Multi-Frame Self-Supervised Depth with Transformers – Supplementary Material –

Vitor Guizilini Rareș Ambruș Dian Chen Sergey Zakharov Adrien Gaidon
Toyota Research Institute (TRI), Los Altos, CA
{first.lastname}@tri.global

1. Network Details

Below we describe each network used in our proposed DepthFormer architecture, and Table 1 shows detailed diagrams for each of them. Note that our contributions do not require any network architecture in particular, and can be extended to incorporate recent developments for potential further improvements in performance [7, 16, 17]. Open-source training and inference code, as well as pre-trained models, will be made available upon publication.

1.1. Cross-Attention Network

Similar to [15], we use an hourglass-shaped architecture as the encoder, modified with residual connections and spatial pyramid pooling modules [1]. The decoder consists of transposed convolutions, dense-blocks [10], and a final convolution layer. The final feature map has the same spatial resolution as the input image, encoding both local and global contexts. This feature map is then downsampled to the cost volume resolution using bilinear interpolation.

1.2. Single-Frame Depth Network

We use a ResNet18 backbone [6] as the single-frame encoder, followed by a decoder that outputs multi-scale depth maps at four different resolutions: one-eighth, one-fourth, one-half, and the original input dimension. Following [21], we concatenate the $H/4 \times W/4 \times 128$ encoded features with the $H/4 \times W/4 \times D$ multi-frame cost volume. A bottleneck convolutional layer, with kernel size 3, is then used to combine these two sources of features (single-frame and multi-frame) into a $H/4 \times W/4 \times 128$ feature map for further encoding and decoding (Figure 4, main paper).

1.3. Context Adjustment Network

The input to our context adjustment network is a $H/4 \times W/4 \times 4$ tensor created by concatenating the normalized high-response depth map $\tilde{D}_H$ and the target image I_t. Depth map normalization is done as such:

$$\tilde{D}_H = \Big(\hat{D}_H - mean(\hat{D}_H)\Big) \Big/ std(\hat{D}_H) \tag{1}$$

This normalized high-response depth map is refined through a series of residual blocks that expand the channel dimensions, before a ReLU activation restores it to the original shape. The high-response depth map is concatenated with the output of each residual block, and added to the final output using a long skip connection. This final output is then un-normalized using the original statistics, generating a context-adjusted predicted depth map $\hat{D}_C$:

$$\hat{D}_C = \Big(\tilde{D}_H + \theta_C(I_t, \tilde{D}_H)\Big)\, std(\hat{D}_H) + mean(\hat{D}_H) \tag{2}$$

1.4. Pose Network

Our pose network uses a ResNet18 backbone, modified to accommodate two input images by duplicating the convolutional weights of the first layer [6]. The bottleneck feature maps are further processed using a series of convolutional layers, with the last one outputting a $H/32 \times H/32 \times 6$ feature map. This feature map is then averaged over the spatial dimensions, generating a 6-dimensional vector containing the relative translation and rotation between frames, in Euler angles. Following [21], we invert the order of input images when predicting backwards motion.

2. Comparison to Supervised Methods

Our DepthFormer architecture was designed for self-supervised learning, in which training is conducted without explicit supervision from ground-truth depth maps. As mentioned in the main paper (Section 2.1), this is a very challenging setting, due to limitations of the photometric objective in the presence of dynamic objects, static frames, changes in luminosity, and so forth. Even so, our contributions in multi-frame feature matching lead to a depth estimation performance that surpasses even current state-of-the-art single-frame supervised depth estimation methods. These results are summarized in Table 2. More specifically, we achieve comparable performance to BTS [3] when training and evaluating at half resolution (640×192), and surpass it in almost all metrics when training and evaluating at the same full resolution (1216×352). We believe the introduction of other self-supervised depth network architectures

