In the outer loop, the base model parameters are optimized using the loss of all the tasks, enabling fast adaptation.

Let $\mathcal{T}$ be the set of tasks in the training set and let the task-specific training and validation sets be $\mathcal{D}_\tau^{train}$ and $\mathcal{D}_\tau^{val}$ respectively for $\tau \in \mathcal{T}$. Assuming a single gradient step in the inner loop, the overall MAML objective can be written as:

$$\min_\theta \sum_{\tau \in \mathcal{T}} \mathcal{L}(\theta - \alpha \nabla_\theta \mathcal{L}(\theta, \mathcal{D}_\tau^{train}), \mathcal{D}_\tau^{val}), \qquad (2)$$

where $\alpha > 0$ is the learning rate used for adaptation. As previously stated, this meta-objective function is optimized via a two-stage gradient descent algorithm. Specifically, at each optimization iteration, the inner-loop performs a gradient descent update for each task separately starting from a common base model θ (adaptation step). Then, the outer-loop performs an update on the common base model, where the gradient is the sum of task-specific gradients computed using the parameters updated in the inner loop. We refer the interested reader to the original paper for more detail [5].

3. Learning to Adapt for Stereo

We first design a meta-learning algorithm for stereo adaptation by incorporating unsupervised continuous adaptation into the training paradigm. Then, we introduce a new mechanism to re-weight the pixel-wise errors estimated by the unsupervised loss function, making the adaptation more effective.

3.1. Meta Learning for Stereo Adaptation

Our hypothesis is that for any deep stereo network, before performing online adaptation to a target domain, it is beneficial to learn a base set of parameters (θ) that can be adapted quickly and effectively to unseen environments. We observe that our objective of learning-to-adapt to unseen video sequences is similar in spirit to that of MAML. Here, we perform the single task of dense disparity regression, but learn how to adapt to different environments and conditions.

We model an environment through a stereo video sequence $\mathcal{V}^\tau = [i_1^\tau, \ldots, i_n^\tau]$[1]. At test time, the parameters are continuously adapted to the sequence $\mathcal{V}^\tau$ in an unsupervised manner according to Eq. 1. At training time, we mimic the same adaptation process on training sequences and evaluate the performance of the model after each adaptation step on the subsequent frame. To measure the performance, we rely on a supervised loss function $\mathcal{L}_s$ (*e.g.*, L_1 or L_2 regression). This procedure for a single sequence $\mathcal{V}^\tau$ is given in Alg. 1.

Algorithm 2 Learning to Adapt for Stereo

```
Require: Training set $\mathcal{D}_s$, and hyper-parameters $\alpha, \beta, k, b$
Initialize $\theta$
while not done do
    $\mathcal{D}^b \sim \mathcal{D}_s$    ▷ Sample a batch of sequences
    for all $\mathcal{V}^\tau \in \mathcal{D}^b$ do
        $\theta^\tau \leftarrow \theta$    ▷ Initialize model
        $L^\tau \leftarrow 0$    ▷ Initialize accumulator
        $[i_s, \ldots, i_{s+k}] \sim \mathcal{V}^\tau$    ▷ Sample $k$ frames
        for $t \leftarrow s, \ldots, s+k-1$ do
            $\theta^\tau \leftarrow \theta^\tau - \alpha \nabla_{\theta^\tau} \mathcal{L}_u(\theta^\tau, i_t)$    ▷ Adaptation
            $L^\tau \leftarrow L^\tau + \mathcal{L}_s(\theta^\tau, i_{t+1})$    ▷ Evaluation
    $\theta \leftarrow \theta - \beta \nabla_\theta \sum_{\mathcal{V}^\tau \in \mathcal{D}^b} L^\tau$    ▷ Optimization
```

During training we perform this adaptation on a supervised training set of video sequences $\mathcal{D}_s$ (*e.g.*, a set of rendered synthetic video sequences). The final objective of our problem is to maximise the measured performance across all frames and all sequences in $\mathcal{D}_s$. This can be written in a compact form as:

$$\min_\theta \sum_{\mathcal{V}^\tau \in \mathcal{D}_s} \sum_{t=1}^{n-1} \mathcal{L}_s(\theta_t^\tau, i_{t+1}^\tau) \,, \qquad (3)$$

where θ_t^τ is obtained sequentially through updates as detailed in Alg. 1.

Note that this formula extends Eq. 2 (MAML) to the continuous and unsupervised adaptation scenario. Contrary to Eq. 2, we use two different loss functions: 1) an unsupervised loss ($\mathcal{L}_u$) to adapt a model to a video sequence; and 2) a supervised loss ($\mathcal{L}_s$) for the optimization of the set of parameters θ. We make this distinction to mimic the test time behaviour. Specifically, $\mathcal{L}_u$ (*i.e.*, some form of unsupervised loss function) will be used at test time, while $\mathcal{L}_s$ can use all the available annotations of the training set for optimization. Our intuition is that by using two different loss functions, θ can be optimized such that it is better suited to be adapted without supervision (*i.e.*, by $\mathcal{L}_u$), while the performance is measured with respect to the ground truth (*i.e.*, by $\mathcal{L}_s$).

