

CamLessMonoDepth: Monocular Depth Estimation with Unknown Camera Parameters

Supplementary Material

Sai Shyam Chanduri¹
chanduriss@gmail.com

Zeeshan Khan Suri¹
zshn25@gmail.com

Igor Vozniak²
Igor.Vozniak@dfki.de

Christian Müller²
Christian.Mueller@dfki.de

¹ University of Applied Sciences (HTW)
Saarbrücken, Germany

² Department of Agents and Simulated
Reality (ASR)
German Research Center for Artificial
Intelligence (DFKI GmbH)
Saarbrücken, Germany

In this document, we provide additional evaluations and ablations experiments in support of our work in Section A and expand on the network architectures and implementation details in Section B.

A Additional Experiments

A.1 Depth

KITTI Improved Ground Truth. For Eigen split evaluation proposed by Eigen et al. [1] on KITTI benchmark, the authors have made use of reprojected LIDAR points for depth evaluation. However, such approach does not account for other important aspects like occlusions, moving objects and ego-vehicle motion [2]. Hence, we make use of improved ground truth from [3], in which occlusions are handled by considering stereo pairs and high quality depth maps which are produced by accumulating five consecutive frames. This results in 652 images, which accounts to 93% of images of the Eigen split. Depth is capped to 80 meters similar to Eigen split evaluation and the error metrics also remain the same. The evaluation results using such improved ground truth data are reported in Table 1, where both variants (V1 and V2) of our proposed method significantly outperform other benchmark methods in all evaluation metrics. Interestingly, when compared in between, version V2, namely the model with uncertainty estimation, exemplified better performance than model V1 without uncertainty estimation, in all the evaluation metrics. This infers the effectiveness of the uncertainty component on depth estimation.

Depth encoder variation. Using high-end architectures for the depth estimation is advantageous, nevertheless, would come at the cost of higher training and inference times. Here, we report the impact of such architectures on depth performance by varying depth encoders. Therefore, we compare depth performance of ResNet-18(B), Resnet-50(B) architectures from [4] and Packnet(B) architecture from, [5] with ResNet-18(P), ResNext-50 (P)

Method	Lower is better				Higher is better		
	Abs Rel	Sq Rel	RMSE	RMSE log	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$
SFMlearner [10]	0.176	1.532	6.129	0.244	0.758	0.921	0.971
Vid2Depth [10]	0.134	0.983	5.501	0.203	0.827	0.944	0.981
GeoNet [10]	0.132	0.994	5.240	0.193	0.833	0.953	0.985
DDVO [10]	0.126	0.866	4.932	0.185	0.851	0.958	0.986
Ranjan [8]	0.123	0.881	4.834	0.181	0.860	0.959	0.985
EPC++ [8]	0.120	0.789	4.755	0.177	0.856	0.961	0.987
MonoDepth2 [10]	0.090	0.545	3.942	0.137	0.914	0.983	0.995
Ours without uncert.(V1)	0.082	0.451	3.666	0.126	0.925	0.986	0.996
Ours with uncert.(V2)	0.081	0.427	3.532	0.124	0.928	0.986	0.996

Table 1: The evaluation of results on KITTI test data with improved ground truth. The corresponding metrics are taken from their corresponding papers and [10], where the best results along each metric are highlighted in bold.

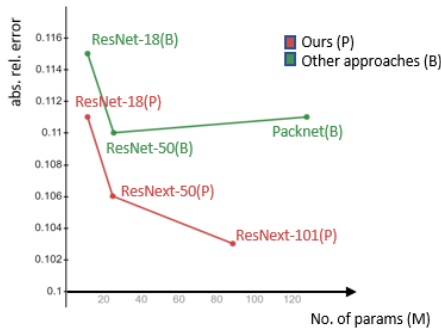


Figure 1: Comparison of baseline (B) methods with proposed (P) methods, with absolute relative error against number of trainable parameters in millions(M). Here ResNet-18, ResNet-50 versions of [10] and Packnet from [8] are compared with ResNet-18, ResNext-50 and ResNext-101 depth encoders of our approach.

