Dense Event Stereo Depth Estimation by a CNN-guided Spiking Neural Network

Zixuan Wang

Nanjing University

Abstract. Event-based sensor, by mimicking the human retina, can asynchronously capture event streams with high temporal resolution, low power consumption, and low latency, presenting great potential on high-speed stereo depth estimation. Currently, learning-based SNN model is proposed to estimate depth from a stereo event camera setup. However, the spiking vanishing phenomenon in deep layers of SNN brings huge performance degradation. To solve this problem, in this work, we present a Feature Supervision Module (FSM) to distill knowledge from CNNs into the SNN model in the training phase while still only using event data and pure SNN architecture in the testing phase. A Mask Regularization Module (MRM) is introduced to achieve elegant edge reconstruction performance at changing regions. The experiment shows that our method receives state-of-the-art quantitative and qualitative results on the Multi Vehicle Stereo Event Camera Dataset (MVSEC).

Keywords: Event Stereo, Spiking Neural Network, CNN Guidance.

1 Introduction

Inspired by human retina, event-based sensors (event cameras), such as *Dynamic Vision Sensor* (DVS) [23] and *Dynamic and Active-pixel Vision Sensor* (DAVIS) [4, 6], are designed to detect logarithm illumination change and output asynchronous binary event stream, which can achieve high dynamic range (∼100dB), microsecond temporal resolution, low latency, and much lower power consumption [13]. With the wide application of event cameras in different fields [26, 7, 32, 48, 45], the binocular event camera systems [47, 14] are proposed for stereo estimation, and many CNN-based algorithms have got promising results [48, 40, 1, 28, 50]. However, the extensive computational cost of CNN-based algorithms greatly impairs the main advantages of event cameras, i.e., the high speed, low latency and low power consumption caused by the sparse events.

Recently, various researches show that Spiking Neural Networks, which only produce binary spikes through spiking neurons in every layer, can be directly implemented with compact neuromorphic vision systems [31, 15], offering a perfect choice for such event-based scenario. However, as mentioned in [22, 21], the spiking vanishing phenomenon brings performance degeneration for deeper SNN architecture, impeding its applications on complex tasks like event stereo.

In this paper, we propose to use the knowledge distillation mechanism to teach the SNN with a well trained CNN. Due to the rich information and the performance superiority of CNNs, transferring knowledge and information of CNN can help SNN generalize better. In the event stereo task, the corresponding intensity images are usually available as for the datasets [47, 14], so that we can utilize the CNNs trained on the normal images as guidance, and supervise the training of SNN with proper SNN-compatible representations. Instead of using hybrid CNN-SNN architectures as in [21, 44], we try to propose a pure-SNN-in-inference architecture. Specifically, we add a feature-level module called Feature Supervision Module which receives spikes from the encoder and output spiking features, and supervise the intermediate spiking features with constant features extracted from a pre-trained CNN architecture. As for the decoder part, in order to reach elegant edge reconstruction performance at changing regions, we design a Mask Regularization Module with IFNeurons, whose spikes are supervised by the binary mask of depth map generated from a pre-trained edge detection network (HEDNet [43]).

To the best of our knowledge, it is the first time that the training process of SNN is supervised by CNN guidance on the event stereo task, whilst the network is still fully spiking during inference. In summary, our contributions can be described as follows:

- We *propose* a method to train the pure-SNN-in-inference architecture with CNN guidance for the binocular event camera stereo task.
- We *design* a Feature Supervision Module (FSM) to leverage the frame image features to enhance the expressiveness of SNN encoder.
- We *present* a Mask Regularization Module (MRM) combined with the decoder to optimize the predicted depth image result.
- We *achieve* state-of-the-art quantitative and qualitative results among SNN methods on the publicly available dataset MVSEC.

2 Related Work

Stereo Event Depth Estimation. With the large-scale application of deep learning, some learning-based CNN methods for stereo event depth estimation are proposed. The first supervised method for dense event stereo depth estimation is proposed by Tulyakov *et al.* [40], which modifies an intensity image stereo network [41] and designs a novel "event queue" embedding as the network input. Ahmed *et al.* [1] further improve this network and get more accurate results by integrating the intensity image reconstruction during training. Mostafavi *et al.* [28] combines events and intensity frames in a sequential manner, and Zuo *et al.* [50] use a hybrid event-RGB setup and propose a novel pyramid attention module to achieve better performance. Although CNN algorithms perform well on modern GPUs, it is hard deploy them on the low-power edge-computing hardwares. Further, the operation mode of CNN does NOT fit the event streams very well, while SNN fits the event data much better for it follows the bio-mechanism as well. Recently, researchers also attempt to complete event stereo

