TESPEC: Temporally-Enhanced Self-Supervised Pretraining for Event Cameras

Supplementary Material

A. Additional Experimental Results

In this section, we provide further analysis to complement the experimental results presented in the main paper. This includes an expanded ablation study to show the effectiveness of our improved intensity estimation video, qualitative results that highlight the strengths of our approach through visual examples, and a discussion of the trade-off involved in choosing the temporal bin in Eq. (7).

A.1. Ablation Study

We present additional evidence to show the effectiveness of our improved intensity video reconstruction method, Eq. (7), compared to the naive approach from prior work, Eq. (6). The performance of fine-tuned models on the downstream DDD17 [1] semantic segmentation task is summarized in Tab. 1. Using our intensity estimation as the reconstruction target achieves improvements of 1.352% in mIoU and 2.691% in mAcc over the naive estimation.

A.2. Qualitative Results

We provide additional qualitative results to demonstrate the performance of our model in both self-supervised pre-training and downstream perception tasks.

Pre-training. The visualized sample in Fig. 1 shows TESPEC reconstruction of the scene. This representation includes static objects that were invisible in recent event frames. This example highlights that the recurrent design of TESPEC helps it to extract a rich representation of various objects in the scene, beyond those that are actively moving.

Object Detection. Qualitative samples of object detection on the Gen1 [2] dataset are presented in Fig. 2. The model successfully detects cars in the scene, even when they are barely visible in recent events.

Semantic Segmentation. Fig. 3 showcases examples from the DSEC [7] and DDD17 [1] datasets. These results show that our pre-trained model adapt effectively to downstream datasets with varying resolutions. Notably, the model predicts accurate segmentation maps, even with sparse inputs.

Monocular Depth Estimation. Qualitative results in Fig. 4 show that the model accurately distinguishes between different objects and predicts precise depth maps.

A.3. Choosing the temporal bin in Eq. (7)

While using small temporal bins increases temporal resolution, it also increases the computational cost, as more iterations are required to update the reconstruction target. In addition, noises, such as hot pixels, might dominate some of the pixels. On the other hand, large bins result in severe information loss due to a large Δ. We find that a 5ms bin size balances target precision and noise robustness.

B. Justification of MAE for sparse event data

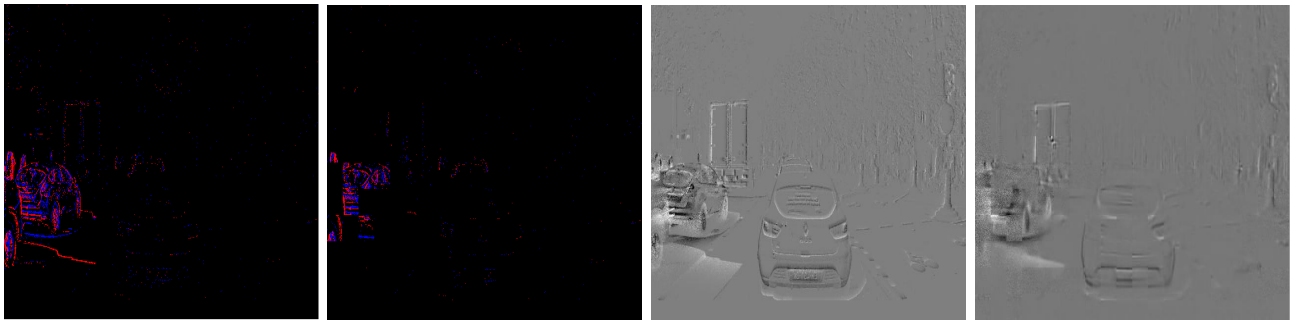
Although the event input is sparse, TESPEC reconstructs a *dense pseudo gray-scale* video as target. Therefore, reconstructing masked patches of our target provides a strong training signal. Moreover, masking a large portion of the event input along the time requires the model to *estimate spatiotemporal information*. This helps the model's scene understanding and leads to higher performance compared to training with no masking (see Tab 4. in the main text). To assess the impact of the pre-training objective, we replace the MAE [22] loss in TESPEC with two alternative loss functions: (i) Contrastive Predictive Coding (CPC) [11, 15], which learns representations by predicting future latent features using contrastive loss, and (ii) the contrastive loss introduced in ECDDP [25]. Their downstream performance on the semantic segmentation task on the DSEC dataset is reported in Tab. 2. While both contrastive losses lead to improved performance over the baseline recurrent architecture without pre-training, MAE yields the highest accuracy. This demonstrates the effectiveness of MAE as a pre-training objective in TESPEC.