and ResNext-101(P) architectures of our proposed method, where (B) indicates baseline approaches and (P) stands for our proposed method. All the ResNet and ResNext architectures (from both (B) and (P)) employ pretrained weights on ImageNet [9] while Packnet [8] is not using any pretraining. From findings reported in Table 2, the proposed method involving ResNet-18(P) depth encoder with just 11.69 million(M) trainable parameters, has achieved comparable performance in regard to the high-end baseline (B) methods including Packnet with 128 M trainable parameters. Meanwhile, the proposed approach with ResNext-50 (P) depth encoder outperforms all the baseline (B) approaches significantly, while ResNext-101(P) depth encoder improve results even further. Comparison of depth performance with various architectures and their corresponding trainable parameters is depicted in Figure 1.

Comparison with MonoDepth2 [10]. In this experiment, we compare our model results with our baseline, MonoDepth2 [10] to show the significance of our approach. Table 3 shows the results of baseline and the proposed method at both, lower (640x192) and higher input resolutions (1024x320) images. Here, the baseline [10] has not shown greater improvement with change in input resolution, nevertheless, our approach manifested significant improvement due to our use of sub-pixel convolutions for upsampling. We have also compared our

Encoder	param.(M)	Lower is better				Higher is better		
		Abs Rel	Sq Rel	RMSE	RMSE log	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$
ResNet-18(B)	11.69	0.115	0.903	4.863	0.193	0.877	0.959	0.981
ResNet-50(B)	25.56	0.110	0.831	4.642	0.187	0.883	0.962	0.982
Packnet(B)	128	0.111	0.785	4.601	0.189	0.878	0.960	0.982
ResNet-18(P)	11.69	0.111	0.832	4.709	0.188	0.881	0.961	0.982
ResNext-50(P)	25.03	0.106	0.750	4.482	0.182	0.891	0.964	0.983
ResNext-101(P)	88.79	0.103	0.774	4.514	0.180	0.897	0.965	0.983

Table 2: Table showing evaluation of results when the model is trained with different types of encoders. (B) indicates baseline and (P) indicates proposed method. Evaluation of results is done on KITTI test data. Except for the ResNet-18 approaches, for all other, ResNet-50 pose encoder is employed.

approach with baseline at edge cases involving thin objects, object boundaries and colour saturated regions, and this qualitative analysis is demonstrated in [Figure 2](#). Our model, especially for higher resolution input, exemplified better depth results at thin objects and the objects nearer to the camera. Also, sharper depth results can be observed at object boundaries, credit to our efficient sub-pixel convolutions which enabled better super-resolution for up-sampling in between decoder layers. In colour saturated regions (such as in [Figure 2\(d\)](#)), the low resolution model of MonoDepth2 shows notable artifacts which are somewhat compensated with the higher resolution model. Both our models, however, do not exhibit any such artifacts and result in smoother and robust depth maps compared to the baseline approach. To support the above listed statements, we additionally rendered 3D (X,Y,Z) perspectives for the generated depth images ([Figure 3](#)), where Z coordinate stands for the normalized value for each pixel of the generated depth image. As seen from [Figure 3](#) our approach shows less uncertainty (less dense points) in boundary regions of the objects. In other words, the proposed approach predicts better the class assignment for each depth point, e.g. vehicle vs background. Moreover, it demonstrates a better accuracy in terms of the reconstructed shape of scene objects.

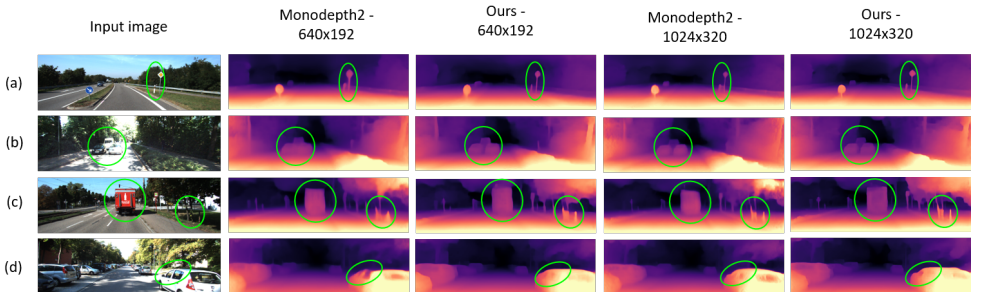


Figure 2: The qualitative comparison of our approach in contrary to MonoDepth2 [1] with (a) near thin objects, (b) with close objects, (c) at object boundaries, (d) colour saturated regions.

