MonoRec: Semi-Supervised Dense Reconstruction in Dynamic Environments from a Single Moving Camera Supplementary Material

Felix Wimbauer[1,⋆] Nan Yang[1,2,⋆] Lukas von Stumberg[1] Niclas Zeller[1,2] Daniel Cremers[1,2]
[1] Technical University of Munich, [2] Artisense
{wimbauer, yangn, stumberg, zellern, cremers}@in.tum.de

1. Introduction

In this supplementary material, we provide additional details in extension to our main paper. This mainly includes more implementation details (Sec. 2) and additional experimental results (Sec. 3).

2. Implementation Details

The exact details of our network architecture can be observed in Figure 2.

As described in section 3.4 of the main paper, we use several different error thresholds to generate the auxiliary training masks. Since for this task it is more important for the error metric to be semantically consistent instead of very detailed, we use perceptual error instead of absolute differences or SSIM. To this end, we employ the first 9 layers of a pretrained VGG-16 network from the PyTorch model zoo. The per-pixel error between two images is defined as the mean squared error between the respective feature vectors for the respective pixels. The thresholds are as follows: (1) $pe^t_{t^S}(\mathbf{x}, D_t(\mathbf{x})) > 12$ (2) $\overline{pe^t_{t'}}(\mathbf{x}, D^S_t(\mathbf{x})) > 8$ (3) $\max\{\frac{D_t(\mathbf{x})}{D^S_t(\mathbf{x})}, \frac{D^S_t(\mathbf{x})}{D_t(\mathbf{x})}\} > 1.5$. If at least two out of these conditions are fulfilled a pixel is considered to be moving. To ensure temporal consistency of the moving object masks, we match every detected segmentation mask with masks from the previous and the following frame. The matched segmentation masks have to be from the same object class and have a minimum IoU of 0.25. A segmentation mask is accepted as a moving object, if it itself and the matched segmentation masks contain on average more than 40% moving pixels.

3. Additional Experiments

We provide additional experimental results. This comprises more extensive ablation studies (Sec. 3.1) where we specifically evaluate the performance of the MaskModule. Furthermore, the effect of different model configurations is evaluated.

Model	Prec	Rec	IoU
Baseline (only ResNet)	0.017	0.658	0.016
Baseline (only cost volume)	0.230	0.642	0.204
Baseline	0.260	0.678	0.232
Mask Refinement	**0.374**	**0.748**	**0.300**

Table 1: **Ablation Study - MaskModule**: Results for the masks predicted by our MaskModule compared to the auxiliary masks on the proposed KITTI Odometry [2] test set using different versions of our model. **Note**: The auxiliary masks can not be compared to ground truth as they themselves contain many mistakes (both missed detections and miss-classifications). Our **Baseline** model was only trained with the auxiliary masks. **Mask Refinement** describes our model after the mask refinement training. It improves the performance across all metrics.

We also provide some of the failure cases in which our method does not achieve optimal performance (Sec. 3.2).

In addition to the qualitative generalization capabilities of our method presented in the main paper, we also provide quantitative results obtained from the Oxford RobotCar dataset [6] (Sec. 3.3) and the TUM RGB-D dataset [9] (Sec. 3.4).

In Sec. 3.5, we show the quantitative evaluation against two other monocular dense reconstruction methods in dynamic scenes [7, 8].

3.1. Ablation Studies

In the ablation studies presented in the main paper, we focused on the overall performance on MVS depth prediction and the contribution of the different components. Here, we pay attention to the MaskModule and its performance with respect to masking out dynamic objects. Furthermore, we evaluation different model configurations.

	Model	Abs Rel	Sq Rel	RMSE	RMSE$_{log}$	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$
(a)	4 Frames	0.045	0.267	2.130	0.082	0.975	0.991	0.995
	6 Frames	0.046	0.271	2.163	0.087	0.972	0.989	0.995
	320x640	0.052	0.309	2.230	0.084	0.970	0.990	0.995
	KITTI poses	0.077	0.077	3.283	0.943	0.943	0.982	0.992
	MonoRec	0.050	0.288	2.269	0.082	0.972	0.991	0.996
(b)	M, D* Baseline	0.059	0.494	2.764	0.096	0.966	0.987	0.994
	MS, D* Baseline	0.054	0.346	2.444	0.088	0.970	0.989	0.995

Table 2: **Ablation Study - Model Configuration**: Depth prediction results using different model configurations. **(a)** All models use the same weights, that were trained with 2 frames, DVSO [10] poses and 256 × 512. **(b)** Mono vs. Mono + Stereo training of depth module.