Note that optimizing Eq. 3 on complete video sequences would be infeasible for long video sequences as the memory requirement grows linearly with n. To alleviate this, we

[1] For simplicity we assumed that all video sequences are of same length, but this is not a necessity.

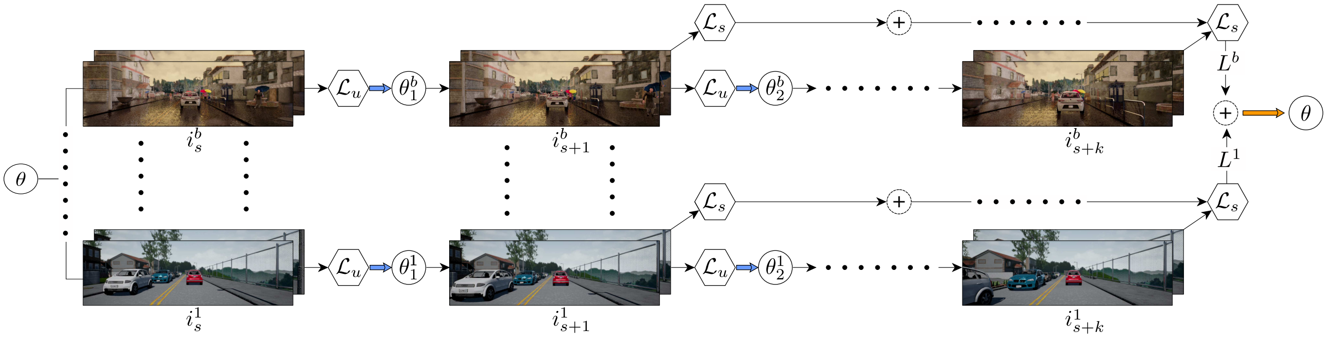


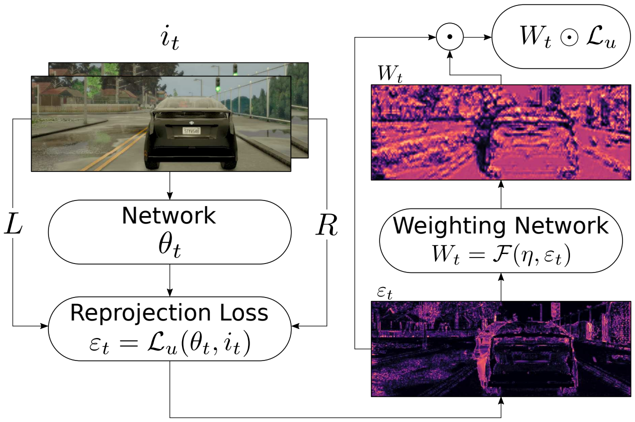
Figure 2. Representation of one iteration of meta-learning of network parameters θ using a batch of b sequences and sampling k frames from each. We represent loss computation steps with hexagons and gradient descent steps with colored arrows. Blue and orange arrows denote adaptation steps and meta-learning steps, respectively. Starting from an initial set of parameters θ, the network is adapted independently on each sequence using loss function $\mathcal{L}_u$. The adapted models are evaluated on the following frame of each sequence using loss function $\mathcal{L}_s$. Finally, the initial parameters θ are updated via a gradient descent to minimize the sum of loss functions obtained by all evaluated models.

approximate it by optimizing over batches of sequences of k randomly sampled consecutive frames. Our meta-learning algorithm is detailed in Alg. 2. After sampling a batch of sequences (line 3) and k random frames from each sequence (line 7), we perform unsupervised adaptation on the current frame (line 9) and measure the effectiveness of this update on the following frame (line 10). This process is repeated for k frames. Finally, we optimize the base model parameters θ to minimize the sum of the supervised losses computed across all the sequences and all the frames (line 11). Here, α and β are the two learning rates used for online adaptation and for meta training, respectively. In Fig. 2, we illustrate one optimization iteration of the network parameters θ using a batch of b sequences and k frames from each.

By optimizing Eq. 3 we are able to learn a base parameter configuration θ suited for adaptation. However, the use of an imperfect unsupervised loss function ($\mathcal{L}_u$) for adaptation introduces mistakes in the optimization process that may have an adverse effect on the overall algorithm. To alleviate this issue, we introduce a mechanism to learn to recognize the noise (or mistakes) in the unsupervised loss estimation which can then be masked effectively.