	Layer Description	K	S	Output Dim.
	ResidualBlock (#0)			
#1	Conv2d (#0a) → BN → ReLU	K	1	
#2	Conv2d → BN → ReLU	K	S	
#3	Downsample(#0) + #2 → ReLU	-	-	
	UpsampleBlock (#0, #s)			
#1	Conv2d → BN → ReLU → Upsample	3	1	
#2	Conv2d (#1 ⊕ #s) → BN → ReLU	3	1	
	InverseDepth (#0)			
#1	Conv2d → Sigmoid	K	S	
#2	(*max* - *min*) ⊙ #1 + *min*	-	-	
#0a	Input RGB image	-	-	3×H×W
#0b	Input cost volume	-	-	128×H/4×W/4
	Encoder			
#1	Conv2d → BN → ReLU	7	1	64×H×W
#2	Max. Pooling	3	2	64×H/2×W/2
#3	ResidualBlock (#2) x2	3	1-2	64×H/4×W/4
#4	#3 ⊕ #0b → Conv2d → BN → ReLU	3	1	64×H/4×W/4
#5	ResidualBlock (#4) x2	3	1-2	128×H/8×W/8
#6	ResidualBlock (#5) x2	3	1-2	256×H/16×W/16
#7	ResidualBlock (#6) x2	3	1-2	512×H/32×W/32
	Decoder			
#8	UpsampleBlock (#7,#6)	3	1	256×H/16×W/16
#9	UpsampleBlock (#8,#5)	3	1	128×H/8×W/8
#10	InverseDepth (#8)	3	1	1×H/8×W/8
#11	UpsampleBlock (#9,#3)	3	1	64×H/4×W/4
#12	InverseDepth (#11)	3	1	1×H/4×W/4
#13	UpsampleBlock (#11,#2)	3	1	32×H/2×W/2
#14	InverseDepth (#13)	3	1	1×H/2×W/2
#15	UpsampleBlock (#13,–)	3	1	32×H×W
#16	InverseDepth (#15)	3	1	1×H×W

(a) **Single-frame depth network.** The target image I_t is used as input, as well as the cross-attention cost volume $\mathcal{A}_{t\to c}$. Bold numbers indicate the 4 multi-scale output inverse depth maps, at increasing resolutions. Each sigmoid output is converted to depth using *min* and *max* ranges.

	Layer Description	K	S	Output Dim.
	ResidualBlock (#0)			
#1	Conv2d → BN → ReLU	K	1	
#2	Conv2d → BN → ReLU	K	S	
#3	Downsample(#0) + #2 → ReLU	-	-	
	SpatialPyramidBlock (#0, N)			
#1	Avg. Pool	N	N	
#2	Conv2d → BN → ReLU	K	S	
#0	Input RGB image	-	-	6×H×W
#1	Conv2d → BN → ReLU	3	2	16×H/2×W/2
#2	Conv2d → BN → ReLU	3	1	16×H/2×W/2
#3	Conv2d → BN → ReLU	3	1	32×H/2×W/2
#4	ResidualBlock (#3)	3	2	64×H/4×W/4
#5	ResidualBlock (#4)	3	2	128×H/8×W/8
#6	SpatialPyramidBlock (#5,16)	1	1	32×H/128×W/128
#7	SpatialPyramidBlock (#5,8)	1	1	32×H/64×W/64
#8	SpatialPyramidBlock (#5,4)	1	1	32×H/32×W/32
#9	SpatialPyramidBlock (#5,2)	1	1	32×H/16×W/16
#10	Downsample(#6 ⊕ #7 ⊕ #8 ⊕ #9)	-	-	128×H/16×W/16
#11	DenseBlock (#0 ⊕ #10)	1	1	128×H×W
#12	DenseBlock (#4 ⊕ #11)	1	1	128×H×W
#13	DenseBlock (#5 ⊕ #12)	1	1	128×H×W
#14	DenseBlock (#10 ⊕ #13)	1	1	128×H×W

(b) **Attention network.** It processes the target I_t and context images I_c independently, and the output is used to generate the cross-attention cost volume $\mathcal{A}_{t\to c}$, as described in Section 3.2.2, main paper.

	Layer Description	K	S	Output Dim.
	ResidualBlock (#0)			
#1	Conv2d → BN → ReLU	K	1	
#2	Conv2d → BN → ReLU	K	S	
#3	Downsample(#0) + #2 → ReLU	-	-	
#0	Input 2 RGB images	-	-	6×H×W
	Encoder			
#1	Conv2d → BN → ReLU	7	1	64×H×W
#2	Max. Pooling	3	2	64×H/2×W/2
#3	ResidualBlock (#2) x2	3	1-2	64×H/4×W/4
#4	ResidualBlock (#3) x2	3	1-2	128×H/8×W/8
#5	ResidualBlock (#4) x2	3	1-2	256×H/16×W/16
#6	ResidualBlock (#5) x2	3	1-2	512×H/32×W/32
	Decoder			
#7	Conv2d → ReLU	1	1	256×H/32×W/32
#8	Conv2d → ReLU	3	1	256 ×H/32×W/32
#9	Conv2d → ReLU	3	1	256×H/32×W/32
#10	Conv2d → ReLU	1	1	6×H/32×W/32
#11	Global Avg. Pooling	-	-	6×1×1

(c) **Pose network.** The target I_t and context I_c images are concatenated and used as input. The 6-dimensional output contains predicted relative translation (x, y, z) and rotation $(roll, pitch, yaw)$ in Euler angles.

