CoMoDA: Continuous Monocular Depth Adaptation Using Past Experiences

Yevhen Kuznietsov[1] Marc Proesmans[1] Luc Van Gool[1,2]
[1]KU Leuven/ESAT-PSI [2]ETH Zurich/CVL
{yevhen.kuznietsov, marc.proesmans, luc.vangool}@esat.kuleuven.be

Abstract

While ground truth depth data remains hard to obtain, self-supervised monocular depth estimation methods enjoy growing attention. Much research in this area aims at improving loss functions or network architectures. Most works, however, do not leverage self-supervision to its full potential. They stick to the standard closed world train-test pipeline, assuming the network parameters to be fixed after the training is finished. Such an assumption does not allow to adapt to new scenes, whereas with self-supervision this becomes possible without extra annotations.

*In this paper, we propose a novel self-supervised Continuous Monocular Depth Adaptation method (**CoMoDA**), which adapts the pretrained model on a test video on the fly. As opposed to existing test-time refinement methods that use isolated frame triplets, we opt for continuous adaptation, making use of the previous experience from the same scene. We additionally augment the proposed procedure with the experience from the distant past, preventing the model from overfitting and thus forgetting already learnt information.*

We demonstrate that our method can be used for both intra- and cross-dataset adaptation. By adapting the model from train to test set of the Eigen split of KITTI, we achieve state-of-the-art depth estimation performance and surpass all existing methods using standard architectures. We also show that our method runs 15 times faster than existing test-time refinement methods. The code is available at https://github.com/Yevkuzn/CoMoDA.

1. Introduction

Depth estimation is a fundamental component of any application involving navigation (e.g., autonomous driving or robotics). Prediction of depth maps from a single image is, however, an ill-posed problem, due to the lack of information on the three-dimensional structure of the scene. Despite this fact, humans perform remarkably well at estimating distances to objects from a single view. Recently, noticeable progress was achieved in the development of learning-based depth estimation methods, which presumably make use of some latent depth cues also utilized by humans.

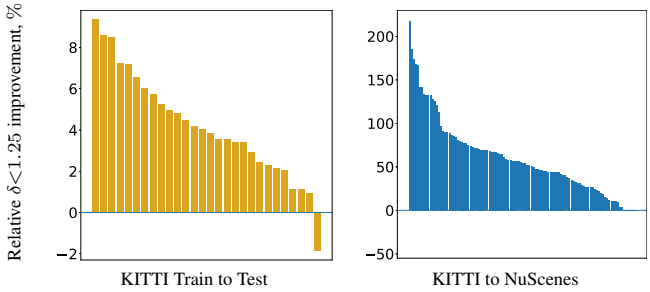

Figure 1. Relative $\delta < 1.25$ improvement achieved by CoMoDA on different test videos when adapting from train to test set of KITTI [11] (intra-dataset) and from KITTI to NuScenes [3] (cross-dataset). The horizontal axis represents the test videos sorted by performance gain.

1.1. Supervised methods

Learning depth in a supervised way requires significant amounts of ground truth, which is usually obtained by RGB-D cameras or LIDARs. One of the first supervised learning-based approaches to depth estimation was proposed by Saxena *et al*. [33]. Eigen *et al*. [8] employed a deep neural network to learn depth end-to-end. Laina *et al*. [21] proposed a stronger encoder-decoder architecture combined with a BerHu loss. Other notable improvements are based on architectural refinements [9], the use of temporal cues [47] and cross-modal (e.g. semantics, normals) consistencies [48, 44].

The ground truth depth maps produced by the sensors may show several imperfections, such as sparsity, measurement noise, the absence of measurements due reflective surfaces, etc. In order to overcome this issue, Kuznietsov *et al*. [20] proposed to leverage stereo cues as an additional supervisory signal during training.

In outdoor scenarios, the high cost of LIDAR is another drawback. To alleviate the need for metric depth ground truth, Chen *et al*. [6] propose to use annotated depth rankings instead. Alternatively, synthetic data can be used to reduce the amount of ground truth required for training. Numerous methods [25, 51, 28, 2, 17] utilize domain adapta-

tion techniques to close the gap between synthetic and real predictions.

1.2. Self-Supervised Methods

As the collection of depth data is cumbersome, and involves the use of specifically designed sensors, researchers have begun to explore more viable alternatives to train depth prediction networks. Xie *et al*. [41] formulated the depth estimation task as a view synthesis problem, which requires only stereo image pairs as a supervisory signal. As stereo images are arguably more easy to acquire than LIDAR, this poses an interesting alternative to the supervised depth estimation methods from Section 1.1. A follow-up work by Garg *et al*. [10] extended the method from [41] by regressing continuous depth instead of discrete values. Additionally, they proposed to add a smoothness regularization term. Godard *et al*. [12] further improved these results by adding left-right consistency and SSIM [39] terms to the image reconstruction loss. Notable improvements were achieved by enforcing left-right consistency between intermediate feature maps [46], cycle-consistency [29, 40], or using temporal cues [22, 46]. Some other self-supervised methods [30, 1] utilize GANs [14] to improve image synthesis, thus also improving the predicted depth quality.

Zhou *et al*. [53] propose to reduce the supervision to monocular videos by reconstructing a video frame from its neighbors. Since there is no fixed known baseline in this problem setup, they train depth and ego-motion networks simultaneously.

Supervision with monocular videos allows to use large amounts of low cost or even free data for training. However, with the availability of data also come the following limitations: (1) the monocular supervision is disturbed by moving objects; (2) scale-ambiguity, due to unknown transformations between frames.

To overcome the problem of moving objects, [53] learned to estimate an explainability mask, that is supposed to filter out the supervisory signal from dynamic objects or areas reprojected outside of the image. A similar method was developed by Vijayanarasimhan *et al*. [37]. Unlike [53], they attempt to train a network for estimating motion for a predefined number of objects. Yin and Shi [45], train a residual flow network to refine the rigid flow estimated from the predicted depth in an end-to-end manner. This improves flow estimation, however, similarly to [53] and [37], has little or no effect on moving objects.

Godard *et al*. [13] introduce significant improvements to the loss function by handling occlusions and neglecting supervision from image areas that do not change significantly over time. The latter noticeably reduces the negative effect of objects moving similarly to the ego-vehicle. Casser *et al*. [5] and Gordon *et al*. [15] make use of a pretrained MaskRCNN [32] to produce masks of possibly moving objects and predict the motion for those separately. [7, 31, 54, 26] additionally train the optical flow network, and utilize the discrepancy between the estimated non-rigid flow and the flow from depth to suppress the effects of moving objects.