C. Implementation Details

We describe the implementation settings used for TESPEC and downstream tasks, including object detection, semantic segmentation, and monocular depth estimation. Tab. 4 summarizes the settings and hyper-parameters.

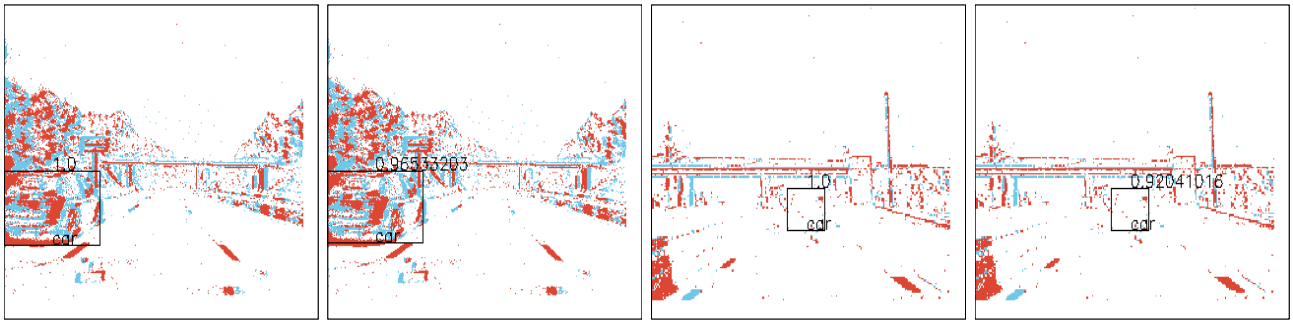
Data Pre-processing. We adopt a unified event representation across TESPEC and all downstream tasks. An event stream is split into non-overlapping event segments of length T. Each segment is then converted into a 2D histogram with 10 bins for positive events and 10 bins for negative ones. Input values are clipped between 0 and 10 to prevent the influence of hot pixels. The input resolution is required to be divisible by 32 due to the characteristics of the used architecture. For datasets with resolutions that do not meet this requirement, zero-padding is applied to align the input dimensions. Data augmentation is performed using flipping and scaling transformations. The probabilities and parameters of transformations are detailed in Tab. 5.

Dataloading. During the training phase of recurrent models, we process multiple stages within a single training iteration. The total number of stages is determined by the sequence length. To enable the model to handle long sequences in TESPEC and object detection, we adopt the dat-



(a) Input events. (b) Masked events. (c) Estimated intensity video. (d) Prediction.

Figure 1. **Qualitative pre-training results on 1Mpx [16].** TESPEC is able to reconstruct static objects that are invisible in recent events, e.g., the car in front of the ego vehicle. This is beneficial to downstream perception tasks such as object detection.



(a) Ground-truth. (b) Prediction. (c) Ground-truth. (d) Prediction.

Figure 2. **Qualitative object detection results on Gen1 [2].** Thanks to the long-term information learned during the self-supervised pre-training stage, the model successfully detects cars, even when they are not clearly visible in the input data.

Recon. Target	DDD17	
	mIoU ↑	mAcc ↑
Eq. (6)	63.835	70.180
Eq. (7)	**65.187**	**72.871**

Table 1. **Ablation on the reconstruction target.** We report downstream performance on DDD17 [1] semantic segmentation.