	Layer Description	K	S	Output Dim.
	ResidualBlock (#0: N × H × W)			
#1	Conv2d → BN → ReLU	K	1	3N ×H×W
#2	Conv2d → BN → ReLU	K	S	N ×H×W
#3	Downsample(#0) + #2 → ReLU	-	-	N ×H×W
#0a	Input RGB image	-	-	3×H×W
#0b	Input norm. depth map	-	-	1×H×W
#1	#0a ⊕ #0b	3	1	4×H×W
#2	Conv2d → GN → ReLU	3	1	16×H×W
#3	ResidualBlock(⊕ #0b) x8	3	1	16×H×W
#4	Conv2d + #0b	3	1	1×H×W

(d) **Context adjustment network.** Input high-response depth maps $\hat{D}_H$ are normalized using Equation 1, and output depth maps $\tilde{D}_C$ are un-normalized using Equation 2.

Table 1. **Network architectures used in our experiments.** *BN* stands for Batch Normalization [12], *Upsample* and *Downsample* respectively increases and decreases spatial dimensions using bilinear interpolation to match the output resolution, *ReLU* are Rectified Linear Units, *Sigmoid* is the sigmoid activation function, and *DenseBlock* are densely connected convolutional layers from [10]. The symbol ⊕ indicates feature concatenation, and ⊙ indicates element-wise multiplication.

Method	Superv.	Multi-Fr.	*Lower is better*				*Higher is better*		
			AbsRel	SqRel	RMSE	RMSE_{log}	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$
Kuznietsov *et al.* [13]	D		0.113	0.741	4.621	0.189	0.862	0.960	0.986
Gan *et al.* [4]	D		0.098	0.666	3.933	0.173	0.890	0.964	0.985
Guizilini *et al.* [8]	D		0.072	0.340	3.265	0.116	0.934	—	—
DORN [3]	D		0.072	0.307	2.727	0.120	0.932	0.984	0.994
Yin *et al.* [22]	D		0.072	—	3.258	0.117	0.938	0.990	**0.998**
PackNet-SFM [7]	M		0.078	0.420	3.485	0.121	0.931	0.986	0.996
ManyDepth [21]	M	✓	0.064	0.320	3.187	0.104	0.946	0.990	0.995
BTS [14]	D		0.059	**0.245**	2.756	0.096	0.956	**0.993**	**0.998**
DepthFormer (MR)	M	✓	**0.055**	0.271	2.917	0.095	0.955	0.991	**0.998**
DepthFormer (HR)	M	✓	**0.055**	0.265	**2.723**	**0.092**	**0.959**	0.992	**0.998**

Table 2. **Depth results** on the KITTI *Eigen* test split [2], for distances up to 80m with the *Garg* crop [5], evaluated on the improved depth maps from [18]. *Superv.* indicates the source of supervision (M for monocular self-supervision and D for depth supervision); and *Multi-Fr.* the use of multiple frames at test time. Monocular results are median-scaled at test time, to account for scale ambiguity. We report DepthFormer results in both half-resolution (MR, 640×192), and full resolution (HR, 1216×352), using the same training and architecture parameters (Section 4.2, main paper).

more suitable for high resolution processing [7] should lead to further improvements, however a more thorough exploration is left to future work.

3. Qualitative Examples

Some examples of predicted depth maps, including common failure cases due to lack of camera motion and dynamic objects, are shown in Figures 1 and 2 for the KITTI and DDAD datasets respectively. High-response depth maps (Section 3.3.1, main paper) are masked out using our proposed low-confidence threshold. These masked out regions usually include far-away objects towards the vanishing point, including the sky, and interestingly also occluded areas and dynamic objects. Context-adjusted depth maps (Section 3.3.2, main paper) are able to reason over these low-confidence areas by conditioning with information from the target image. However, they still fail in situations where multi-frame matching is inaccurate or ill-posed (e.g., lack of camera motion or dynamic objects). By introducing single-frame features for joint decoding (Section 3.3.3, main paper), we are able to also reason over these situations and achieve our reported state-of-the-art results. Quantitative evaluation of these intermediate depth maps is provided in Table 2 of the main paper.