Scale ambiguity limits the applicability of the methods supervised with monocular videos to real-world scenarios, where absolute measurements are desired. To produce absolute depth, these methods require depth priors. Such priors are normally not available without the use of additional depth sensors. Guizilini *et al*. [16], along with architectural improvements, proposed to use a velocity supervision term to enable the learning of scale.

Due to the intensive research of self-supervised depth estimation, the gap to supervised performance is getting smaller. Yet, the previous methods mainly focus on improving the model training, and leave the test-time underexplored. In particular: (1) the use of video data beyond training is currently limited to architectural (RNNs [36]); (2) only few methods [7, 5] use self-supervision at test time to improve their predictions, but they tune their models on isolated frame triplets only, resetting the model parameters for every new frame[1]. Without weight reset, the performance drops substantially.

Similarly to this work, Li *et al*. [23] and Zhang *et al*. [50] proposed to adapt depth and odometry continuously. However, the experimental setup of these concurrent works is different from ours, which makes the direct comparison to them not possible.

1.3. Our Contribution

Inspired by continual learning and the recent advances in stereo matching [34, 35, 52, 49], we leverage both self-supervision and long-term video information at test time. More specifically:

- we propose to adapt depth and ego-motion estimators continuously on monocular videos (in contrast to the isolated fine-tuning [5, 7]);
- we introduce a 'baseline' adaptation procedure adopted from stereo matching approaches, yet we observe it is sensitive to overfitting to short video segments;
- so next, we incorporate the experience from the distant past in a replay buffer fashion [24], effectively mitigating the consequences of overfitting and thus stabilizing the adaptation.

Adapting the model continuously allows us to benefit from the data previously observed in a video and perform only one model update per frame, which results in an order of magnitude faster runtime compared to [5, 7]. We perform an extensive evaluation of our method and demonstrate the

[1]The weight reset is not mentioned in [5], however, we refer the reader to the official code repository (cfr [4] , l.321)

state-of-the-art performance among depth estimation approaches supervised with monocular videos.

2. Method

In this section we provide a detailed overview of the proposed CoMoDA method. We start with explaining the background for self-supervised depth estimation with monocular videos.

2.1. Warping

Given a video recorded by a moving camera, a reconstruction $\hat{\mathbf{I}}_{\mathbf{s}\to\mathbf{t}}$ of a target frame $\mathbf{I_t}$ at time t can be obtained from a source frame $\mathbf{I_s}$ at time s by bilinear interpolation over the reprojected image coordinates (also known as warping):

$$\hat{\mathbf{I}}_{\mathbf{s}\to\mathbf{t}}(\mathbf{p_t}) = \mathbf{I_s}(\hat{\mathbf{p}}_\mathbf{s}), \tag{1}$$

where $\hat{\mathbf{p}}_\mathbf{s}$ is the reprojection of point $\mathbf{p_t}$ into frame $\mathbf{I_s}$. To obtain the mapping from $\mathbf{p_t}$ to $\hat{\mathbf{p}}_\mathbf{s}$, $\mathbf{p_t}$ is first backprojected into 3D point $\mathbf{P}$ using camera intrinsics matrix $\mathbf{K}$ and the depth map $\mathbf{D_t}$ corresponding to $\mathbf{I_t}$. Then $\mathbf{P}$ is transformed to account for camera motion $\mathbf{M_{t\to s}}$ and projected onto the image plain:

$$\hat{\mathbf{p}}_\mathbf{s} \sim \mathbf{K}\mathbf{M}_{\mathbf{t}\to\mathbf{s}}\underbrace{\mathbf{D_t}(\mathbf{p_t})\mathbf{K}^{-1}}_{\mathbf{P}} \tag{2}$$

2.2. Loss

Zhou *et al*. [53] propose to optimize depth estimator $\mathcal{E}_D(\mathbf{I_s}) = \mathbf{D_s}$ and ego-motion estimator $\mathcal{E}_M(\mathbf{I_t}, \mathbf{I_s}) = \mathbf{M_{t\to s}}$ simultaneously, using the discrepancy between $\mathbf{I_t}$ and $\hat{\mathbf{I}}_{\mathbf{s}\to\mathbf{t}}$ as a supervisory signal. The typical inputs for the corresponding loss are the triplet of consecutive video frames, the depth map $\mathbf{D}$ predicted for the central frame $\mathbf{t}$ and the reconstructions of $\mathbf{I_t}$ obtained from its neighbors $\mathbf{I_s}$ via warping.

We build our loss upon Monodepth2 [13] as one of the best performing and reliably reproducible depth estimation methods. In order to obtain absolute depth, we augment their loss with a velocity supervision term from [16]. In driving scenarios, such an additional supervision is usually possible without extra expense, since most cars already have velocity measuring systems on board. Our total loss is formalized by Eq. 3:

$$\mathcal{L} = \mathcal{L}_{rec} + \gamma\mathcal{L}_{sm} + \lambda\mathcal{L}_{vel} \tag{3}$$

Image Reconstruction Loss. $\mathcal{L}_{rec}$ used by [13] can be defined as auto-masking μ_{auto} applied to minimum reprojection error $\mathcal{L}_{min}$:

$$\mathcal{L}_{rec} = \mu_{auto} \cdot \mathcal{L}_{min}, \tag{4}$$

where $\mathcal{L}_{min}$ is the per-pixel minimum over photoconsistency errors $\mathcal{L}_p$ between target image $\mathbf{I_t}$ and its reconstructions $\hat{\mathbf{I}}_{\mathbf{s}\to\mathbf{t}}$:

$$\mathcal{L}_{min} = \min_{\mathbf{s}} \mathcal{L}_p(\mathbf{I_t}, \hat{\mathbf{I}}_{\mathbf{s}\to\mathbf{t}}) \tag{5}$$

This term diminishes the negative effect of occlusions. If an object is occluded in only one of the source images $\mathbf{I_s}$, one can assume that the reconstruction affected by occlusion is less similar to the target $\mathbf{I_t}$, implying higher photoconsistency error.

Auto-masking μ_{auto} is designed to suppress the trivial supervisory signal from stationary or almost stationary image regions (e.g., from objects moving similarly to ego-vehicle or static scenes). Small appearance changes between the same image region in different frames imply low photoconsistency error $\mathcal{L}_p$ for that region. Thus, $\mathcal{L}_{rec}$ is computed only for those pixels for which holds the right side of Eq. 6:

$$\mu_{auto} = \bigwedge_{\mathbf{s}} [\mathcal{L}_{min} < \mathcal{L}_p(\mathbf{I_s}, \mathbf{I_t})] \tag{6}$$

For the photoconsistency error, we use the standard image dissimilarity measure $\mathcal{L}_p$, as introduced by [12]:

$$\mathcal{L}_p(\mathbf{I}, \hat{\mathbf{I}}) = \alpha\frac{1-SSIM(\mathbf{I},\hat{\mathbf{I}})}{2} + (1-\alpha)|\mathbf{I}-\hat{\mathbf{I}}|, \tag{7}$$

where $SSIM$ is the pixel-wise variant of the widely-used Structure Similarity Image Matching index [39].