Pre-training Objective	DSEC	
	mIoU ↑	mAcc ↑
CPC [15]	59.921	67.486
ECDDP [25]	61.012	68.231
MAE [10]	**62.774**	**70.612**

Table 2. **Ablation on the pre-training objective.** We report downstream performance on DSEC [7] semantic segmentation.

aloading mechanism from RVT [6]. This approach allows the model to be trained on minute-long sequences.

Recurrent Backbone. Following recent work [25], we choose the Swin Transformer [14] architecture with a window size of 7 (referred to as Swin-T/7) as the encoder for both pre-training and downstream tasks. The implementation of this architecture is borrowed from Timm [23]. Since we employ a 2D histogram representation with 20 bins for events, the first embedding layer of Swin-T/7 is modified to accept 20 input channels instead of 3. For baselines where pre-trained weights with a different representation are used, the embedding layer's weights are repeated to match the 20 channels. A ConvLSTM [18] layer is inserted after each Swin Block to add recurrency into the backbone. The number of parameters for backbones and task-specific heads used in this work are show in Tab. 3.

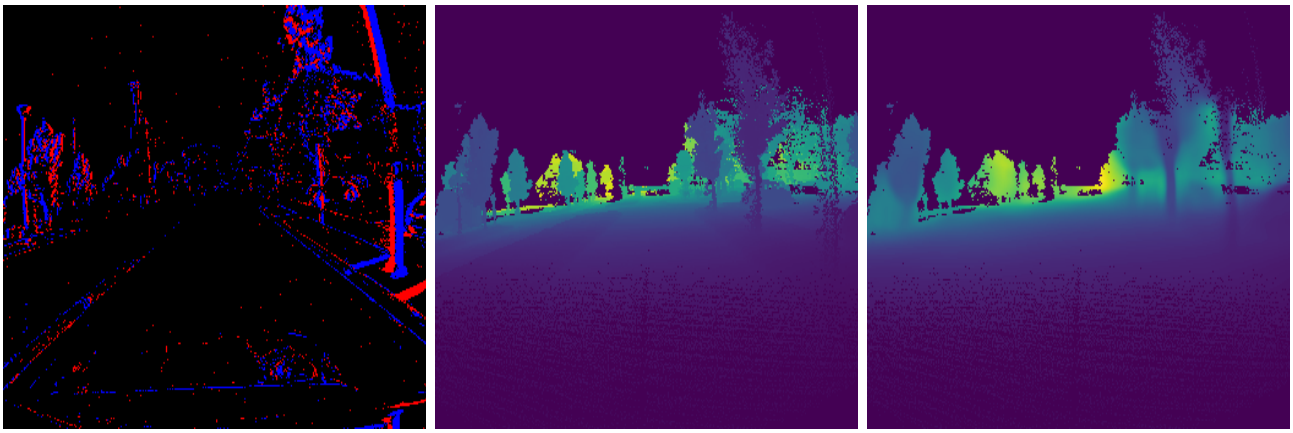
Optimization. We use the Adam [12] optimizer for pre-training and all tasks, along with a learning rate scheduler. The schedulers, learning rates, and warm-up steps are specified in Tab. 4. The initial and final learning rates are set to the peak learning rate divided by c_1 and c_2, respectively.

Evaluation. For evaluating feedforward models during test time, an event segment is extracted for each label from the corresponding event stream. As a result, the inputs for consecutive labels may overlap. In contrast, recurrent models can process the events of an entire stream only once and predict all labels for that stream. This approach requires setting T based on the label frequency for each dataset in downstream tasks. Since the time intervals between consecutive labels may vary slightly, all events occurring between two labels are used as the input for the later label.

(a) Input events. (b) Ground-truth. (c) Prediction.

(d) Input events. (e) Ground-truth. (f) Prediction.