4. Reconstructed Pointclouds

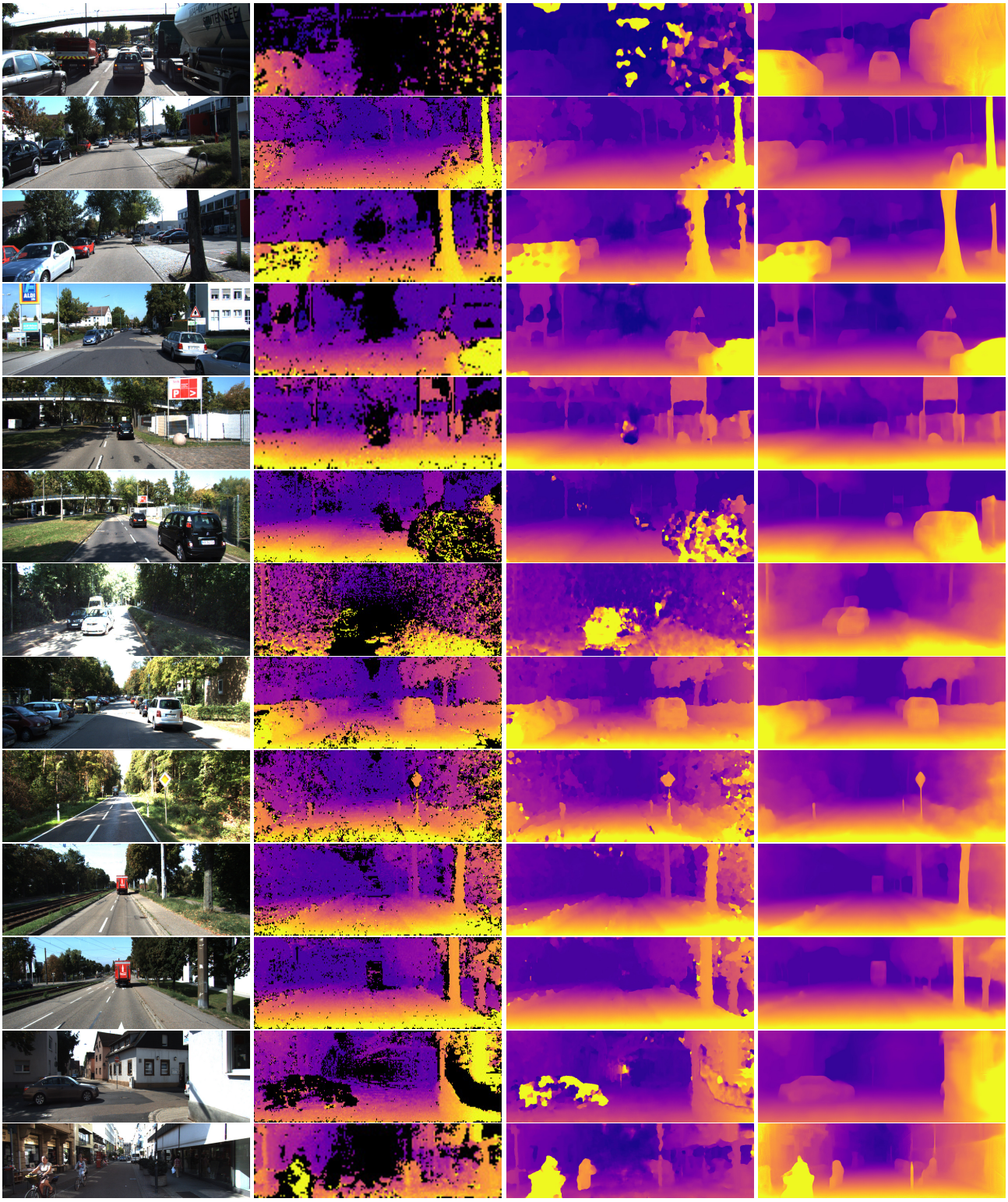
We also show examples of reconstructed KITTI and DDAD pointclouds in Figures 3 and 4. These pointclouds are obtained by unprojecting pixel colors to 3D space using known camera intrinsics, predicted depth maps, and predicted relative motion between frames. We reiterate here that no ground-truth is used at training or inference time, only videos. Even so, our architecture is able to reconstruct the observed environment, including low-texture regions, object boundaries, and dynamic objects to a high degree of accuracy, as shown in our quantitative evaluation (Table 2). For examples of pointcloud reconstruction over entire sequences, please refer to the supplementary video.