Smoothness Term. $\mathcal{L}_{sm}$ is typically used to regularize the predicted depth in the textureless areas. It penalizes depth discontinuities in continuous image regions:

$$\mathcal{L}_{sm} = [\partial_x\mathbf{D}]e^{-\partial_x\mathbf{I}} + [\partial_y\mathbf{D}]e^{-\partial_y\mathbf{I}}, \tag{8}$$

where ∂_i denotes the gradient along axis i. This term is computed only for the target frame t.

Velocity Supervision Term. $\mathcal{L}_{vel}$ penalizes the difference between the magnitude of the predicted ego-translation $\mathbf{T}$ and the distance covered by the car:

$$\mathcal{L}_{vel} = \sum_s |\mathbf{T}_{\mathbf{s}\to\mathbf{t}} - \Delta\tau_{\mathbf{s}\to\mathbf{t}}\mathbf{V_s}|, \tag{9}$$

where $\Delta\tau$ is the time difference between the frames, and $\mathbf{V}$ – the instantaneous velocity magnitude measured by a sensor. The term is computed for each source frame $\in \mathbf{s}$.

All computations for smoothness and image reconstruction terms are initially performed independently for different pixels and scales. Then, the losses are added over all scales and averaged over all pixels.

2.3. Continuous Adaptation

Given the depth and ego-motion estimators $\mathcal{E}^\theta_D(\mathbf{I}) = \mathbf{D}$ and $\mathcal{E}^\theta_M(\mathbf{I_i}, \mathbf{I_j}) = \mathbf{M_{i\to j}}$ pretrained on the source set of data $\mathbf{\Omega}$ with the loss described in Section 2.2, the model is adapted on a target monocular video ν ($\nu \cap \mathbf{\Omega} = \emptyset$) on the fly. More specifically, for each frame in the video, the depth map is produced before accessing any following frames.

For simplicity, we will first describe our baseline (gray area of Figure 2). A similar adaptation algorithm works off-the-shelf for stereo matching approaches [52]. Yet, it has not been previously applied to the monocular setup.

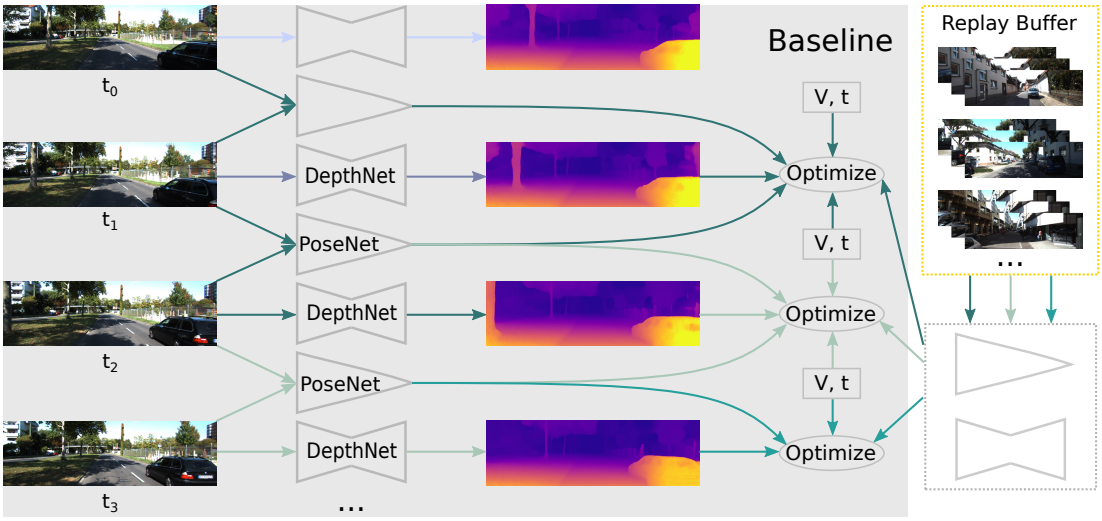

Figure 2. Proposed continuous adaptation. Images on the left correspond to the video on which the adaptation is performed. Different colors represent the operations performed at different iterations of the adaptation. We omit motion handling for simplicity.

Baseline. The depth estimates for the two first frames are produced without adaptation. For every frame t in the video (starting from third), the baseline works as follows:

1. Estimate the camera motion between the previous frame and its two neighbors – $\mathbf{M_{t\text{-}1\to t}}$, $\mathbf{M_{t\text{-}2\to t\text{-}1}}$;
2. Check if the minimum motion constraints are satisfied (e.g., ego-translations from Step 1 exceed some value). If not, the supervisory signal might be close to trivial, proceed to Step 5;
3. Reconstruct the frame $t-1$ (as described in the Section 2.1) from its neighbors t and $t-2$ using the ego-motion estimates from Step 1, and the depth previously inferred for $t-1$;
4. Compute the loss from Eq. 3 and update the network parameters θ via backpropagation;
5. Estimate the depth map for the current frame t (this estimate is further used for evaluation). Proceed to the next frame.

CoMoDA. In our experiments we observe that the supervision obtained from monocular videos is too unstable to achieve any performance improvement using the previously described baseline (see Section 4.2). In order to overcome this issue, we introduce a replay buffer as a crucial element of monocular adaptation, augmenting the supervisory signal of the baseline with the experience from the distant past. In particular, every iteration of our method (Fig. 2) is performed using a batch of samples. Every batch consists of a single sample (velocity, time data and a triplet of consecutive images) from the test video ν and one or several samples randomly drawn from the replay buffer with the data previously observed by the model (e.g. during pretraining). As demonstrated in Section 4.2, the replay buffer stabilizes the adaptation substantially.

While auto-masking (Eq. 6) does not fully prevent moving objects from disrupting the adaptation (see Section 4.2), we employ an additional motion handling mechanism. Similarly to [5] and [7], it enhances the performance of our method, in particular the predictions for moving objects. Since the way we handle motion is not essential for showing the benefits of our method, we incorporate the most primitive strategy - masking out potentially moving objects from the total loss using semantic information. This allows us to achieve plausible depth predictions for dynamic objects.

3. Experimental Setup

In this section, we describe the setup used for the experimental validation of our method. The adaptation is performed from a source set of data to a target set of test videos. We adapt the model independently on every video in the way described in Section 2. In the beginning of each video, the model is initialized from the one pretrained on the source data. Such a separation prevents the order in which we adapt on the videos from affecting the evaluation.