matching task by Spiking Neural Networks. Osswald *et al.* [31] propose a spiking stereo neural network inspired by [27] and demonstrate its features with a neuromorphic hardware system, and [11, 2, 36] complete event stereo matching using neuromorphic vision setup with low latency. Barbier *et al.* [3] use unsupervised learning to train a SNN with a simple cell layer and a complex cell layer. However, the above methods either have limited disparity map resolution, shallow structure or are not learning-based. Recently, Ranccon *et al.* [34] propose the first learning-based supervised deep UNet-like SNN architecture for event stereo depth estimation.

Learning Strategy of SNN. There are generally three ways to train a SNN. Spike-Timing Dependent Plasticity (STDP) [9] has been the dominant unsupervised strategy for training SNN. Another popular choice is the ANN-to-SNN conversion [8, 37]. These methods firstly train a CNN with SNN constraints, then transfer weights of CNN to SNN. Apart from that, various supervised error backpropagation methods are developed, such as SpikeProp [5], SLAYER [38], spike-timing dependent backpropagation (STDB) [35], and spatial-temporal backpropagation (STBP) [42]. Recently, Fang *et al.* also develop an open-source Python package, i.e., SpikingJelly [12], which provides different surrogate gradient learning methods [29] and miscellaneous types of spiking neurons. However, although the training of SNNs becomes simple, the performance gap between SNN and CNN in large-scale computer vision tasks is still a problem. There are a few works that leverage the benefits of these two paradigms. Lee *et al.* [21] construct a SNN encoder and a CNN decoder network to estimate optical flow from events, and Zhang *et al.* [44] address the problem of disturbances in synthetic aperture imaging under very dense occlusions and extreme lighting conditions in the same way. Such methods unlock the potential of SNN-CNN combination during training, but none of them train SNN with CNN guidance and inference in pure SNN mode.

Edge Guidance for Computer Vision Tasks. Edge information is widely used in various CNN-based computer vision algorithms. For example, Luan *et al.* [24] utilize the semantic segmentation of the input image to address the content-mismatch problem in style transfer, and Ma *et al.* [25] preserve the structure in single image super resolution by exploiting gradient maps of images. Zhu *et al.* [49] propose an Edge-Enhanced Multi-Exposure Fusion Network to enhance low-light images, while Huang *et al.* [18] propose a lightweight light-field depth estimation network with edge guidance to recover detailed textures. Such structural information can improve the performance on changing regions to a large extent. Inspired by these methods, we propose to supervise the spikes by a binary mask generated from an edge detection CNN model (i.e. HEDNet [43]) to elegantly reconstruct the edge details of the depth.

3 Stereo SNN Trained with CNN Guidance

In order to improve the performance of SNN, we propose a pure-SNN-in-inference network trained with a counterpart CNN guidance for stereo depth estimation

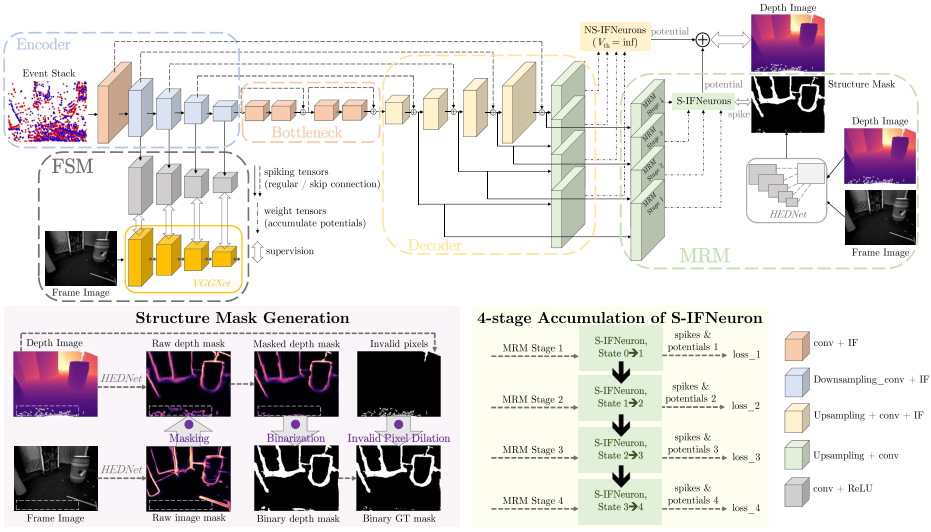

Fig. 1: Overview of the entire network.

from binocular event camera system. The overview of the proposed method is shown in Fig. 1. The UNet structure is applied as backbone of the SNN network, and a Feature Supervision Module (FSM) is used for transferring the intermediate feature knowledge learned by the CNN network from the intensity images to the proposed stereo SNN during training process. A Mask Regularization Module (MRM) is introduced to further improve the performance of proposed method in edge regions. Details about the network are discussed as follows.