5. Negative Impact

Because our proposed method operates on a monocular self-supervised setting, it can process arbitrarily large amounts of unlabeled visual data without human intervention. However, more does not necessarily means better, and some amount of data curation is still desirable, to avoid the introduction of biases in trained models due to data imbalance. Another potential issue is privacy, and proper procedures should be taken when processing large quantities of data without supervision, to preserve individual anonymity.

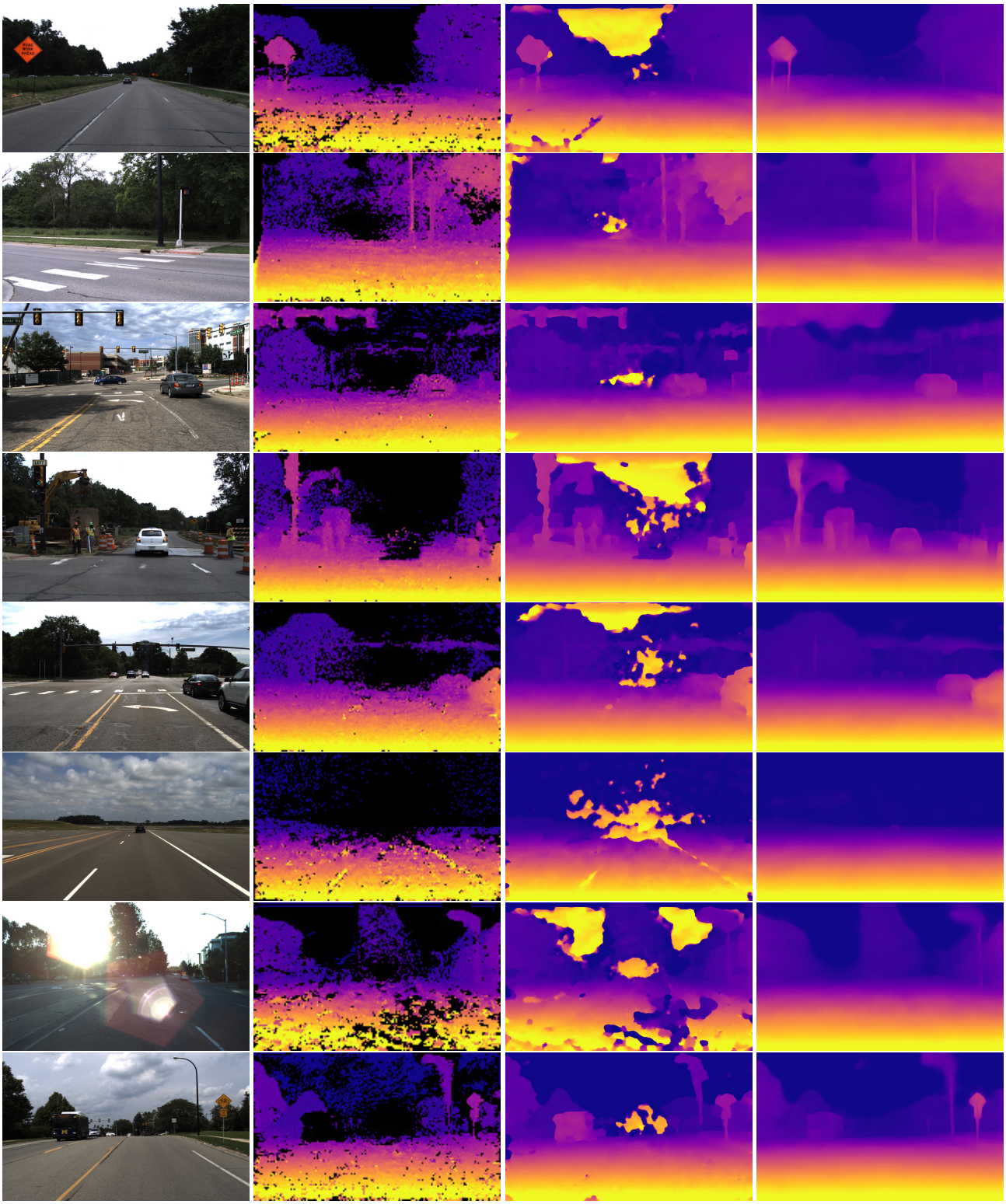
6. Limitations

Our proposed method increases robustness to some of the common challenges found in self-supervised monocular depth estimation, such as dynamic objects and static frames, by improving feature matching across frames. However, it does not explicitly address these issues, which would require 3D motion modeling in the form of scene flow [11] or tracking [23]. Another common limitation of self-supervised monocular depth estimation is scale ambiguity, since models trained purely on image information cannot produce metrically-accurate predictions. Scale-aware results are necessary for downstream tasks that ingest our reconstructed pointclouds, such as 3D object detection [20]. Some works have addressed this limitation in the self-supervised setting by introducing weak velocity supervision [7] or additional geometric information such as camera height [19] or multi-camera extrinsics [9]. Our proposed method does not address this issue, however it can directly benefit from these works to produce scale-aware estimates.



(a) Target image (b) High-response depth (c) Context-adjusted depth (d) Decoded depth

Figure 1. **Qualitative depth estimation results** of our proposed DepthFormer architecture, on the KITTI dataset.



(a) Target image (b) High-response depth (c) Context-adjusted depth (d) Decoded depth

Figure 2. **Qualitative depth estimation results** of our proposed DepthFormer architecture, on the DDAD dataset.