To be clear, this setup is completely different from training on the test set. When producing a depth estimate for a particular frame of a particular test video, our method neither has the future frame access, nor the access to the images from other test videos.

Datasets. Our method requires videos and either velocity or translation data. Thus, we chose KITTI [11] and NuScenes [3] for our experiments. In the above described setup, we apply our method to two different tasks:

- Intra-dataset adaptation: KITTI Eigen train to test;
- Cross-dataset adaptation: KITTI Eigen train to NuScenes test.

We utilize the commonly used Eigen split [8] as the evaluation set for KITTI. The test set of the Eigen split contains

28 videos of length varying from 28 to 4550 frames. We evaluate the predictions produced during the adaptation using the standard procedure (as in [13]). In all our experiments, KITTI images are resized to 192×640 pixels.

The NuScenes test set consists of 100+ short videos of $\sim$ 240 frames (or 20 seconds) each. First, we evaluate on each test video separately, applying the evaluation (as in [13], but without cropping) to all test frames with available ground truth. Then, we report the metrics averaged over the test videos. We resize all NuScenes images to 256×480 pixels, so that the focal length matches the one of KITTI.

Model Setup. Our pretraining is performed using the same hyper-parameters and network architectures as in Monodepth2 [13]. However, we do not apply depth normalization in order to produce absolute depth. The velocity supervision term weight is empirically set to $\lambda = 0.005$, and only auto-masking is used for motion handling.

For the test-time adaptation, we use the same hyper-parameters as during the pretraining, but freeze the batch normalization layers. We mask out the potentially moving objects using both auto-masking and the bounding boxes produced by off-the-shelf YOLOv5m [19] object detection model. No data augmentation is performed, and the network weights are not updated if the magnitude of the translation between any neighboring frames in the test triplet does not exceed 0.2m.

Unless stated otherwise, three samples are drawn from the replay buffer for every iteration of the adaptation. This allows the adaptation to be performed at the frame rate of KITTI. Our method runs at 107ms per frame on an Nvidia GTX 1080Ti (excluding 10ms for the preprocessing by YOLOv5m). Since our model does not use the semantic information for inference, the preprocessing should be done parallel to it, thus, not increasing the runtime.

Unless stated otherwise, we use the entire set of pretraining data for experience replay, resulting in 39810 samples for KITTI. Drawing samples from the replay buffer introduces a degree of randomness into our method. Therefore, we report our quantitative results averaged over six runs with the samples drawn in a predetermined order.

4. KITTI Train to Test

4.1. Comparison to Other Methods

In this section we provide qualitative and quantitative comparison to other depth estimation methods. In order to show the benefits of the continuous depth adaptation concept, we will focus on comparison to the approaches performing test-time fine-tuning [5, 7]. To produce the fine-tuned predictions for a particular image, these methods make use of its neighboring frames (also future). This implies the need for video data, even if other video benefits are not explored. Thus, the comparison of [5, 7] to our method, which also requires video for the adaptation, is fair. Especially, if taking into account that we do not make use of the future frames while producing predictions.

Table 1 shows the performance of our method compared to other self-supervised approaches on the test set of Eigen split [8] of KITTI [11]. Even though our method is scale-aware, we evaluate it both with and without median rescaling [53] to allow better comparison to the scale-ambiguous methods. In order to restore the scale for a particular test frame, these methods use the median of the ground truth depth map corresponding to that frame. As a note, such a rescaling provides the unfair advantage of known ground truth depth statistics, and, as shown in [16], leads to significantly better quantitative results than using a model trained with velocity supervision.

Our approach surpasses all methods supervised with monocular videos that do not perform test-time network updates (except PackNet-SfM [16] which performs on par with our method). While [16] gains performance predominantly from the architecture improvements, we use a standard backbone (ResNet18 [18]) in all our experiments. Moreover, our method is architecture-agnostic and can utilize the PackNet-SfM architecture for further performance improvement.

Struct2Depth [5]. Compared to the refined results of [5], we demonstrate the superior performance on 5 metrics out of 7, with RMSE and $\delta < 1.25$ improvements reaching 15.6cm and 2.8% respectively. The use of the previous experience from the same scene allows us to perform 1 backpropagation per frame instead of 20 for [5]. This results in **15** times faster runtime of our method (0.107s vs $\sim$1.6s, or 0.117s vs $\sim$1.75s with the preprocessing time[2] added).

GL-Net [7]. Similarly to [5], we outperform [7] on 5 out of 7 metrics, with RMSE and $\delta < 1.25$ improvements reaching 14.9cm and 1.5% respectively. The quality of the depth maps predicted by [7] is closer to our method than [5]. However, 50 backpropagation iterations per frame performed by [7] result in $\sim$40s runtime, which is **374** times slower than our method (or 342 times slower with the preprocessing time added for our method).

Qualitative Comparison. We additionally demonstrate the advantages of the continuous adaptation by visualizing the depth maps generated by our method and other approaches with code. Fig. 3 shows that we produce the sharpest and the most accurate depth estimates for background objects. For instance, our method is the only to capture the ground altitude variation on the right of the right-most image.

4.2. Ablation Studies

We analyze the performance of our method by assessing the importance of the replay buffer, the additional motion

[2]The preprocessing does not influence our runtime if performed in parallel to inference. [5] require temporally aligned instance masks for their fine-tuning procedure.

Method	Ref	Lower is better				Higher is better		
		Abs Rel	Sq Rel	RMSE	RMSE_{log}	δ<1.25	δ<1.25^2	δ<1.25^3
Zhou *et al.* [53]	-	0.183	1.595	6.709	0.270	0.734	0.902	0.959
Yang *et al.* [43]	-	0.182	1.481	6.501	0.267	0.725	0.906	0.963
LEGO [42]	-	0.162	1.352	6.276	0.252	-	-	-
Mahjourian *et al.* [27]	-	0.163	1.240	6.220	0.250	0.762	0.916	0.968
Wang *et al.* [38]	-	0.151	1.257	5.583	0.228	0.810	0.936	0.974
GeoNet [45]	-	0.149	1.060	5.567	0.226	0.796	0.935	0.975
Zou *et al.* [54]	-	0.150	1.124	5.507	0.223	0.806	0.933	0.973
Ranjan *et al.* [31]	-	0.140	1.070	5.326	0.217	0.826	0.941	0.975
Luo *et al.* [26]	-	0.141	1.029	5.350	0.216	0.816	0.941	0.976
Struct2depth [5]	-	0.141	1.026	5.291	0.215	0.816	0.945	0.979
Gordon *et al.* [15]	-	0.128	0.959	5.23	-	-	-	-
GL-Net [7]	-	0.135	1.070	5.230	0.210	0.841	0.948	0.980
Monodepth2 [13]	-	0.115	0.903	4.863	0.193	0.877	0.959	0.981
Patil *et al.* [36]	-	0.112	0.863	4.730	0.188	0.879	0.960	0.981
Packnet-SfM* [16]	-	0.111	0.829	4.788	0.199	0.864	0.954	0.980
Struct2depth [5]	✓	0.109	0.825	4.750	0.187	0.874	0.958	**0.982**
GL-Net [7]	✓	**0.099**	**0.796**	4.743	0.186	0.884	0.955	0.979
CoMoDA: μ, σ	✓	0.103 7.5e-4	0.862 4.1e-3	**4.594** 1.9e-2	**0.183** 8.9e-4	**0.899** 9.8e-4	**0.961** 1.2e-3	0.981 5.2e-4
CoMoDA*: μ, σ	✓	0.117 1.0e-3	0.873 5.0e-3	4.753 1.6e-2	0.200 1.0e-3	0.877 1.0e-3	0.956 1.2e-3	0.979 7.5e-4