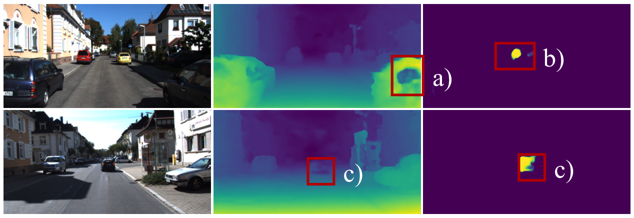

Figure 1: **Failure Cases**: a) Non-lambertian surfaces, especially ones that are very close, can lead to mis-predictions due to a wrong cost volume prior. b) The MaskModule sometimes detects the focal point, if far away, as a moving object. The effect is minimal, because these pixels are not used for reconstruction. c) If the predicted mask does not cover the moving object entirely, the network might produce artifacts due wrong cost volume priors.

3.1.1 MaskModule

For MaskModule it is more important to filter out all moving objects reliably than having a very high precision, since DepthModule is able to fill out small missing patches in the cost volume. Therefore, in the trade-off between recall and precision we put higher emphasis on recall. As baseline we consider MaskModule only trained based on the the auxiliary masks. This baseline is compared against the mask prediction after refinement training. The baseline already achieves fairly high recall, however, the precision is not very strong (see Table 1). Through the refinement training, which puts the mask prediction into direct context with the cost volume input, the performance is improved across all metrics, especially the precision.

3.1.2 Model Configuration

The standard configuration of our model receives a keyframe and two additional mono frames (the one before and after the keyframe) at a resolution 256 × 512 as well as poses generated by DVSO [10] as input. However, our implementation is very flexible. It can take any number of frames at any resolution that is a multiple of 16. Furthermore, the pose source can easily be replaced, e.g. by another visual odometry (VO) algorithm or other sensors (e.g. INS). The results in Table 2 shows that by feeding more frames into the model, one can, in fact, improve the performance. However, this effect saturates after a certain number of frames. Interestingly, our model works significantly worse with the ground truth poses provided by KITTI Odometry [2]. We believe that this is because DVSO [10] computes poses solely based on monocular photometric error, similarly to the way our cost volume is built. Furthermore, since the ground truth poses in KITTI are obtained based on an INS system, they might be locally less accurate than the VO poses and not perfectly synchronized with the images. Finally, our model does not seem to significantly benefit from a larger image input size.

Method	Abs Rel	RMSE	$\delta < 1.25$
Monodepth2 [3]	0.220	7.328	0.616
PackNet [4]	0.233	7.512	0.606
PackNet [4](supervi.)	0.229	7.983	0.620
DORN [1]	0.215	7.966	0.651
DeepMVS [5]	**0.142**	7.379	<u>0.780</u>
DeepMVS [5] (pretr.)	0.153	**6.656**	0.770
DeepTAM [11] (only FB)	0.154	7.355	0.776
DeepTAM [11] (1x Ref.)	0.152	7.211	0.749
MonoRec	<u>0.143</u>	<u>7.180</u>	**0.806**

Table 3: **Oxford RobotCar**: Quantitative performance of different models on the Oxford RobotCar dataset. Best / Second best results are marked **bold** / <u>underlined</u>.

3.2. Failure Cases

In Figure 1 we visualize typical failure cases of our method. Some of the show failure cases, like the ones caused by non-lambertian surfaces are typical for MVS methods. Other failures are a result of miss-detections of the MaskModule. However, at least partially, those miss-detections can be compensated by our DepthModule.