3.1 Basic Model and Network Overview

For an event camera, the event stream follows the format of (t, x, y, p), where t denotes the timestamp, (x, y) denotes the coordinate, and $p \in \{1, 0\}$ denotes the event polarity. Events are generated only when over-threshold log pixel intensity change happens, and polarity of 1 denotes the increase of intensity while polarity of 0 denotes the decrease.

To utilize the event data, many CNN algorithms design complicated descriptors to represent events. In this paper, benefiting from the working mode of SNN, we can process them in a more model-friendly way. Specifically, we directly convert monocular events to a 2-channel event image to preserve the event dynamics in a time range of t ms, where the first channel counts the number of positive events at every pixel and the second channel counts the number of the negative ones. Although this may bring non-binary values, it can be addressed by multi-bit hardware or serial operations [34]. Finally the binocular event images are stacked in the channel dimension, which means that the size of the input is $[4, H, W]$ and $[H, W]$ is the spatial resolution of the event camera.

We use IFNeurons in our overall network, which can integrate synaptic current from previous connected neurons, update membrane potential, fire spike

and reset membrane potential if the potential exceeds the threshold (otherwise remain constant). The mathematical expression of the dynamics between layer l and $l+1$ is

$$\begin{aligned} V_{t+1}^{l+1}(i) &= V_t^{l+1}(i)(1-o_t^{l+1}(i)) + \sum_{j=0}^{l(n)} W(i,j)o_{t+1}^{l}(j), \\ o_{t+1}^{l+1}(i) &= f(V_{t+1}^{l+1}(i) - V_{\text{th}}), \end{aligned} \tag{1}$$

where $V(\cdot)$ is the membrane potential, $o(\cdot)$ is the output spike of neuron (0 or 1), $W(i,j)$ is the convolutional weight between neuron i at layer $l+1$ and neuron j at layer l, $f(\cdot)$ is the non-differentiable Dirac function, and V_{th} is the threshold. In order to train the network in a gradient-based way, we replace Dirac function with arctan forward function

$$g(x) = \frac{1}{\pi}\arctan(\frac{\pi}{2}\alpha x) + \frac{1}{2}, \tag{2}$$

whose gradient is

$$g'(x) = \frac{\alpha}{2(1+(\frac{\pi}{2}\alpha x)^2)}. \tag{3}$$

Backbone with UNet Structure. The backbone structure includes encoder, bottleneck, and decoder [34], with the input of event stack in t ms. The encoder consists of conv-IFNeuron blocks to extract multi-scale features, where downsampling is done by 2-strided convolution. The bottleneck follows the structure of ResNet [16] with residual connection. In the decoder, each yellow cube contains upsampling, convolution and IFNeurons layers, upsampling features $2\times$. The green cubes upsample the features of encoder at different scales to the original resolution, followed by 1-strided convolution to reduce the features to single channel. The NS-IFNeurons (non spiking IFNeurons) with $V_{\text{th}} = \infty$ and size $[H, W]$ receives the 4-stage decoder outputs sequentially to accumulate potential, and the final potential represents the depth.

3.2 Feature Supervision Module

According to the principle of event cameras, most detected events are distributed on the edges of the scene. Therefore, the stack of events looks alike an edge map which contains basic structural information. However, it always suffers from noise and the lack of detailed information that frame images can present, thus would bring inaccuracy and inefficiency into the network encoder.

On the contrary, the generic and classical deep convolutional neural networks such as VGGNet [39] which is trained with different large-scale datasets is able to implement various demanding computer vision tasks. These models are capable of extracting rich and meaningful features from the input and have strong expressiveness, therefore we believe that this kind of information can also serve as cross-modality guidance to enhance the encoder part of SNN.