Table 1. Quantitative results of self-supervised depth estimation methods on the test set of Eigen split [8] of KITTI [11]. Ref indicates the use of model refinement or adaptation at test time. * means that the median rescaling [53] was not applied for evaluation. Bold and underlined numbers indicate the best results obtained with and without median rescaling respectively. For our method, we report mean and standard deviation on every metric.

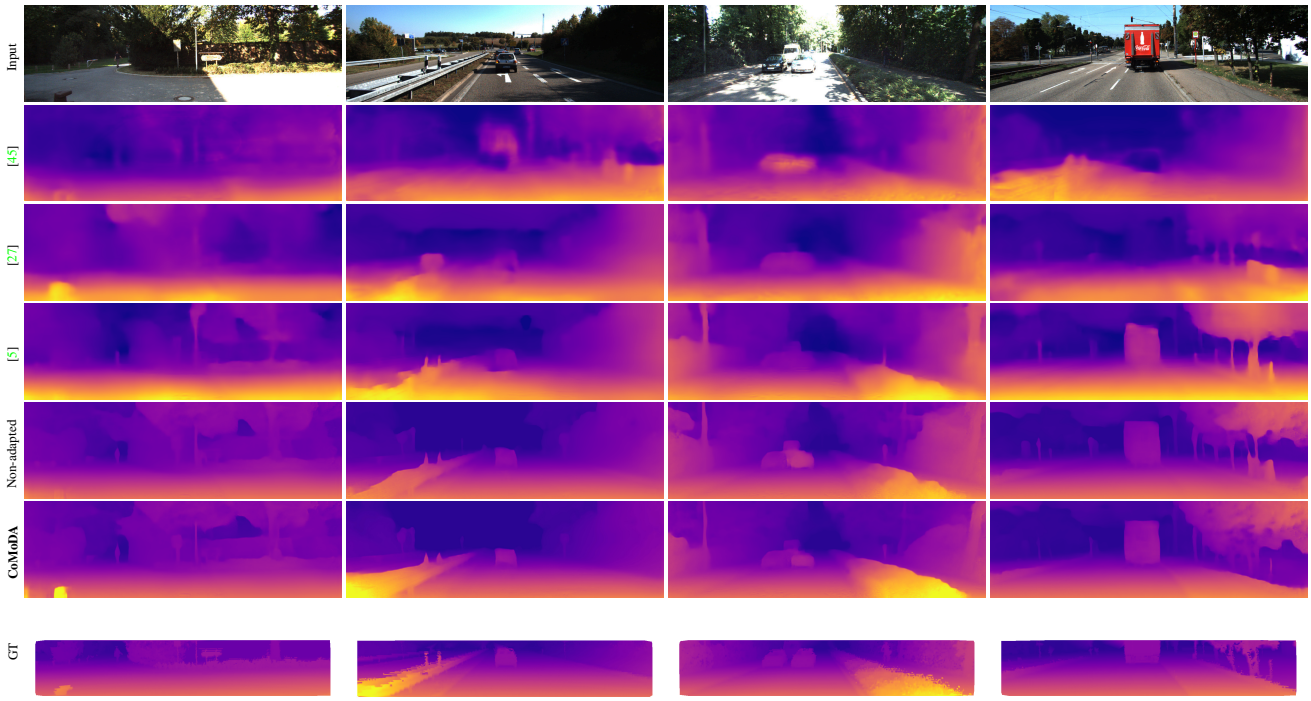

Figure 3. Qualitative results of self-supervised methods on Eigen test split [8] of KITTI [11]. We do not visualize the predictions of Monodepth2 [13], since our non-adapted model is a scale-aware variant of it. GT stands for the interpolated depth ground truth.

handling, and the effect of velocity supervision on monocular test-time adaptation on KITTI.

Table 2 shows that the baseline adaptation (no replay buffer and motion handling) has a negative influence on depth prediction. We assume there are two reasons behind it: 1) the effect of moving objects and 2) the instability caused by overfitting.

Motion Handling. In our experiments, we observe that the auto-masking has a smaller effect on moving objects during adaptation (see Fig. 4). We assume this is caused by moving objects affecting every iteration of the procedure from appearance till vanishing (that can be hundreds

Experiment	Lower is better				Higher is better		
	Abs Rel	Sq Rel	RMS	RMS_{log}	δ<1.25	δ<1.25^2	δ<1.25^3
No adaptation	0.129	0.947	5.199	0.219	0.841	0.946	0.976
Baseline	0.266	3.182	7.551	0.698	0.690	0.801	0.837
No RB	0.164	1.537	5.711	0.291	0.801	0.914	0.950
No MH	0.118	0.885	4.803	0.202	0.876	0.955	0.978
CoMoDA	**0.117**	**0.873**	**4.753**	**0.200**	**0.877**	**0.956**	**0.979**

Table 2. Effect of various design choices evaluated on KITTI [11]. RB stands for the replay buffer, and MH – for the motion handling.

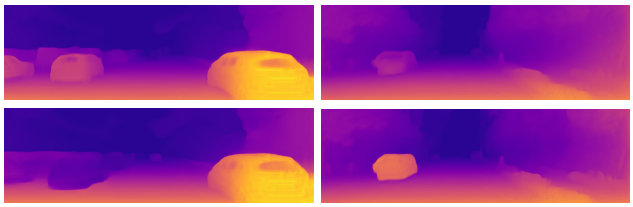

Figure 4. Examples of the predictions for adaptation with the additional motion handling (top) and without (bottom). The automasking and the replay buffer are used in both cases.

of frames), while being scarce during the normal training. Table 2 shows that the additional motion handling has a significant effect on the baseline adaptation. If used together with the replay buffer, the effect of the motion handling on the quantitative results becomes smaller, probably due to the use of pretraining samples with few moving objects. However, as shown in Fig. 4, it still plays a role in maintaining the quality of the depth estimates for moving objects.