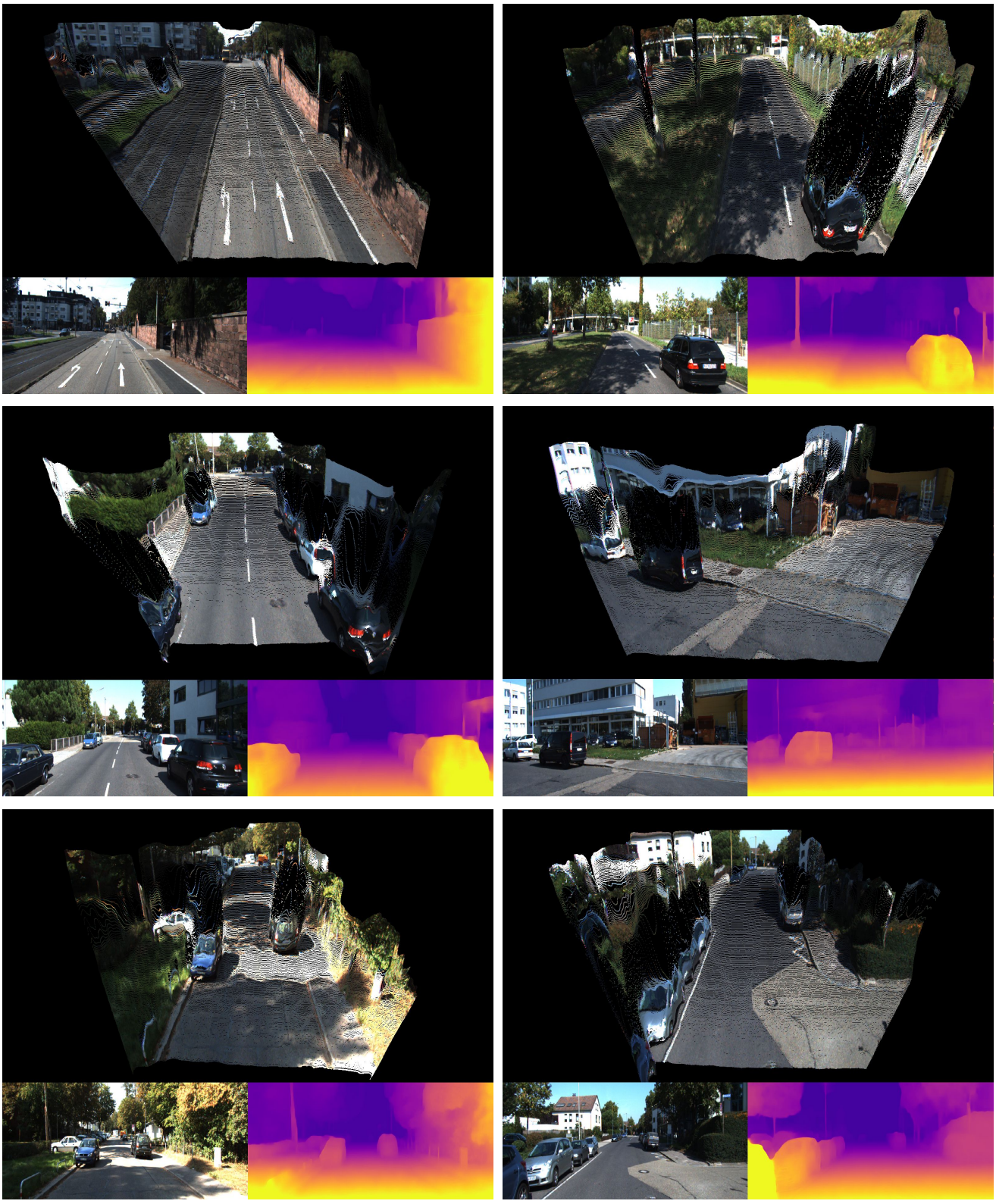


Figure 3. **Pointcloud reconstructions** obtained using our DepthFormer architecture, on the KITTI dataset.