3.2. Confidence Weighted Adaptation

Unsupervised loss functions for dense disparity estimation often compute some form of pixel-wise error map and minimize over the average mistake. Unfortunately, this process is not perfect and usually introduces errors in the optimization process. This may result in sub-optimal performance when compared to the use of supervised loss functions. For example, the left-right re-projection loss proposed in [7] is well-known to produce mistakes in occluded areas and reflective surfaces. These mistakes are not due to bad predictions by the disparity estimation model, but instead are due to differences between the left and right frames. Ideally, we would like to have a confidence function to detect erroneous estimations of this loss such that they can be effectively masked. However, training such a confidence function might be difficult since there is no easy procedure to obtain ground-truth annotations for this task. We propose to avoid explicit supervised training, and instead, automatically learn to detect noise in the loss estimations by incorporating this new objective into our meta-learning formulation.



i_t
L
R
Network
θ_t
Reprojection Loss
$\varepsilon_t = \mathcal{L}_u(\theta_t, i_t)$
$W_t \odot \mathcal{L}_u$
W_t
Weighting Network
$W_t = \mathcal{F}(\eta, \varepsilon_t)$
ε_t

Figure 3. Schematic representation of our weighted adaptation for a single stereo frame using an unsupervised re-projection based loss function $\mathcal{L}_u$ (bright colors indicate higher values). The system takes a stereo-pair (i_t) and computes a disparity map as well as a re-projection loss (ε_t). This loss is then weighted according to W_t effectively masking the mistakes.

In particular, we propose to learn a small CNN that takes a pixel-wise error map estimated by $\mathcal{L}_u$ as an input and produces a tensor W as an output, which has the same shape as the input and its elements are between 0 and 1. This output can be interpreted as a per pixel confidence on the reliability of the loss estimation with 1 corresponding to high reliability. We can now mask out potentially erroneous estimations by multiplying the loss values by their corresponding confidence values. The result is a cleaner measurement of the loss function that reduces detrimental weight updates due to incorrect loss values. The idea of masking or weighting the contribution of single examples in the presence of noise or class imbalance in the labels has been previously studied for supervised classification in [11, 24]. In our case, we transpose the similar idea to pixel-wise loss, estimated for a dense regression task and directly predict a dense confidence map.

Let $W = \mathcal{F}(\eta, \varepsilon)$ be the mask produced by the re-weighting network parametrized by η and $\varepsilon = \mathcal{L}_u(\theta, i)$ be the estimated pixel-wise error map computed on the prediction of the disparity model with parameter θ on stereo frame i. We normalize the elements of W by dividing each one of them by the number of elements in W. Now, by modifying Eq. 1, the final weighted adaptation formula can be written as:

$$\begin{aligned}&\widetilde{\theta}_t \leftarrow \widetilde{\theta}_{t-1} - \alpha \nabla_{\widetilde{\theta}} \left(W_t \odot \mathcal{L}_u(\widetilde{\theta}_{t-1}, i_t) \right) \ , \\ &W_t = \mathcal{F}(\eta, \mathcal{L}_u(\widetilde{\theta}_{t-1}, i_t)) \ .\end{aligned} \tag{4}$$

where $\widetilde{\theta}_0 = \theta$ and $\odot$ indicating the element-wise product between the matrices. Note that, the dimension of $\widetilde{\theta}$ is the same as that of θ, however we denote it differently to highlight the fact that $\widetilde{\theta}$ depends on both the base model (θ) as well as the re-weighting network (η).

In Fig. 3, we show a schematic representation of our proposed weighted adaptation computed from a single stereo input frame i_t. On the bottom right corner we give a visualization of the error map produced by an unsupervised re-projection loss $\mathcal{L}_u$, while the top right corner shows a possible confidence mask W_t. In this example the weighting network is masking errors due to occlusions (*e.g.*, on the left side of the car) and due to reflections (*e.g.*, the puddles on the road).

Since supervision is not available for W, we indirectly train η by incorporating Eq. 4 inside the meta-learning objective described in Eq. 3. The final objective of our complete system becomes:

$$\min_{\theta, \eta} \sum_{\mathcal{V}_\tau \in \mathcal{D}_s} \sum_{t=1}^{n-1} \mathcal{L}_s(\widetilde{\theta}_t^\tau, i_t^\tau) \ . \tag{5}$$

Here, $\widetilde{\theta}_t^\tau$ are the parameters of the model updated according to the weighted unsupervised loss function on sequence $\mathcal{V}^\tau$. As such it depends on both η and θ according to Eq. 4. The whole network can be trained end-to-end with the only supervision coming from the depth annotations used to compute $\mathcal{L}_s$. Both θ and η are tuned to maximize the network performances after few steps of optimization as measured by $\mathcal{L}_s$. By optimizing a single objective function we are able to learn the parameters (η) of the weighting network, and a set of base weights for the disparity estimation model (θ) which allow for fast adaptation.