Replay Buffer. Even with the supervision from moving objects suppressed, we notice that the adaptation without the replay buffer does not improve over the non-adapted model. During adaptation, the model is supervised with many consecutive video frames, which look very similar most of the time. This is very different from training, when the images used for the supervision are randomly drawn from the entire training set, and can cause the monocular adaptation to overfit to short video segments.

To support our claim, we visualise the error (RMSE) evolution on several test videos of KITTI (see Fig. 5). We observe that the adaptation without the replay buffer often starts performing worse than the non-adapted model when there are significant changes in observed scenes, e.g. when the car turns or starts driving on the bridge. For instance, the adaptation with the additional motion handling degrades when the car turns in video 2011_09_26_0117. Such a behaviour implies that the model sometimes overfits to short video segments with low image variation, suffering from the noticeable changes in the distribution of the supervisory data in the future.

To mitigate possible overfitting, we augment the adaptation with the experience from the distant past. As shown in Table 2 and Fig. 5, the use of the replay buffer effectively stabilizes the adaptation procedure.

Compared to the best of all experiments without replay buffer CoMoDA achieves:

- $> 15cm$ mean RMSE decrease for 18 videos;
- $< 15cm$ mean RMSE decrease for 5 videos;
- $< 15cm$ mean RMSE increase for 4 videos;
- $> 15cm$ mean RMSE increase for 1 video.

It only performs noticeably worse than the adaptation without the replay buffer on the shortest video 2011_09_26_0048, with a duration of about 2 seconds. Such behaviour can be explained by the fact that CoMoDA is augmented with the samples from past and trades short-term adaptation performance in for stability. Figure 1 shows that CoMoDA improves over the non-adapted model on all videos except 2011_09_26_0052. This video is short, with almost stationary ego-vehicle and many other cars.

We additionally show (Fig. 6) that our method is robust to the choice of hyperparameters such as the replay buffer size and the number of samples drawn from the replay buffer for every iteration.

Experiment	Lower is better				Higher is better		
	Abs Rel	Sq Rel	RMS	RMS_{log}	δ<1.25	δ<1.25^2	δ<1.25^3
-Vel*	0.135	0.924	4.920	0.217	0.845	0.953	0.977
-Vel$^+$	0.124	0.885	4.802	0.207	0.869	0.955	0.978
-Vel	0.102	0.871	4.596	0.183	0.898	0.961	0.981
CoMoDA*	0.117	0.873	4.753	0.200	0.877	0.956	0.979
CoMoDA	0.103	0.862	4.594	0.183	0.899	0.961	0.981

Table 3. The effect of velocity supervision on the adaptation. -Vel indicates that velocity supervision is disabled, while the superscript indicates the type of depth rescaling used for evaluation. No superscript corresponds to median rescaling [53], * indicates no rescaling, and $^+$ – that the predictions were rescaled with the depth inferred by the non-adapted model.

Velocity Supervision. Table 3 shows the effect of velocity supervision on the adaptation. If median rescaling is applied, this effect is negligible. Without median rescaling, however, the model adapted without velocity supervision performs significantly worse. Interestingly, if we keep the scale of the depth predictions during the adaptation (by using the median of the non-adapted model), the results do not get much worse compared to those with velocity supervision. Based on these observations, we conclude that velocity supervision does not influence structural adaptation, but allows our method to slightly adapt the scale.

5. KITTI to NuScenes

Table 4 and Figure 1 (left part) show that CoMoDA dramatically improves over the non-adapted model in this setup. Yet, the variant of the adaptation without the replay buffer suffers from overfitting, which results in the increased Abs Rel and the average RMSE not improving towards the ends of videos. While the typical pattern of error evolution affected by overfitting can be observed in video 0549 (Figure 7), we explain this effect in detail in Section 4.2.

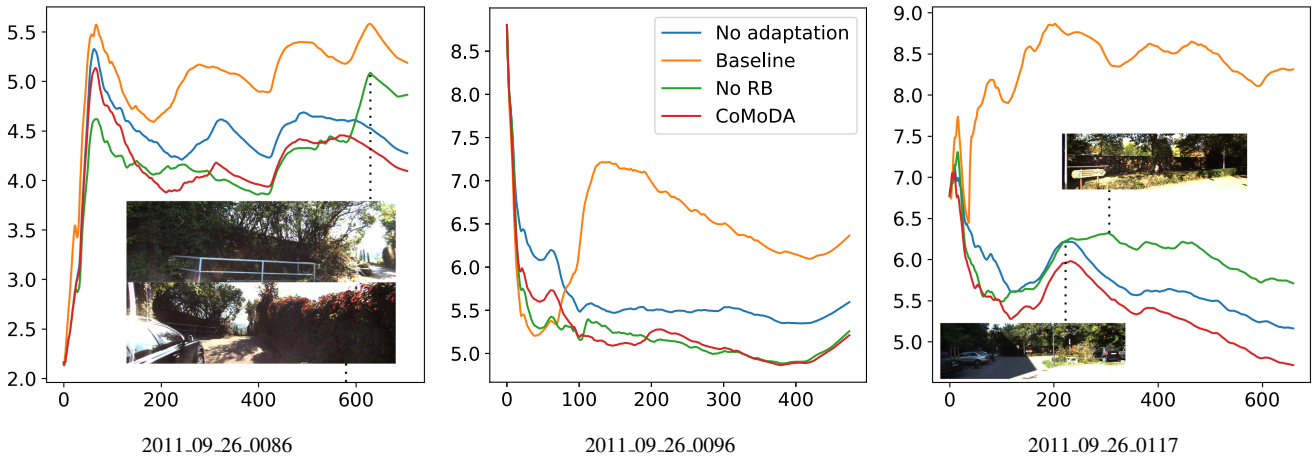

Figure 5. The effect of the adaptation on moving mean RMSE on the test videos of KITTI [11]. Horizontal axis represents video frames, vertical – RMSE in meters. Instead of showing RMSE at a particular time step, we show it averaged over all frames up until that time. This makes the graph less noisy and more readable.

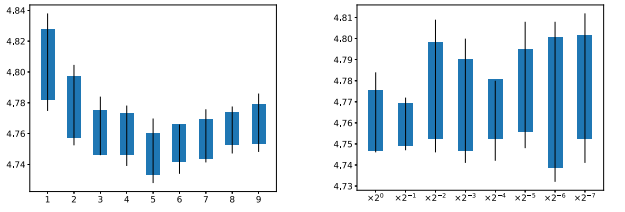

Figure 6. The effect of the replay buffer parameters on the adaptation. Vertical axis represents RMSE in meters. The ends of the sticks correspond to minimum and maximum measurements, while the bars represent the regions within $\mu \pm \sigma$.