Method	Resolution	Lower is better				Higher is better		
		Abs Rel	Sq Rel	RMSE	RMSE log	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$
MonoDepth2 [10]	640x192	0.115	0.903	4.863	0.193	0.877	0.959	0.981
Ours (V2)	640x192	0.106	0.750	4.482	0.182	0.891	0.964	0.983
MonoDepth2 [10]	1024x320	0.115	0.882	4.701	0.190	0.879	0.961	0.982
Ours (V2)	1024x320	0.102	0.723	4.374	0.178	0.898	0.966	0.983

Table 3: The evaluation results on KITTI benchmark with change in input resolution. For 1024x320 resolution, only ResNet-18 pose encoder is used due to computational limitations. For 640x192 resolution, ResNext-50 pose encoder is used.

A.2 Odometry Evaluation

KITTI Odom Dataset. KITTI Odometry dataset [10] consists of 22 driving sequences, of which only 11 sequences have ground truth trajectories and the remaining 11 sequences are unlabelled. We make use of only labelled sequences for both, training and evaluation. Similar to the prior approaches [10, 15], we first train our models on sequences - 00 to 08 using the KITTI Odometry split which has 36630 images and then evaluate separately on sequence 09 (1591 images) and sequence 10 (1201 images) data.

Performance Metrics. For odometry evaluation, we use Absolute Trajectory Error (ATE) as performance metric as proposed in [15]. This metric computes the Root Mean Square Error (RMSE) between the ground truth and the estimated trajectory. ATE can be computed with any of these: translation, rotation or velocity. All these parameters return the same single error metric, making it easier to compare [15]. We opt translation to compute ATE similar to previous approaches [10] for odometry evaluation.

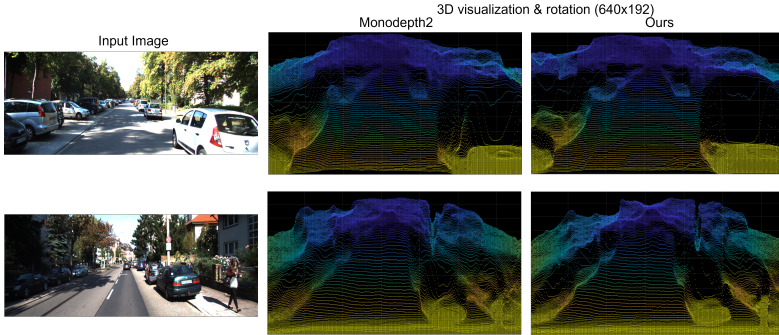


Figure 3: The 3D qualitative comparison of our approach in contrary to MonoDepth2 [10] in 3D (X, Y, Z as in point cloud datasets) format, where the normalized color value for every single pixel from the predicted images is used as depth (Z) value during 3D visualization. In addition, the rotation of the point cloud data has been applied during renderings in order to emphasize the improvements of the proposed approach over the baseline, namely less uncertainty on neighbour object boundary regions as well as more accurate reconstruction of shapes.

Evaluation. For odometry evaluation, we follow [10] and predict on five-frame test sequence, which is used in [15]. We calculate the ATE of our predictions for each of the four

pairs in a five-frame test sequence, and then combine them to report the metric’s mean and standard deviation. Results of odometry evaluation are reported in Table 4. For this experiment, we have employed default settings of the proposed method. Our proposed approach exemplifies similar performance with our baseline [10]. There is a significant improvement in depth results when compared with our baseline, however, not in pose, which is predominantly impacted by the use of common encoder for both camera pose and intrinsics estimation. In addition, when compared with other approaches [10, 8, 13], similar to [10], the improvement is not significant because of (i) not using custom architectures for odometry evaluation as this particular pose network is designed particularly for better depth estimation and (ii) using only two frames for pose estimation.