4. Related Work

Machine Learning for Stereo. Mayer *et al.* [15], proposed the first end-to-end stereo architecture that, despite not having achieved state-of-the-art accuracy, initiated a huge shift in stereo literature towards CNN based models. More recent proposals [13, 20, 14, 2, 12] have quickly reached top performance on the challenging KITTI benchmarks by deploying 3D convolution [13], two-stage refinement [20] and pyramidal elaboration [2]. All these works share the same training protocol. Specifically, the network is first pretrained on the large and perfectly annotated synthetic FlyingThings3D dataset [15] and then fine tuned on the smaller KITTI training sets.

Unsupervised Adaptation for Stereo. Tonioni *et al.* [29] highlight how machine learning models for stereo are data dependent and, if exposed to environments different from the ones observed during training, will suffer from a severe loss in performance. To overcome this problem they introduce an unsupervised way of adapting networks to new domains by deploying traditional stereo algorithms and confidence measures. Pang *et al.* [21] achieve the same objective by an iterative optimization of predictions obtained at multiple resolutions, while many recent works [35, 10, 33, 22] warp different views according to the predicted disparity and minimize the re-projection error.

Recently, the adaptation problem has also been addressed through an online learning perspective focusing on inference speed [30]. On a related topic, Zhong *et al.* [34] propose to use video sequences to train a deep network online from random initialization. Moreover, they employ a LSTM in their model to leverage temporal information during the prediction. Similarly to [30, 34], we constantly train our network when deployed on unseen environments, but we additionally propose to learn a good set of initial weights and a confidence function for the loss function that will improve the adaptation process.

Meta Learning. Meta-learning is a long-standing problem in machine learning [19, 28, 26] that tries to exploit structures in data to learn more effective learning rules or algorithms. Most of the recent developments in meta-learning algorithms have focused on the task of few shot classification [32, 27, 23], with few exceptions like [5, 18] extending their models to simple function regression and reinforcement learning. In [5], the authors propose to constrain the learning rule for the model to be stochastic gradient descent and update the initial weight configuration of a network to make it more suited to learning new tasks. This simple formulation has been recently extended to address online adaptation in reinforcement learning using meta-learning to adapt to changing agents [3] or non-stationary and competitive environments [1]. Our work builds on [5] by modifying it to use structured regression, unsupervised loss functions and temporal consistency during the update process.

5. Experiments

This section presents an evaluation of the quality of our proposed adaptation method. Firstly, we lay out our evaluation setup in Sec. 5.1. Secondly, in Sec. 5.2, we provide qualitative and quantitative results for two pairs of domains: 1) synthetic to real (*i.e.*, training on synthetic data and testing on real data from KITTI); and 2) synthetic to synthetic (*i.e.*, training on one synthetic dataset and testing on a different synthetic domain). Finally, in Sec. 5.3 we report qualitative results illustrating our confidence weighted loss. We provide the code needed to implement our framework to ease further research in this field [2].

5.1. Experimental Setup

Datasets. In our experimental evaluation we simulate realistic test conditions, in which no data from the target domain is available. We therefore use training and testing data sampled from two completely disjoint datasets. For the real dataset, we use the 71 different sequences of the KITTI-raw dataset [8] (denoted as KITTI) accounting for ∼43K images with sparse depth annotations provided by [31].

For the synthetic dataset, we have used the FlyingThings3D dataset [15] (shortened F3D) to perform an initial training of the networks from random initialization. Then, we use Synthia [25] as a synthetic dataset containing scenarios similar to KITTI. The dataset is composed of 50 different video sequences rendered in different seasons and weather conditions for a total of ∼45K images. For this dataset we scaled the image to half resolution to bring the disparity into the same range as KITTI.

Finally, using the Carla simulator [4], we have rendered a new dataset (referenced as Carla) composed of 25 different video sequences, each being a thousand frames long, with accurate ground truth data for each frame. Each video sequence is rendered in 15 different weather conditions to add variance to the dataset. Resulting in a total of 375K frames. During the rendering we set up the virtual cameras to match the geometry of the real KITTI dataset (*i.e.*, same baseline, field of view and similar image resolution).

Network Architectures. For the experiments we have selected the Dispnet-Corr1D [15] architecture (shortened to Dispnet). For all the evaluation tests, we pretrain the networks on F3D to obtain a set of weights that will be used as an initialization across all the other tests. We implement the confidence function introduced in Sec. 3.2 as a small three layer fully convolutional CNN with batch normalization. The network takes the re-projection error scaled to quarter resolution as an input and produces an output at the same resolution. The prediction is then scaled to full resolution using bilinear upsampling. More details on the network architectures and on the hyper-parameters used to pretrain them are reported in the supplementary material.