Figure 3. **Qualitative semantic segmentation results on DSEC [7] and DDD17 [1].** Figures (a-c) are from the DSEC dataset, and figures (d-f) are from the DDD17 dataset. Leveraging long-term information learned in the pre-training stage, the fine-tuned models achieve high accuracy in predicting segmentation maps across datasets with varying resolutions and sparse event frames.



(a) Input events. (b) Ground-truth. (c) Prediction.

Figure 4. **Qualitative monocular depth estimation results on MVSEC [27].** Thanks to the long-term information learned with TESPEC, the model detects the surfaces of various objects and predicts accurate depth maps from sparse event frames.

C.1. TESPEC Pre-training

Our pre-training hyper-parameters are detailed in Tab. 4a. **Dataset.** We use the 1Mpx [16] dataset as the default for pre-training. This dataset contains approximately 15 hours of autonomous driving scenarios, captured during both daytime and nighttime. The original resolution of the dataset

Module	Parameters (M)
Feedforward backbone	27.5
Recurrent backbone	33.8
TESPEC decoder head	3.7
Object Detection head	12.9
Semantic Segmentation head	11.2
Monocular Depth Estimation head	4.1

Table 3. Size of backbones and task-specific heads.

is 720×1280, but we downsample it by 2 to reduce computational costs. Compared to other event camera datasets, 1Mpx features a higher resolution and a greater number of moving objects, resulting in more diverse motion patterns.

Data Representation. We set T to 50ms for pre-training.

Masking. To perform the masking, each bin of the histograms is divided into 32×32 non-overlapping patches. Then, 50% of the patches are randomly masked. The masked patches are replaced with a 32×32 learnable parameter. The same mask is applied across all bins within a single training step following VideoMAE [22].

Architecture. We adopt an asymmetric design for the encoder-decoder paradigm, where the decoder consists of a single Swin Block with a window size of 7. The Swin block is followed by a convolutional layer to match the reconstructed output size with the input size.

Loss Function. We apply the Mean Squared Error (MSE) loss only to the masked patches. The loss value is computed between the predictions and the normalized ground-truth patches following prior works [10, 21].

C.2. Object Detection

We fine-tune our models on the Gen1 [2] and 1Mpx [16] datasets for object detection as one of the downstream tasks.

Datasets. The characteristics of the 1Mpx dataset are described in Appendix C.1. Gen1 dataset is a dataset for detecting objects from event cameras mounted on vehicles. It contains 2,358 event sequences, each lasting 60 seconds (39 hours in total) with a resolution of 304×240 pixels.

Data Representation. We set T to 50ms for all models following RVT [6].

Architecture. We adopt the RVT architecture design for object detection, with one key difference: we replace their recurrent backbone with our Swin-T/7 recurrent backbone. Specifically, we use the YOLOX framework [4], which includes intersection over union (IoU) loss, classification loss, and regression loss. These losses are averaged over both the batch and sequence length for each optimization step.

Training. We fine-tune our models for 400,000 training steps. Experimentally, we found that a learning rate of 1×10^{-4} achieves the best performance on the 1Mpx dataset when pre-training is conducted on the same dataset. We hypothesize that this is because the pre-trained model already extracts informative representations of the data, which reduces the need for a larger learning rate during fine-tuning. All other hyper-parameters are detailed in Tab. 4b.

C.3. Semantic Segmentation

We fine-tune our models on the DSEC [7] and DDD17 [1] datasets for semantic segmentation.

Datasets. DSEC and DDD17 are both autonomous driving datasets. The DSEC dataset has a resolution of 640×480 and contains 53 sequences. However, semantic maps are available for only 11 of them. The DDD17 dataset is relatively longer, with a resolution of 346×260 and 40 sequences. Similar to DSEC, semantic maps for DDD17 are provided for only 6 sequences.

Data Representation. DSEC and DDD17 labels are available at 20 Hz and 33 Hz, respectively. Consequently, T is set to 50 ms for DSEC and 30 ms for DDD17 when using recurrent models. For feedforward baselines, T is set to 100 ms to avoid extremely sparse inputs.