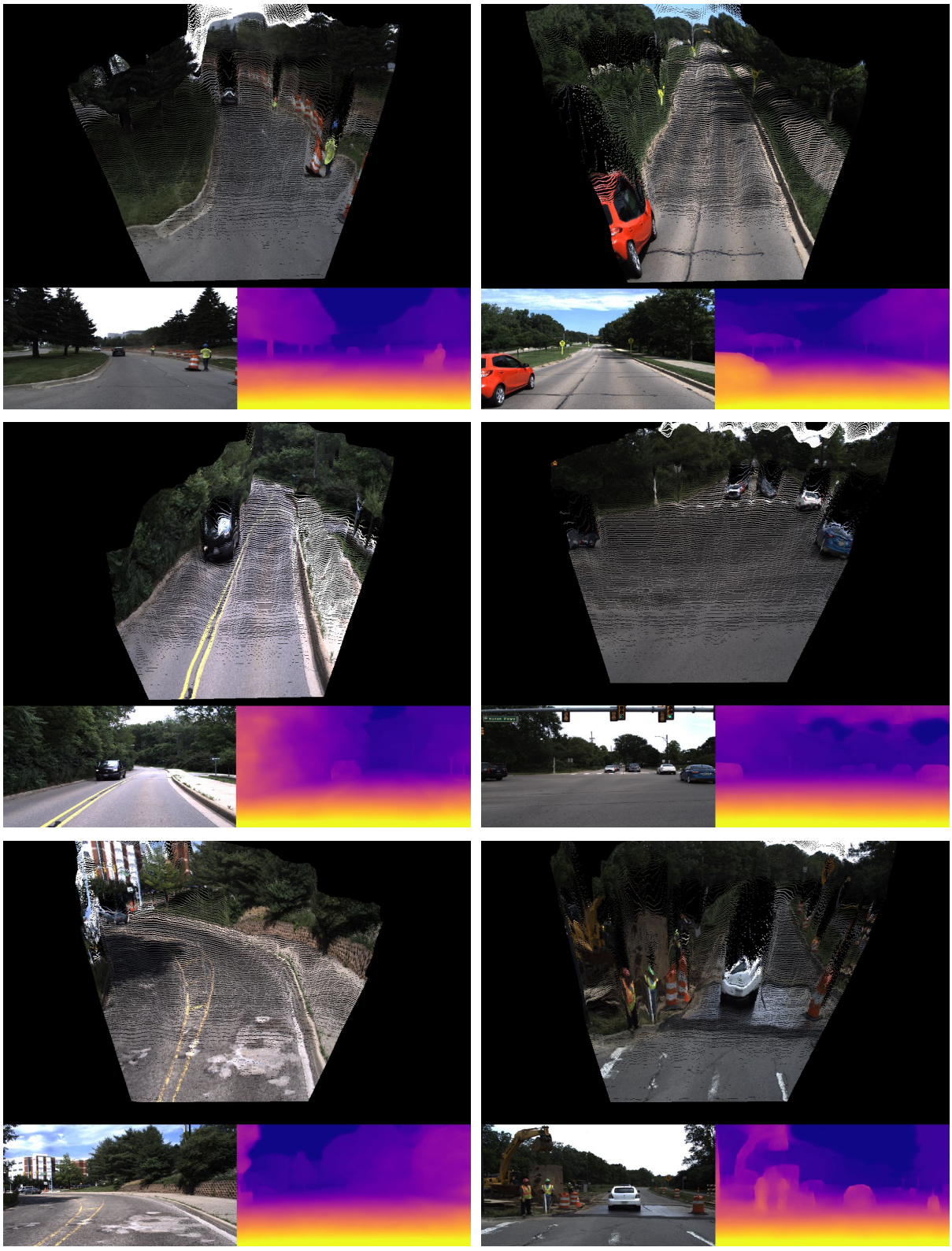


Figure 4. **Pointcloud reconstructions** obtained using our DepthFormer architecture, on the DDAD dataset.