Evaluation Protocol. After an initial offline training, we perform online adaptation and evaluate models on sequences of stereo frames. To test independent adaptations for each sequence, we reset the disparity network to its trained weight configuration at the beginning of each test sequence. Then, for each frame, first, we measure the performance of the current model and then we adapt it by a single step of back-propagation and weight update according to Eq. 1 before moving to the next frame. We do not measure the performance on frames used for adaptation.

Metrics. We measure performance according to both average End Point Error (EPE) and percentage of pixels with disparity error larger than 3 (D1-all). Firstly, we measure both metrics independently for each frame to plot performance as a function of the number of frames processed for adaptation. Secondly, we average over each sequence and finally over all the dataset.

Offline Training. After the initial pretraining on F3D we fine tune the networks on a training set with our learning-to-adapt framework Alg. 2, we use $k = 3$ consecutive frames for each sample and set the learning rates $\alpha = 0.00001$ and $\beta = 0.0001$

Online Adaptation. We use the left-right re-projected unsupervised loss [10] for the adaptation. Optimization is performed using gradient descent with momentum, where the momentum value is set to 0.9 and a learning rate set to 0.0001.

5.2. Results

We evaluate our learning-to-learn method between pairs of datasets, one for training, and one for evaluation. We consider two scenarios: 1) synthetic to real and 2) synthetic to synthetic. We compare the results of our learning-to-adapt framework (**L2A**), and the method trained using a supervised L_1 regression loss (**SL**). Methods performing unsupervised online adaptation at test time are indicated by appending **+Ad** to the training method, and confidence weighted adaptation by **+WAd**. It is worth noting that **SL+Ad** corresponds to the adaptation method proposed in [30].

[2] https://github.com/CVLAB-Unibo/Learning2AdaptForStereo

		Method	Training set	D1-all (%)	EPE	ΔD1	ΔEPE
	(a)	**SL**	-	9.43	1.62	-	-
	(b)	**SL+Ad**[30]	-	7.81	1.44	-1.62	-0.18
	(c)	**SL**	Carla	7.46	1.48	-	-
	(d)	**SL+Ad**[30]	Carla	5.26	1.20	-2.20	-0.28
	(e)	**SL**	Synthia	8.55	1.51	-	-
	(f)	**SL+Ad**[30]	Synthia	5.33	1.19	-3.22	-0.32
Ours	(g)	**L2A**	Carla	8.41	1.51	-	-
	(h)	**L2A+WAd**	Carla	**4.49**	**1.12**	**-3.92**	**-0.39**
	(i)	**L2A**	Synthia	8.22	1.50	-	-
	(j)	**L2A+WAd**	Synthia	4.65	1.14	-3.57	-0.36
	(k)	**SL** (ideal)	KITTI	4.26	1.12	-	-

Table 1. Performance on KITTI for the Dispnet network trained according to different methods after initialization from F3D. It can clearly be seen that online adaptation (**+Ad/+WAd**) provides a significant improvement compared to when it is left out. The best results are obtained when the training is achieved using our **L2A+WAd** framework. Line (k) indicates an upper bound on how well Dispnet can perform when fine tuned on a subset of samples from the target domain. The last two columns indicate the performance improvement with adaptation, and as it is evident in the table, our **L2A+WAd** method obtains the largest increase in performance with adaptation.

5.2.1 Synthetic to Real

The most interesting scenario is the one where training on a synthetic domain is followed by testing on a real-life domain. Specifically, we train on Synthia or Carla and then evaluate on the KITTI dataset.

The results for the Dispnet architecture are provided in Tab. 1. Lines (a) to (f) report the performance when the network weights are obtained in a standard way (using a supervised L_1 loss function). As expected, the network performs poorly when tested on a different domain with respect to the one it was trained on - lines (a, c, e). The use of adaptation for this setup provides a significant improvement - lines (b, d, f) - further motivating the need to adapt to a new domain.

The two rows (h) and (j) report results obtained by learning to adapt on Carla or Synthia using the **L2A+WAd** framework. Our proposed framework clearly outperforms the baseline methods for both training datasets. Comparing lines (h) and (d) clearly shows that our training process is able to learn a model which is better suited for continuous adaptation. The same conclusions hold even for the results with Synthia (lines (j) and (f)). In the last two columns we can observe the relative improvement provided by adaptation for each method. In these results, it is evident that our **L2A+WAd** framework provides the largest increase in accuracy when performing adaptation. Lastly, in line (k), we provide the performance of Dispnet obtained in the ideal scenario where samples from the target domains are available (*i.e.*, KITTI2012 and KITTI2015 training sets) and used to fine tune the base model with a supervised L_1 regression loss. Although having access to such samples would defeat the purpose of our approach, the result listed here ultimately serves as an upper bound on the attainable performance. As shown, our **L2A+WAd** framework obtains competitive results.