Experiment	Lower is better				Higher is better	
	Abs Rel		RMS		δ<1.25	
	100%	20%	100%	20%	100%	20%
No adaptation	0.277	0.277	9.367	9.291	0.434	0.431
No RB	0.289	0.304	8.733	8.964	0.610	0.676
CoMoDA	**0.228**	**0.206**	**7.774**	**7.571**	**0.669**	**0.744**
NuScenes trained	0.131	0.131	6.606	6.674	0.859	0.856

Table 4. Quantitative results on the NuScenes test set evaluated on the whole videos (100%) and only on the last 20% of frames (20%). The bottom row corresponds to the model (monodepth2 + velocity supervision) trained on the NuScenes train set without any adaptation.

On average, our adaptation still falls behind the model trained on NuScenes. However, we observe that CoMoDA already demonstrates reasonably good performance after only 16 seconds of adaptation. In particular, the gap between our method and the model trained on NuScenes noticeably diminishes for all metrics on the last 20% of video frames ($\sim$ 4 last seconds). This encourages us to assume even better adaptation performance given the availability of longer videos. To support this, we provide additional results and analysis in the supplementary materials.

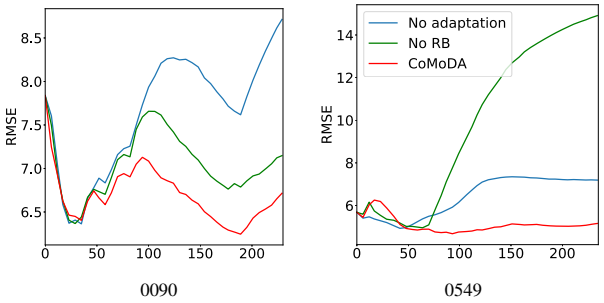

Figure 7. Running mean RMSE on the NuScenes test videos. Horizontal axis represents video frames, vertical – RMSE in meters.

6. Conclusion

In this paper, we propose a novel approach for continuous monocular depth adaptation. By complementing the monocular supervisory signal with past experience, we are able to substantially improve the quality of depth predictions in both intra- and cross-dataset setups.

Our scale-aware method achieves state-of-the-art results on Eigen split [8] of KITTI [11], while being an order of magnitude faster compared to the competing methods performing test-time fine-tuning. We investigated the influence of various design choices on the model performance and showed that the replay buffer is the efficient and robust technique to mitigate overfitting, affecting the 'baseline' variant of monocular adaptation.

For future work, we would like to incorporate a stronger motion handling strategy (e.g., use optical flow or model motion), which would allow us to obtain additional supervision from objects such as static cars. Further improvements might be achieved by strengthening temporal consistency (e.g., by using RNNs or depth network with multiple image inputs), or by integrating more sophisticated strategies for the replay buffer generation and sampling.

Acknowledgment. The authors thankfully acknowledge the support by Toyota via the TRACE project.

References

[1] Filippo Aleotti, Fabio Tosi, Matteo Poggi, and Stefano Mattoccia. Generative adversarial networks for unsupervised monocular depth prediction. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 0–0, 2018.

[2] Amir Atapour-Abarghouei and Toby P Breckon. Real-time monocular depth estimation using synthetic data with domain adaptation via image style transfer. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2800–2810, 2018.

[3] Holger Caesar, Varun Bankiti, Alex H. Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuscenes: A multimodal dataset for autonomous driving. *arXiv preprint arXiv:1903.11027*, 2019.

[4] Vincent Casser, Soeren Pirk, Reza Mahjourian, and Anelia Angelova. Depth prediction without the sensors: Leveraging structure for unsupervised learning from monocular videos. Code repository https://github.com/tensorflow/models/blob/archive/research/struct2depth/optimize.py.

[5] Vincent Casser, Soeren Pirk, Reza Mahjourian, and Anelia Angelova. Depth prediction without the sensors: Leveraging structure for unsupervised learning from monocular videos. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 8001–8008, 2019.

[6] Weifeng Chen, Zhao Fu, Dawei Yang, and Jia Deng. Single-image depth perception in the wild. In *Advances in neural information processing systems*, pages 730–738, 2016.

[7] Yuhua Chen, Cordelia Schmid, and Cristian Sminchisescu. Self-supervised learning with geometric constraints in monocular video: Connecting flow, depth, and camera. In *The IEEE International Conference on Computer Vision (ICCV)*, October 2019.

[8] David Eigen, Christian Puhrsch, and Rob Fergus. Depth map prediction from a single image using a multi-scale deep network. In *Advances in neural information processing systems*, pages 2366–2374, 2014.

[9] Huan Fu, Mingming Gong, Chaohui Wang, Kayhan Batmanghelich, and Dacheng Tao. Deep ordinal regression network for monocular depth estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2002–2011, 2018.

[10] 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.

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

[12] Clément Godard, Oisin Mac Aodha, and Gabriel J Brostow. Unsupervised monocular depth estimation with left-right consistency. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 270–279, 2017.

[13] Clement Godard, Mac Aodha Oisin, Michael Firman, and Gabriel J. Brostow. Digging into self-supervised monocular depth estimation. In *The IEEE International Conference on Computer Vision (ICCV)*, October 2019.

[14] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[15] Ariel Gordon, Hanhan Li, Rico Jonschkowski, and Anelia Angelova. Depth from videos in the wild: Unsupervised monocular depth learning from unknown cameras. In *The IEEE International Conference on Computer Vision (ICCV)*, October 2019.

[16] Vitor Guizilini, Rares Ambrus, Sudeep Pillai, Allan Raventos, and Adrien Gaidon. 3d packing for self-supervised monocular depth estimation. In *The IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[17] Xiaoyang Guo, Hongsheng Li, Shuai Yi, Jimmy Ren, and Xiaogang Wang. Learning monocular depth by distilling cross-domain stereo networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 484–500, 2018.

[18] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[19] Glenn Jocher. Yolov5. Code repository https://github.com/ultralytics/yolov5.

[20] Yevhen Kuznietsov, Jorg Stuckler, and Bastian Leibe. Semi-supervised deep learning for monocular depth map prediction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6647–6655, 2017.

[21] Iro Laina, Christian Rupprecht, Vasileios Belagiannis, Federico Tombari, and Nassir Navab. Deeper depth prediction with fully convolutional residual networks. In *2016 Fourth international conference on 3D vision (3DV)*, pages 239–248. IEEE, 2016.