3.3. Oxford RobotCar Dataset

In Table 3 we show the quantitative results of Oxford RobotCar generated with the official long sample sequence. To get the ground truth, we aggregated multiple LiDAR scans within a range of 0.25 s before and after the frame

Method	Abs Rel	RMSE	$\delta < 1.25$
MonoDepth2 [3]	0.353	1.240	0.458
DeepTAM [11] (1xRef)	*0.210*	*0.792*	*0.701*
MonoRec	**0.189**	**0.756**	**0.725**

Table 4: **TUM RGB-D**: Quantitative performance of different methods on the TUM RGB-D dataset. Specifically, we evaluate on the `freiburg3_long_office_household` sequence. Best / Second best results are marked **bold** / underlined. All methods are trained on KITTI and MonoRec shows stronger generalization capability.

Method	Abs Rel	RMSE	$\delta < 1.25$
DenseMono [7]	0.148	2.408	not provided
MonoRec	**0.079**	**1.469**	0.949
VideoPopup [8]	0.154	2.631	0.752
MonoRec	**0.054**	**2.304**	**0.970**

Table 5: **Quantitative Results - Further Methods**: Comparisons of depth evaluation to further methods. Best results are marked **bold**. In the comparison to DenseMono [7], sequences 11-21 of the KITTI odometry dataset are used. For the comparison to VideoPopup, sequence 05 of the KITTI odometry dataset is used.

timestamp and transformed it using the odometry poses. Note that, due to the short sequence and the low quality of LiDAR data, one has to consider the provided numbers with caution. Nevertheless, considering the numbers our method performs arguably overall the best among all evaluated methods.

3.4. TUM RGB-D

To further demonstrate MonoRec's generalization capabilities, we also performed quantitative analysis on the indoor TUM RGB-D [9] dataset using the models trained on KITTI. Table 4 shows that MonoRec delivers better results compared to other methods.

3.5. Further Quantitative Evaluations

In Table 5 we show quantitative comparisons to DenseMono [7] and VideoPopup [8]. These methods, like MonoRec, aim to deliver accurate depths for dynamic scenes and make use of consecutive frames as input additional to the keyframe. Both methods employ classical optimization methods instead of neural networks. The evaluation results suggest that MonoRec performs better than DenseMono and VideoPopup.

References

[1] Huan Fu, Mingming Gong, Chaohui Wang, Kayhan Batmanghelich, and Dacheng Tao. Deep ordinal regression network for monocular depth estimation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2002–2011, 2018. 2

[2] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the KITTI vision benchmark suite. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3354–3361. IEEE, 2012. 1, 2

[3] Clément Godard, Oisin Mac Aodha, Michael Firman, and Gabriel J. Brostow. Digging into self-supervised monocular depth estimation. In *International Conference on Computer Vision (ICCV)*, pages 3828–3838, 2019. 2, 3

[4] Vitor Guizilini, Rares Ambrus, Sudeep Pillai, Allan Raventos, and Adrien Gaidon. 3D packing for self-supervised monocular depth estimation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2485–2494, 2020. 2

[5] Po-Han Huang, Kevin Matzen, Johannes Kopf, Narendra Ahuja, and Jia-Bin Huang. DeepMVS: Learning multi-view stereopsis. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2821–2830, 2018. 2

[6] Will Maddern, Geoff Pascoe, Chris Linegar, and Paul Newman. 1 Year, 1000km: The Oxford RobotCar Dataset. *International Journal of Robotics Research (IJRR)*, 36(1):3–15, 2017. 1

[7] Rene Ranftl, Vibhav Vineet, Qifeng Chen, and Vladlen Koltun. Dense monocular depth estimation in complex dynamic scenes. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4058–4066, 2016. 1, 3

[8] Chris Russell, Rui Yu, and Lourdes Agapito. Video pop-up: Monocular 3d reconstruction of dynamic scenes. In *European Conference on Computer Vision (ECCV)*, pages 583–598. Springer, 2014. 1, 3

[9] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers. A benchmark for the evaluation of rgb-d slam systems. In *Proc. of the International Conference on Intelligent Robot Systems (IROS)*, Oct. 2012. 1, 3

[10] Nan Yang, Rui Wang, Jörg Stückler, and Daniel Cremers. Deep virtual stereo odometry: Leveraging deep depth prediction for monocular direct sparse odometry. In *European Conference on Computer Vision (ECCV)*, pages 817–833, 2018. 2