Architecture. We fine-tune the backbones with an attached UperNet [24] decoder.

Loss Function. We use the sum of cross-entropy and Dice loss [19] as the loss function, as suggested in [20].

C.4. Monocular Depth Estimation

We fine-tune our models on MVSEC [27] dataset for the monocular depth estimation task.

Dataset. The MVSEC dataset consists of event data and grayscale images recorded by a DAVIS event camera with a resolution of 346×260 pixels, mounted on a driving car. Ground-truth depth maps are recorded at 20 Hz by a LiDAR sensor. The dataset contains several sequences captured during daytime and nighttime. Following ECDDP [25], we use the "outdoor_day2" sequence for fine-tuning and the "outdoor_day1", "outdoor_night1", "outdoor_night2", and "outdoor_night3" sequences for evaluation.

Data Representation. T is set to 50 ms for recurrent models, based on the MVSEC label frequency (20 Hz). For feedforward models, T is set to 100 ms.

Architecture. We attach a depth prediction head from MiDaS [17] to our backbone.

Loss Function. Following HMNet [9], we train the model to predict the normalized log depth $\hat{d}$, defined as:

$$\hat{d} = \frac{1}{\alpha} \log \frac{d}{d_{\text{max}}} + 1, \tag{1}$$

where d is the metric depth, d_{max} is the maximum depth in the dataset, and α is determined by the ratio between the maximum depth d_{max} and the minimum depth d_{min}:

$$\alpha = \log \frac{d_{\text{max}}}{d_{\text{min}}}. \tag{2}$$

For the MVSEC dataset, d_{max} and α equal to 80 and 3.7.

Architecture	Recurrent
Dataset	1Mpx
Effective Batch Size	8
Peak Learning Rate	2e-4
c_1, c_2	25, 1,000
Optimizer	Adam
Scheduler	Linear
Training Steps	400,000
Warm Up Steps	2,000
T	50ms
Sequence Length	15
Masking Ratio	50%
N	5,000
GPU	4× RTX6000

(a) TESPEC pre-training stage

Architecture	Recurrent		Feedforward	
Dataset	Gen1	1Mpx	Gen1	1Mpx
Effective Batch Size	8	24	64	32
Peak Learning Rate	2e-4	3.46e-4	2e-4	3.46e-4
c_1, c_2	25, 1,000			
Optimizer	Adam			
Scheduler	Linear			
Training Steps	400,000			
Warm Up Steps	2,000			
T	50ms			
Sequence Length	21	5	1	
GPU	2× RTX6000		1× RTX6000	

(b) Object Detection

Architecture	Recurrent	Feedforward
Dataset	MVSEC	
Effective Batch Size	8	16
Peak Learning Rate	1e-4	
c_1, c_2	25, 1,000	
Optimizer	Adam	
Scheduler	Cosine	
Training Steps	20,000	
Warm Up Steps	100	
T	50ms	100ms
Sequence Length	10	1
$d_{\max}$, α	80, 3.7	
GPU	1× RTX6000	

(c) Monocular Depth Estimation

Architecture	Recurrent		Feedforward	
Dataset	DSEC	DDD17	DSEC	DDD17
Effective Batch Size	8	16	16	
Peak Learning Rate	1e-4			
c_1, c_2	25, 1,000			
Optimizer	Adam			
Scheduler	Cosine			
Training Steps	50,000			
Warm Up Steps	250			
T	50ms	30ms	100ms	
Sequence Length	10	15	1	
GPU	2× RTX6000		1× RTX6000	

(d) Semantic Segmentation

Table 4. **Implementation Settings.** We report the implementation details for both pre-training and downstream tasks.

Augmentation	Probability	Magnitude	
		min	max
Horizontal Flip	0.5	-	-
Apply Zoom	0.8	-	-
Zoom In	0.8	1	1.5
Zoom Out	0.2	1	1.2

(a) TESPEC, Object Detection, and Semantic Segmentation.