Method	#frames	Sequence 09	Sequence 10
DDVO [13]	3	0.045±0.108	0.033±0.074
SFMlearner [13]	5	0.021±0.017	0.020±0.015
Mahjourian [10]	3	0.013±0.010	0.012±0.011
GeoNet [13]	3	0.012±0.007	0.012±0.009
EPC++ [10]	3	0.013±0.007	0.012±0.008
Ranjan [8]	3	0.012±0.007	0.012±0.008
MonoDepth2 [10]	2	0.017±0.008	0.015±0.010
Ours	2	0.018±0.009	0.015±0.009

Table 4: Pose evaluation performed with KITTI Odom data. Sequences 09 and 10 of this dataset are used for evaluation purposes, where #frames indicate the number of input frames for pose network. The mean ATE along with its standard deviation is reported for each approach.

A.3 Camera Intrinsics Evaluation

We opt for KITTI odometry dataset [10] for camera intrinsics evaluation similar to our odometry evaluation (refer subsection A.2). Here, the model is trained with sequences - 00 to 08, and we use sequences 09 and 10 for evaluation. The mean and standard deviation of focal length parameters (f_x and f_y) and principal offsets (x_0 and y_0) are reported with each of the evaluation sequences, and these parameters are compared with the ground truth data. Distortion parameters are not modelled, hence are not reported. For this setup, we use ResNext-50 depth encoder and ResNet-18 pose encoder, with input resolution to the framework as 1024x320. The quantitative evaluation findings for this experiment are shown in Table 5. It can be observed that the results are impacted significantly by using a light-weight camera network and also due to the usage of a common encoder for camera pose and intrinsics. Nevertheless, the depth estimation results were improved, with this intrinsics estimations, when compared to given camera intrinsics as input (shown in KITTI ablation experiments as part of the main paper).

B Additional Implementation Details

Depth Network. We embed efficient sub-pixel convolutions into the depth decoder for better upsampling in contrary to the nearest-neighbour interpolation followed in previous approaches [10, 13, 13, 13]. These efficient sub-pixel convolutions involve three convolution operations followed by a final pixel shuffling operation, which operates at lower resolution

Method	Sequence 09	Sequence 10	Ground Truth
Horizontal focal length (f_x)	0.6902 ± 0.0138	0.6868 ± 0.0208	0.5767
Vertical focal length (f_y)	1.9208 ± 0.0805	1.8947 ± 0.0807	1.9111
Horizontal centre (x_0)	0.5032 ± 0.0031	0.5017 ± 0.0029	0.4909
Vertical centre (y_0)	0.4681 ± 0.0025	0.4678 ± 0.0030	0.4949

Table 5: Our camera network estimates of the intrinsics parameters, normalized by the input image resolution, are compared against the ground truth data. The mean of these normalized intrinsics estimates with standard deviations are reported for each parameter computed on all the images from KITTI Odom 09 and 10 sequences.