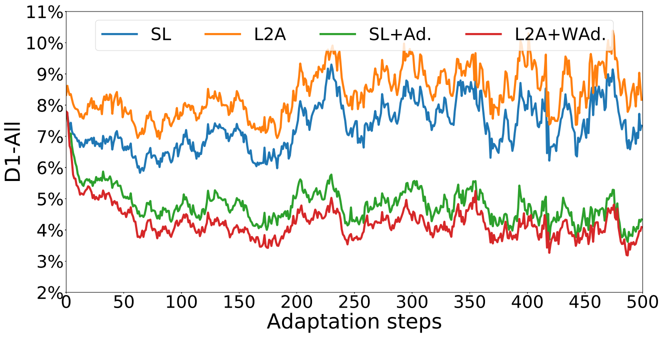


Figure 4. Average D1-all error with respect to the number of adaptation steps performed on the KITTI database by a Dispnet network trained according to supervised learning (**SL**) or our learning to adapt framework (**L2A**).

	Method	Training Set	Ad. Unsupervised		Ad. Supervised	
			D1-all (%)	EPE	D1-all (%)	EPE
(a)	**SL+Ad**[30]	-	26.56	3.96	15.60	2.24
(b)	**SL+Ad**[30]	Carla	25.07	3.62	13.89	1.97
(c)	**L2A+Ad**	Carla	22.69	3.08	**12.01**	**1.80**
(d)	**L2A+WAd**	Carla	**21.07**	**2.90**	✗	✗

Table 2. Comparison of the training methods when evaluated on sequences from Synthia. It can be seen that the best performing training method is **L2A+WAd**. We also provide results for when we use a L_1 supervised adaptation loss. Best results in bold.

Adaptation Performance Over Time: To further highlight the difference of behaviour between models trained to regress and those trained to adapt, we plot the average D1-all error achieved by Dispnet on KITTI as a function of the number of adaptation steps in Fig. 4. The vertical axis represents the average D1-all error of the k^{th} frame in all of the sequences in KITTI. Comparing the methods with and without online adaptation, it is clear that in both cases, adaptation drastically improves the performance. The comparison between **SL+Ad** (green line) and **L2A+WAd** (red line) shows how quickly our method adapts to the given video sequence. The poor results of **L2A** can easily be explained since our formulation never explicitly optimizes the base model for regression. Instead it optimizes the network to quickly learn-to-adapt, therefore the base model results can be sub-optimal, providing the performance can be improved in a few adaptation steps.

5.2.2 Synthetic to Synthetic

Here, we perform a more controlled synthetic-to-synthetic evaluation where we can measure the difference in performance more explicitly thanks to the availability of dense and accurate ground truth labels. The aim of the following series of tests will be to quantify the performance of the two key aspects of the learning-to-adapt framework, namely, learning to adapt through meta-learning and learning to weight noisy loss estimation. To further prove the generality of our learning to adapt formulation, we also provide results when the networks are trained to perform online adaptation using a supervised L_1 loss (*i.e.*, $\mathcal{L}_u \equiv \mathcal{L}_s$).

For these tests, we again use Dispnet trained on Carla but tested on all the sequences of the full Synthia dataset. Specifically, to show that we can adapt using different loss functions, we train for both unsupervised and supervised adaptation[3], and evaluate the performance of the following training scenarios: (a) Using the initial model trained using F3D; (b) Training on Carla using a supervised L_1 loss; (c) Using the learning-to-adapt framework **without** confidence weighted loss; (d) Using the learning-to-adapt framework **with** confidence weighted loss.

We report the results in Tab. 2, where it can be seen that explicitly training Dispnet to adapt using our learning-to-learn formulation (c), allows the network to exploit the online adaptation and greatly improve the performance both in the unsupervised and supervised adaptation setups. Finally, it can also be seen that weighting the unsupervised loss values results in superior performance (d). For this test set up, the results clearly demonstrate how our formulation is able to learn a weight configuration that is more inclined to be adapted to a new environment.

5.3. Confidence Weighted Loss Function

In Fig. 5, we show a visualization of the confidence masks and weighted errors optimized by our confidence guided adaptation loss described in Sec. 3.2. A quantitative evaluation is not possible due to the unavailability of the corresponding ground-truth data and obtaining it is not straightforward. The predicted confidence maps effectively mask out occluded regions in the image while keeping the useful error signals in the rest of the image (low confidence areas are encoded as dark pixels). Errors on occluded regions, *e.g.*, to the left of the traffic sign in the left column or to the left of the car in the right column, are effectively masked out, producing a cleaner error estimation that will improve adaptation performances. We wish to highlight that the confidence network has been trained without any direct supervision and only on Carla, nevertheless, it seems to be able to generalize well to KITTI. We believe this ability to generalize is mainly due to the avoidance of working directly with RGB inputs, which inevitably change drastically between datasets. Instead, the confidence network relies on the estimated re-projection error, which is more consistent across different environments.