Augmentation	Probability
Horizontal Flip	0.5

(b) Monocular Depth Estimation.

Table 5. **Augmentation Transformations.** Description of data augmentations used during training.

	ECDP	ECDDP	TESPEC
Runtime	45h	120h	135h

Table 6. **Runtime comparison of event-based SSL methods.** Each experiment is conducted on four RTX6000 GPUs.

We train our models using the same loss function as in previous work [5]. We compute a weighted sum of the scale-invariant loss [3] and the multi-scale scale-invariant gradient matching loss [13] as the loss function. The weights for the scale-invariant loss and the gradient matching loss are set to 1 and 0.125, respectively.

C.5. Pre-training Runtime & Model Size Analysis

As shown in Tab. 6, TESPEC only adds a marginal overhead in pre-training runtime. Accumulating events to estimated intensity videos can be implemented efficiently on GPUs, taking less than 1 ms at each step. Detailed training and in-

Recurrent	Semantic Seg.		Depth Est.	Object Det.	
	DSEC	DDD17	MVSEC	Gen1	1Mpx
Training time (in hours). Training settings are in Tab. 4.					
No	12.3	6.1	1.7	104.3	120.0
Yes	23.1	14.8	4.6	138.9	155.2
Inference time on 1× RTX6000 (in milli-seconds).					
No	13.0	12.7	12.4	18.7	19.1
Yes	14.6	14.0	13.8	20.4	20.9

Table 7. Training and inference time for downstream datasets.

ference runtime on downstream tasks for both feed-forward and recurrent backbones are provided in Tab. 7.

We report model size of our backbone and task-specific heads in Tab. 3. Event-based SSL baselines such as ECDP [26] and ECDDP [25] use the same backbone as ours. However, the model size for their heads is not reported in the papers, and their open-source codebase only implements the semantic segmentation task. Thus, we only know their segmentation head is $3\times$ larger than ours.

D. Limitations and Future Work

Our approach demonstrates significant improvement across several downstream tasks. However, the recurrent model training requires processing of multiple stages in each iteration, which increases the training time and memory consumption. Nevertheless, our recurrent backbone only adds lightweight recurrent modules over its feedforward counterpart, resulting in comparable inference time.

One of the future directions is to explore alternative recurrent architectures that are better suited for event data. Prior work [28] has demonstrated that state space models [8] can achieve strong performance with smaller architectures. These characteristics offer a promising direction to reduce training time while maintaining competitive results.

Another potential direction is the use of more diverse datasets. As shown in our experiments, the 1Mpx [16] dataset proves to be a better candidate than the Gen1 [2] dataset for pretraining. The 1Mpx dataset features a higher resolution and greater number of moving objects, which result in more diverse motion patterns. However, the 1Mpx dataset is limited to autonomous driving scenarios. We believe that a dataset containing a broader range of movement scenarios could enable the model to extract more generalizable representations from the environment.

References

[1] Jonathan Binas, Daniel Neil, Shih-Chii Liu, and Tobi Delbruck. DDD17: End-to-end davis driving dataset. *arXiv preprint arXiv:1711.01458*, 2017. 1, 2, 3, 4

[2] Pierre De Tournemire, Davide Nitti, Etienne Perot, Davide Migliore, and Amos Sironi. A large scale event-based detection dataset for automotive. *arXiv preprint arXiv:2001.08499*, 2020. 1, 2, 4, 6

[3] David Eigen, Christian Puhrsch, and Rob Fergus. Depth map prediction from a single image using a multi-scale deep network. *NeurIPS*, 2014. 5

[4] Zheng Ge, Songtao Liu, Feng Wang, Zeming Li, and Jian Sun. Yolox: Exceeding yolo series in 2021. *arXiv preprint arXiv:2107.08430*, 2021. 4