References

[1] Jia-Ren Chang and Yong-Sheng Chen. Pyramid stereo matching network. In *CVPR*, 2018. 1

[2] David Eigen, Christian Puhrsch, and Rob Fergus. Depth map prediction using a multi-scale deep network. *arXiv:1406.2283*, 2014. 3

[3] Huan Fu, Mingming Gong, Chaohui Wang, Kayhan Batmanghelich, and Dacheng Tao. Deep ordinal regression network for monocular depth estimation. In *CVPR*, 2018. 1, 3

[4] Yukang Gan, Xiangyu Xu, Wenxiu Sun, and Liang Lin. Monocular depth estimation with affinity, vertical pooling, and label enhancement. In *ECCV*, 2018. 3

[5] Ravi Garg, Vijay Kumar Bg, Gustavo Carneiro, and Ian Reid. Unsupervised cnn for single view depth estimation: Geometry to the rescue. In *ECCV*, 2016. 3

[6] Clément Godard, Oisin Mac Aodha, Michael Firman, and Gabriel J. Brostow. Digging into self-supervised monocular depth prediction. In *ICCV*, 2019. 1

[7] Vitor Guizilini, Rares Ambrus, Sudeep Pillai, Allan Raventos, and Adrien Gaidon. 3d packing for self-supervised monocular depth estimation. In *CVPR*, 2020. 1, 3

[8] Vitor Guizilini, Jie Li, Rares Ambrus, Sudeep Pillai, and Adrien Gaidon. Robust semi-supervised monocular depth estimation with reprojected distances. In *CoRL*, 2019. 3

[9] Vitor Guizilini, Igor Vasiljevic, Rares Ambrus, Greg Shakhnarovich, and Adrien Gaidon. Full surround monodepth from multiple cameras. *arXiv:2104.00152*, 2021. 3

[10] Gao Huang, Zhuang Liu, Laurens van der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *CVPR*, 2017. 1, 2

[11] Junhwa Hur and Stefan Roth. Self-supervised monocular scene flow estimation. In *CVPR*, 2020. 3

[12] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv:1502.03167*, 2015. 2

[13] Yevhen Kuznietsov, Jorg Stuckler, and Bastian Leibe. Semi-supervised deep learning for monocular depth map prediction. In *CVPR*, 2017. 3

[14] Jin Han Lee, Myung-Kyu Han, Dong Wook Ko, and Il Hong Suh. From big to small: Multi-scale local planar guidance for monocular depth estimation. *arXiv:1907.10326*, 2019. 3

[15] Zhaoshuo Li, Xingtong Liu, Nathan Drenkow, Andy Ding, Francis X Creighton, Russell H Taylor, and Mathias Unberath. Revisiting stereo depth estimation from a sequence-to-sequence perspective with transformers. *arXiv:2011.02910*, 2020. 1

[16] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *ICCV*, 2021. 1

[17] Chang Shu, Kun Yu, Zhixiang Duan, and Kuiyuan Yang. Feature-metric loss for self-supervised learning of depth and egomotion. In *ECCV*, 2020. 1

[18] J. Uhrig, N. Schneider, L. Schneider, U. Franke, T. Brox, and A. Geiger. Sparsity invariant cnns. In *3DV*, 2017. 3

[19] Brandon Wagstaff and Jonathan Kelly. Self-supervised scale recovery for monocular depth and egomotion estimation. *arXiv:2009.03787*, 2021. 3

[20] Tai Wang, Xinge Zhu, Jiangmiao Pang, and Dahua Lin. FCOS3D: Fully convolutional one-stage monocular 3d object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) Workshops*, 2021. 3

[21] Jamie Watson, Oisin Mac Aodha, Victor Prisacariu, Gabriel Brostow, and Michael Firman. The Temporal Opportunist: Self-Supervised Multi-Frame Monocular Depth. In *CVPR*, 2021. 1, 3

[22] Wei Yin, Yifan Liu, Chunhua Shen, and Youliang Yan. Enforcing geometric constraints of virtual normal for depth prediction. In *ICCV*, 2019. 3

[23] Xingyi Zhou, Vladlen Koltun, and Philipp Krähenbühl. Tracking objects as points. *ECCV*, 2020. 3