[22] Ruihao Li, Sen Wang, Zhiqiang Long, and Dongbing Gu. Undeepvo: Monocular visual odometry through unsupervised deep learning. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 7286–7291. IEEE, 2018.

[23] Shunkai Li, Xin Wang, Yingdian Cao, Fei Xue, Zike Yan, and Hongbin Zha. Self-supervised deep visual odometry with online adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6339–6348, 2020.

[24] Long-Ji Lin. Self-improving reactive agents based on reinforcement learning, planning and teaching. *Machine learning*, 8(3-4):293–321, 1992.

[25] Keng-Chi Liu, Yi-Ting Shen, Jan P. Klopp, and Liang-Gee Chen. What synthesis is missing: Depth adaptation integrated with weak supervision for indoor scene parsing. In *The IEEE International Conference on Computer Vision (ICCV)*, October 2019.

[26] Chenxu Luo, Zhenheng Yang, Peng Wang, Yang Wang, Wei Xu, Ram Nevatia, and Alan Yuille. Every pixel counts++:

Joint learning of geometry and motion with 3d holistic understanding. *arXiv preprint arXiv:1810.06125*, 2018.

[27] Reza Mahjourian, Martin Wicke, and Anelia Angelova. Unsupervised learning of depth and ego-motion from monocular video using 3d geometric constraints. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5667–5675, 2018.

[28] Jogendra Nath Kundu, Phani Krishna Uppala, Anuj Pahuja, and R Venkatesh Babu. Adadepth: Unsupervised content congruent adaptation for depth estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2656–2665, 2018.

[29] Andrea Pilzer, Stephane Lathuiliere, Nicu Sebe, and Elisa Ricci. Refine and distill: Exploiting cycle-inconsistency and knowledge distillation for unsupervised monocular depth estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9768–9777, 2019.

[30] Andrea Pilzer, Dan Xu, Mihai Puscas, Elisa Ricci, and Nicu Sebe. Unsupervised adversarial depth estimation using cycled generative networks. In *2018 International Conference on 3D Vision (3DV)*, pages 587–595. IEEE, 2018.

[31] Anurag Ranjan, Varun Jampani, Lukas Balles, Kihwan Kim, Deqing Sun, Jonas Wulff, and Michael J Black. Competitive collaboration: Joint unsupervised learning of depth, camera motion, optical flow and motion segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 12240–12249, 2019.

[32] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015.

[33] Ashutosh Saxena, Min Sun, and Andrew Y Ng. Make3d: Learning 3d scene structure from a single still image. *IEEE transactions on pattern analysis and machine intelligence*, 31(5):824–840, 2008.

[34] Alessio Tonioni, Oscar Rahnama, Thomas Joy, Luigi Di Stefano, Thalaiyasingam Ajanthan, and Philip H.S. Torr. Learning to adapt for stereo. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[35] 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)*, June 2019.

[36] Patil Vaishakh, Wouter Van Gansbeke, Dengxin Dai, and Luc Van Gool. Don't forget the past: Recurrent depth estimation from monocular video. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020.

[37] Sudheendra Vijayanarasimhan, Susanna Ricco, Cordelia Schmid, Rahul Sukthankar, and Katerina Fragkiadaki. Sfm-net: Learning of structure and motion from video. *arXiv preprint arXiv:1704.07804*, 2017.

[38] Chaoyang Wang, José Miguel Buenaposada, Rui Zhu, and Simon Lucey. Learning depth from monocular videos using direct methods. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2022–2030, 2018.

[39] Zhou Wang, Alan C Bovik, Hamid R Sheikh, Eero P Simoncelli, et al. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004.

[40] Alex Wong and Stefano Soatto. Bilateral cyclic constraint and adaptive regularization for unsupervised monocular depth prediction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5644–5653, 2019.

[41] Junyuan Xie, Ross Girshick, and Ali Farhadi. Deep3d: Fully automatic 2d-to-3d video conversion with deep convolutional neural networks. In *European Conference on Computer Vision*, pages 842–857. Springer, 2016.

[42] Zhenheng Yang, Peng Wang, Yang Wang, Wei Xu, and Ram Nevatia. Lego: Learning edge with geometry all at once by watching videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 225–234, 2018.

[43] Zhenheng Yang, Peng Wang, Wei Xu, Liang Zhao, and Ramakant Nevatia. Unsupervised learning of geometry from videos with edge-aware depth-normal consistency. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[44] Wei Yin, Yifan Liu, Chunhua Shen, and Youliang Yan. Enforcing geometric constraints of virtual normal for depth prediction. In *The IEEE International Conference on Computer Vision (ICCV)*, October 2019.

[45] Zhichao Yin and Jianping Shi. Geonet: Unsupervised learning of dense depth, optical flow and camera pose. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1983–1992, 2018.

[46] Huangying Zhan, Ravi Garg, Chamara Saroj Weerasekera, Kejie Li, Harsh Agarwal, and Ian Reid. Unsupervised learning of monocular depth estimation and visual odometry with deep feature reconstruction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 340–349, 2018.

[47] Haokui Zhang, Chunhua Shen, Ying Li, Yuanzhouhan Cao, Yu Liu, and Youliang Yan. Exploiting temporal consistency for real-time video depth estimation. In *The IEEE International Conference on Computer Vision (ICCV)*, October 2019.

[48] Zhenyu Zhang, Zhen Cui, Chunyan Xu, Yan Yan, Nicu Sebe, and Jian Yang. Pattern-affinitive propagation across depth, surface normal and semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4106–4115, 2019.

[49] Zhenyu Zhang, Stéphane Lathuilière, Andrea Pilzer, Nicu Sebe, Elisa Ricci, and Jian Yang. Online adaptation through meta-learning for stereo depth estimation. *arXiv preprint arXiv:1904.08462*, 2019.

[50] Zhenyu Zhang, Stephane Lathuiliere, Elisa Ricci, Nicu Sebe, Yan Yan, and Jian Yang. Online depth learning against forgetting in monocular videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4494–4503, 2020.

[51] Shanshan Zhao, Huan Fu, Mingming Gong, and Dacheng Tao. Geometry-aware symmetric domain adaptation for

monocular depth estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9788–9798, 2019.

[52] Yiran Zhong, Hongdong Li, and Yuchao Dai. Open-world stereo video matching with deep rnn. In *The European Conference on Computer Vision (ECCV)*, September 2018.

[53] Tinghui Zhou, Matthew Brown, Noah Snavely, and David G Lowe. Unsupervised learning of depth and ego-motion from video. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1851–1858, 2017.

[54] Yuliang Zou, Zelun Luo, and Jia-Bin Huang. Df-net: Unsupervised joint learning of depth and flow using cross-task consistency. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 36–53, 2018.