[5] Daniel Gehrig, Michelle Rüegg, Mathias Gehrig, Javier Hidalgo-Carrió, and Davide Scaramuzza. Combining events and frames using recurrent asynchronous multimodal networks for monocular depth prediction. *IEEE Robotics and Automation Letters*, 2021. 5

[6] Mathias Gehrig and Davide Scaramuzza. Recurrent vision transformers for object detection with event cameras. In *CVPR*, 2023. 2, 4

[7] Mathias Gehrig, Willem Aarents, Daniel Gehrig, and Davide Scaramuzza. DSEC: A stereo event camera dataset for driving scenarios. *RA-L*, 2021. 1, 2, 3, 4

[8] Albert Gu, Tri Dao, Stefano Ermon, Atri Rudra, and Christopher Ré. Hippo: Recurrent memory with optimal polynomial projections. *NeurIPS*, 2020. 6

[9] Ryuhei Hamaguchi, Yasutaka Furukawa, Masaki Onishi, and Ken Sakurada. Hierarchical neural memory network for low latency event processing. In *CVPR*, 2023. 4

[10] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *CVPR*, 2022. 2, 4

[11] Olivier Henaff. Data-efficient image recognition with contrastive predictive coding. In *ICML*, 2020. 1

[12] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 2

[13] Zhengqi Li and Noah Snavely. Megadepth: Learning single-view depth prediction from internet photos. In *CVPR*, 2018. 5

[14] 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. 2

[15] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018. 1, 2

[16] Etienne Perot, Pierre De Tournemire, Davide Nitti, Jonathan Masci, and Amos Sironi. Learning to detect objects with a 1 megapixel event camera. *NeurIPS*, 2020. 2, 3, 4, 6

[17] René Ranftl, Katrin Lasinger, David Hafner, Konrad Schindler, and Vladlen Koltun. Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer. *TPAMI*, 2020. 4

[18] Xingjian Shi, Zhourong Chen, Hao Wang, Dit-Yan Yeung, Wai-Kin Wong, and Wang-chun Woo. Convolutional lstm network: A machine learning approach for precipitation nowcasting. *NeurIPS*, 2015. 2

[19] Carole H Sudre, Wenqi Li, Tom Vercauteren, Sebastien Ourselin, and M Jorge Cardoso. Generalised dice overlap as a deep learning loss function for highly unbalanced segmentations. In *DLMIA*, 2017. 4

[20] Zhaoning Sun, Nico Messikommer, Daniel Gehrig, and Davide Scaramuzza. ESS: Learning event-based semantic segmentation from still images. In *ECCV*, 2022. 4

[21] Zhan Tong, Yibing Song, Jue Wang, and Limin Wang. VideoMAE: Masked autoencoders are data-efficient learners for self-supervised video pre-training. *NeurIPS*, 2022. 4

[22] Limin Wang, Bingkun Huang, Zhiyu Zhao, Zhan Tong, Yinan He, Yi Wang, Yali Wang, and Yu Qiao. VideoMAE V2: Scaling video masked autoencoders with dual masking. In *CVPR*, 2023. 1, 4

[23] Ross Wightman. Pytorch image models, 2019. 2

[24] Tete Xiao, Yingcheng Liu, Bolei Zhou, Yuning Jiang, and Jian Sun. Unified perceptual parsing for scene understanding. In *ECCV*, 2018. 4

[25] Yan Yang, Liyuan Pan, and Liu Liu. Event camera data dense pre-training. In *ECCV*, 2023. 1, 2, 4, 6

[26] Yan Yang, Liyuan Pan, and Liu Liu. Event camera data pre-training. In *ICCV*, 2023. 6

[27] Alex Zihao Zhu, Dinesh Thakur, Tolga Özaslan, Bernd Pfrommer, Vijay Kumar, and Kostas Daniilidis. The multivehicle stereo event camera dataset: An event camera dataset for 3d perception. *RA-L*, 2018. 3, 4

[28] Nikola Zubic, Mathias Gehrig, and Davide Scaramuzza. State space models for event cameras. In *CVPR*, 2024. 6