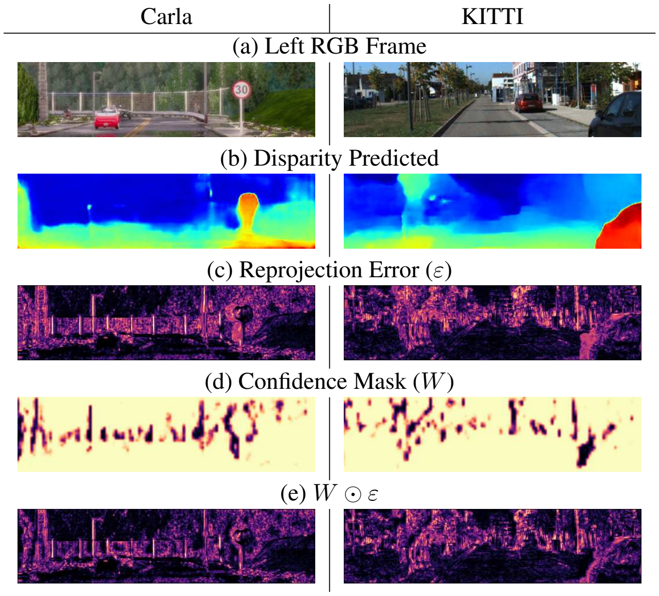

Carla
KITTI
(a) Left RGB Frame
(b) Disparity Predicted
(c) Reprojection Error (ε)
(d) Confidence Mask (W)
(e) $W \odot \varepsilon$

Figure 5. Visualization of the errors optimized to achieve unsupervised adaptation with reprojection based loss function and using our weighting function. Brighter colours encode higher values.

6. Discussion

We have introduced a learning to adapt framework for stereo and demonstrated how the performance of deep stereo networks can be improved by explicitly training the network to be suited for adaptation. Moreover, we are able to automatically learn an implicit confidence measure, for noisy unsupervised error estimation, directly in our learning-to-adapt framework. Specifically, we showed the ability of a Dispnet [15] network to adapt to a real and synthetic domain, when training is performed on a different synthetic domain. In this setting, we obtained increased performance when applying our learning-to-adapt formulation. In future, we plan to test this framework on more complex network architectures (*e.g.*, [13, 34]) and to extend it to use different unsupervised loss functions for online adaptation (*e.g.*, the improved re-projection loss described in [33]).

7. Acknowledgement

This work was supported by the ERC grant ERC-2012-AdG 321162-HELIOS, EPSRC grant Seebibyte EP/M013774/1, EPSRC/MURI grant EP/N019474/1 and TOSHIBA Research. We would also like to acknowledge the Royal Academy of Engineering and FiveAI.

[3] In online adaptation we use the L_1 loss between the predicted disparity and the ground truth annotations for each stereo pair.

References

[1] Maruan Al-Shedivat, Trapit Bansal, Yura Burda, Ilya Sutskever, Igor Mordatch, and Pieter Abbeel. Continuous adaptation via meta-learning in nonstationary and competitive environments. In *International Conference on Learning Representations*, 2018. 1, 6

[2] Jia-Ren Chang and Yong-Sheng Chen. Pyramid stereo matching network. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 5

[3] Ignasi Clavera, Anusha Nagabandi, Simin Liu, Ronald S. Fearing, Pieter Abbeel, Sergey Levine, and Chelsea Finn. Learning to adapt in dynamic, real-world environments through meta-reinforcement learning. In *International Conference on Learning Representations*, 2019. 1, 6

[4] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. CARLA: An open urban driving simulator. In *Proceedings of the 1st Annual Conference on Robot Learning*, pages 1–16, 2017. 2, 6

[5] 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. 1, 2, 3, 5, 6

[6] Chelsea Finn and Sergey Levine. Meta-learning and universality: Deep representations and gradient descent can approximate any learning algorithm. *ICLR*, 2018. 1

[7] Ravi Garg, Vijay Kumar BG, Gustavo Carneiro, and Ian Reid. Unsupervised cnn for single view depth estimation: Geometry to the rescue. In *European Conference on Computer Vision*, pages 740–756. Springer, 2016. 1, 4

[8] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The kitti dataset. *International Journal of Robotics Research (IJRR)*, 2013. 6

[9] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 3354–3361. IEEE, 2012. 2