[11] Huizhong Zhou, Benjamin Ummenhofer, and Thomas Brox. DeepTAM: Deep tracking and mapping. In *European Conference on Computer Vision (ECCV)*, pages 822–838, 2018. 2, 3

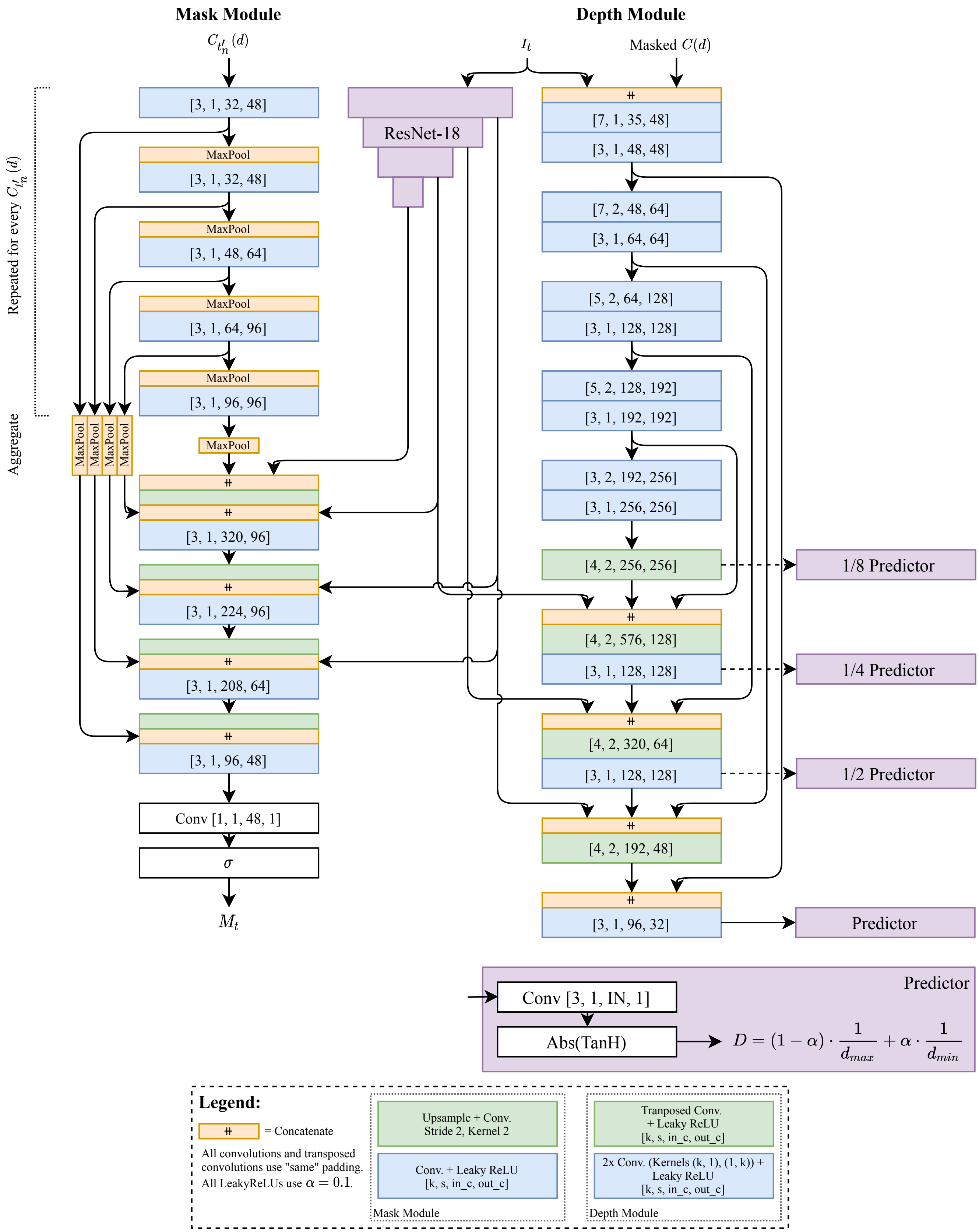

Figure 2: **Detailed Architecture of MonoRec**