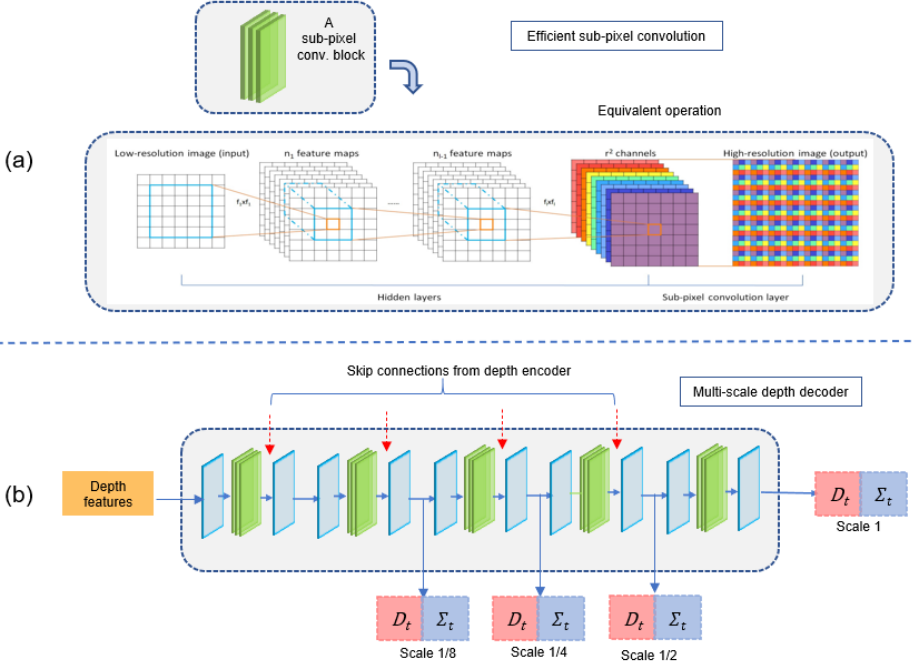


Figure 4: (a) shows the sub-pixel convolution operation adapted from [10] depicting three 3×3 2D convolutions followed by a pixel-shuffling operation which rearranges pixels from higher number of channels into higher resolution of width and height. (b) shows the multi-scale decoder architecture with 3×3 convolutions (blue) and sub-pixel convolutions (green) predicting outputs at four scales.

in extracting feature information necessary to perform super-resolution. Just before the final pixel shuffle operation, the output is arranged to have the channels multiplied by a factor of, r^2 where r is the upsample factor. This output is finally shuffled across the pixels along channels to obtain image super-resolution. This operation is shown in Figure 4(a).

The depth network, in overall, takes in a single image as input and outputs disparity or inverse-depth along with a pixel-wise uncertainty map. These outputs are obtained at four scales, i.e. at 1/8, 1/4, 1/2 and 1, scaling with input image resolution as shown in Figure 4(b). Such multi-scale decoder is employed to prevent the gradient locality of the bilinear sampler

layer	k	s	p	chns	res	input	activation
dconv5	3	1	1	256	32	econv5	ELU
s1conv5	5	1	2	64	32	dconv5	ReLU
s2conv5	3	1	1	32	32	s1conv5	ReLU
upconv5	3	1	1	256*4	32	s2conv5	ReLU
iconv5	3	1	1	256	16	[ps]upconv5, econv4	ELU
dconv4	3	1	1	128	16	iconv5	ELU
s1conv4	5	1	2	64	16	dconv4	ReLU
s2conv4	3	1	1	32	16	s1conv4	ReLU
upconv4	3	1	1	128*4	16	s2conv4	ReLU
iconv4	3	1	1	128	8	[ps]upconv4, econv3	ELU
disp_uncert4	3	1	1	2	1	iconv4	Sigmoid
dconv3	3	1	1	64	8	iconv4	ELU
s1conv3	5	1	2	64	8	dconv3	ReLU
s2conv3	3	1	1	32	8	s1conv5	ReLU
upconv3	3	1	1	64*4	8	s2conv5	ReLU
iconv3	3	1	1	64	4	[ps]upconv3, econv2	ELU
disp_uncert3	3	1	1	2	1	iconv3	Sigmoid
dconv2	3	1	1	32	4	iconv3	ELU
s1conv2	5	1	2	64	4	dconv2	ReLU
s2conv2	3	1	1	32	4	s1conv5	ReLU
upconv2	3	1	1	32*4	4	s2conv5	ReLU
iconv2	3	1	1	32	2	[ps]upconv2, econv1	ELU
disp_uncert2	3	1	1	2	1	iconv2	Sigmoid
dconv1	3	1	1	16	2	iconv2	ELU
s1conv1	5	1	2	64	2	dconv1	ReLU
s2conv1	3	1	1	32	2	s1conv1	ReLU
upconv1	3	1	1	16*4	2	s2conv1	ReLU
iconv1	3	1	1	16	1	[ps]upconv1	ELU
disp_uncert1	3	1	1	2	1	iconv1	Sigmoid

Table 6: The network details of depth decoder used in our approach. Here, k indicates kernel size, p indicates padding, $chns$ indicate number of channels for that layer, res stands for the downsampling factor, where 1 indicates full resolution, $input$ stands for the input to that layer, [ps] indicates pixel shuffle operation with an upsample factor of 2, $econv$ represents inputs from various levels of the encoder, and lastly $activation$ indicates the activation function used for that corresponding layer.

and to prevent the loss objective getting stuck at local minimum [2]. The disparity map obtained is later converted to depth using $D = 1/(a * \sigma + b)$ where D represents depth and σ represents the disparity map. Here, a and b are constants which are chosen such as to constrain the depth values between 0.1 and 100. The detailed network architecture is shown in Table 6.