[10] 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, 5, 6

[11] Lu Jiang, Zhengyuan Zhou, Thomas Leung, Li-Jia Li, and Li Fei-Fei. Mentornet: Regularizing very deep neural networks on corrupted labels. In *Procedings of the International Conference on Machine Learning (ICML)*, 2018. 5

[12] Zequn Jie, Pengfei Wang, Yonggen Ling, Bo Zhao, Yunchao Wei, Jiashi Feng, and Wei Liu. Left-right comparative recurrent model for stereo matching. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 5

[13] Alex Kendall, Hayk Martirosyan, Saumitro Dasgupta, Peter Henry, Ryan Kennedy, Abraham Bachrach, and Adam Bry. End-to-end learning of geometry and context for deep stereo regression. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017. 5, 8

[14] Zhengfa Liang, Yiliu Feng, Yulan Guo Hengzhu Liu Wei Chen, and Linbo Qiao Li Zhou Jianfeng Zhang. Learning for disparity estimation through feature constancy. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 5

[15] 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, 2, 5, 6, 8

[16] Stephan Meister, Bernd Jähne, and Daniel Kondermann. Outdoor stereo camera system for the generation of real-world benchmark data sets. *Optical Engineering*, 51(2):021107, 2012. 2

[17] Moritz Menze and Andreas Geiger. Object scene flow for autonomous vehicles. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015. 1, 2

[18] Nikhil Mishra, Mostafa Rohaninejad, Xi Chen, and Pieter Abbeel. A simple neural attentive meta-learner. In *In International Conference on Learning Representations (ICLR)*, 2018. 5

[19] Devang K Naik and RJ Mammone. Meta-neural networks that learn by learning. In *Neural Networks, 1992. IJCNN., International Joint Conference on*, volume 1, pages 437–442. IEEE, 1992. 5

[20] Jiahao Pang, Wenxiu Sun, Jimmy SJ. Ren, Chengxi Yang, and Qiong Yan. Cascade residual learning: A two-stage convolutional neural network for stereo matching. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017. 5

[21] Jiahao Pang, Wenxiu Sun, Chengxi Yang, Jimmy Ren, Ruichao Xiao, Jin Zeng, and Liang Lin. Zoom and learn: Generalizing deep stereo matching to novel domains. *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 1, 5

[22] Matteo Poggi, Fabio Tosi, and Stefano Mattoccia. Learning monocular depth estimation with unsupervised trinocular assumptions. In *6th International Conference on 3D Vision (3DV)*, 2018. 5

[23] Sachin Ravi and Hugo Larochelle. Optimization as a model for few-shot learning. In *In International Conference on Learning Representations (ICLR)*, 2017. 5

[24] Mengye Ren, Wenyuan Zeng, Bin Yang, and Raquel Urtasun. Learning to reweight examples for robust deep learning. In *Procedings of the International Conference on Machine Learning (ICML)*, 2018. 5

[25] 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. 2, 6

[26] Jürgen Schmidhuber. *Evolutionary principles in self-referential learning, or on learning how to learn: the meta-meta-... hook*. PhD thesis, Technische Universität München, 1987. 5

[27] Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. In *Advances in Neural Information Processing Systems*, pages 4077–4087, 2017. 5

[28] Sebastian Thrun and Lorien Pratt. Learning to learn: Introduction and overview. In *Learning to learn*, pages 3–17. Springer, 1998. 5

[29] Alessio Tonioni, Matteo Poggi, Stefano Mattoccia, and Luigi Di Stefano. Unsupervised adaptation for deep stereo. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017. 1, 5

[30] 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. 1, 2, 5, 6, 7

[31] Jonas Uhrig, Nick Schneider, Lukas Schneider, Uwe Franke, Thomas Brox, and Andreas Geiger. Sparsity invariant cnns. In *International Conference on 3D Vision (3DV)*, 2017. 2, 6

[32] Oriol Vinyals, Charles Blundell, Tim Lillicrap, Daan Wierstra, et al. Matching networks for one shot learning. In *Advances in Neural Information Processing Systems*, pages 3630–3638, 2016. 5

[33] Yinda Zhang, Sameh Khamis, Christoph Rhemann, Julien Valentin, Adarsh Kowdle, Vladimir Tankovich, Michael Schoenberg, Shahram Izadi, Thomas Funkhouser, and Sean Fanello. Activestereonet: End-to-end self-supervised learning for active stereo systems. In *15th European Conference on Computer Vision (ECCV 2018)*, 2018. 5, 8

[34] Yiran Zhong, Hongdong Li, and Yuchao Dai. Open-world stereo video matching with deep rnn. In *15th European Conference on Computer Vision (ECCV 2018)*, 2018. 1, 2, 5, 8

[35] Chao Zhou, Hong Zhang, Xiaoyong Shen, and Jiaya Jia. Unsupervised learning of stereo matching. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1567–1575, 2017. 5