Pose Decoder							
layer	k	s	p	chns	res	input	activation
pconv0	1	1	1	256	32	econv5	ReLU
pconv1	3	1	1	256	32	pconv0	ReLU
pconv2	3	1	1	256	32	pconv1	ReLU
pconv3	1	1	1	6	32	pconv2	-
Camera Network							
inconv0	1	1	1	256	32	econv5	ReLU
inconv1	3	1	1	2	32	inconv0	Softplus
inconv2	3	1	1	2	32	inconv0	-
inconcat	-	-	-	-	-	inconv1, inconv2	-

Table 7: The network details of the pose decoder and the camera network are shown. Here, k indicates the kernel size, p indicates the padding parameter, $chns$ stands for the number of channels for that particular convolutional layer, res corresponds to the downsampling factor, where 1 indicates full resolution, $input$ stands for the input to that layer, $econv5$ is the output from pose encoder, and lastly $activation$ shows the used activation function

Pose and Camera Networks. The pose encoder is modified to take in a pair of images as input, which are concatenated channel-wise. Also, the weights in the expanded filter are divided by 2, which makes the output of the convolution similar to ResNet default output with a single image input [2]. The camera network follows a light-weight architecture to estimate camera intrinsic parameters. Horizontal focal length (f_x) and Horizontal centre (x_0) parameters are initialized to $W/2$, and Vertical focal length (f_y) and Vertical centre (y_0) parameters are initialized to $H/2$ for better convergence following [2], where W and H represents width and height of the input image respectively. Network details of both pose and camera networks are shown in Table 7.

References

- [1] David Eigen, Christian Puhrsch, and Rob Fergus. Depth map prediction from a single image using a multi-scale deep network. *arXiv preprint arXiv:1406.2283*, 2014.
- [2] Clément Godard, Oisín Mac Aodha, Michael Firman, and Gabriel J Brostow. Digging into self-supervised monocular depth estimation. In *Proceedings of the IEEE international conference on computer vision*, pages 3828–3838, 2019.
- [3] Ruben Gomez-Ojeda and Javier Gonzalez-Jimenez. Robust Stereo Visual Odometry through a Probabilistic Combination of Points and Line Segments. In *Robotics and Automation (ICRA), 2016 IEEE International Conference on*. IEEE, 2016.
- [4] Ariel Gordon, Hanhan Li, Rico Jonschkowski, and Anelia Angelova. Depth from videos in the wild: Unsupervised monocular depth learning from unknown cameras. In

- Proceedings of the IEEE International Conference on Computer Vision*, pages 8977–8986, 2019.
- [5] Vitor Guizilini, Rares Ambrus, Sudeep Pillai, Allan Raventos, and Adrien Gaidon. 3d packing for self-supervised monocular depth estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2485–2494, 2020.
 - [6] 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.
 - [7] 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.
 - [8] 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.
 - [9] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115 (3):211–252, 2015.
 - [10] Wenzhe Shi, Jose Caballero, Ferenc Huszár, Johannes Totz, Andrew P Aitken, Rob Bishop, Daniel Rueckert, and Zehan Wang. Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1874–1883, 2016.
 - [11] Jonas Uhrig, Nick Schneider, Lukas Schneider, Uwe Franke, Thomas Brox, and Andreas Geiger. Sparsity invariant cnns. In *2017 international conference on 3D Vision (3DV)*, pages 11–20. IEEE, 2017.
 - [12] 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.
 - [13] 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.
 - [14] Zichao Zhang and Davide Scaramuzza. A tutorial on quantitative trajectory evaluation for visual (-inertial) odometry. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 7244–7251. IEEE, 2018.
 - [15] 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.