As shown in Fig. 1, the frame image whose timestamp is the closest to that of the ground truth depth map is fed into a VGGNet [39] pre-trained on ImageNet [10]. In the VGGNet [39], features are downsampled by max-pooling layers after the convolutional layers. The features of the first four scales are used for guiding the SNN training in our scenario with 64, 128, 256, and 512 channels respectively. The spikes of the first four scales of the SNN encoder are fed into a spike-to-float transfer block composed of a spike-counter, a convolution layer and a ReLU layer. Through this spike-to-float transfer block, the spike output of SNN encoder could be compared with the floating features computed by CNN architectures, guaranteeing that the encoder can extract efficient and effective features as CNN at different scales in the spiking domain. Specifically, the transfer operation could be derived by

$$f = ReLU(Conv(\sum_{t\in(t_0-T,t_0)} Spike)), \tag{4}$$

where f is the transfered floating features, $ReLU(\cdot)$ and $Conv(\cdot)$ represent the ReLU and convolutional layers, t_0 is the current time for training (i.e., the timestamp of the depth map), and T is the integration time for SNN features, in our experiment T is set to 50 ms. As for bakpropagation, the gradients jump the spike-counter directly, linking both the SNN network and the CNN supervisor seamlessly in training process.

3.3 Mask Regularization Module

Although events can effectively store edge information, in the experiment, the reconstruction of depth image still cannot well get rid of blur and imprecise details. Therefore, we propose to add an Mask Regularization Module (MRM) branch in the reconstruction process to improve the performance around edge.

As show in Fig. 1, four upsampling-conv layers in MRM receive the multi-scale intermediate spiking features of the SNN decoder and resize the feature size to the original resolution. The weighted spiking output from these upsampling-conv layers are fed into MRM supervising S-IFNeurons (spiking IFNeurons) with size $[H, W]$ consecutively. Considering the edge mask we used for supervision here is binary, we directly use a cross entropy between the spiking states of the MRM supervising S-IFNeurons and the binary edge mask as the MRM loss here.

In addition to providing supervision during training, we also utilize the information in MRM module like attention mechanism. Specifically, we add the potentials of the MRM supervising S-IFNeurons before spiking to the NS-IFNeurons potentials.

For generating the structure mask for supervision, as shown in Fig. 1, we first use a pre-trained HEDNet [43] to detect the raw depth structure S_{d} from depth images. To eliminate the artifacts caused by invalid pixels in depth images, we also detect the structure mask of the frame image S_{img} and compute the final mask by

$$M_{\mathrm{final}} = Bin_{\tau_2}(S_{\mathrm{d}} \odot Bin_{\tau_1}(S_{\mathrm{img}})) \cap Dilation(M_{\mathrm{false}}), \tag{5}$$

where M_{final} is the final mask used for MRM supervision, $Bin_{\tau}(\cdot)$ is the thresholding-based binarization operator, τ_1 and τ_2 are the thresholds, $\odot$ is the dot product operation, M_{false} is the mask of the invalid pixel in the depth map, and $Dilation(\cdot)$ denotes the dilation operator.

3.4 Losses and Training Strategy

We train our network in a fully-supervised way using STBP [42], with supervision from depth image, CNN features, and structure map. For depth regression, we use scale-invariant loss and multi-scale scale-invariant gradient matching loss as [17, 34]. Given the output depth $\hat{d}$ and ground truth depth d, denoting the residual $R = \hat{d} - d$, the scale-invariant loss is defined as

$$l_{\text{si}}(R) = \frac{1}{N}\sum_{\mathbf{u}}(R(\mathbf{u}))^2 - \frac{1}{N^2}(\sum_{\mathbf{u}} R(\mathbf{u}))^2, \tag{6}$$

where N is the number of valid pixels $\mathbf{u}$ in the ground truth.

The gradient matching loss aims at keeping smooth depth changes and sharp depth discontinuities, which is defined as

$$l_{\text{grad}}(R) = \frac{1}{N}\sum_{\mathbf{u}} |\nabla_x R(\mathbf{u})| + |\nabla_y R(\mathbf{u})|, \tag{7}$$

where ∇ means gradient calculation using Sobel filters. Therefore, the total loss for depth regression is $l_{\text{pix}}(d, \hat{d}) = l_{\text{si}} + \lambda_0 l_{\text{grad}}$, where λ_0 is a hyper-parameter to balance between two losses.

To supervise the SNN encoder output, we use smooth L1 loss to compare VGGNet [39] features f extracted from frame images and SNN features $\hat{f}$, which is defined as

$$l_{\text{FSM}}(f, \hat{f}) = \frac{1}{M}\sum_{(i,j,k)} \text{SmoothL1}(f(i,j,k) - \hat{f}(i,j,k)), \tag{8}$$

where M is the total number of elements in the current 3-dimensional feature map, and (i, j, k) is the coordinate. According to mask supervision between spiking neurons output $\hat{y}$ and ground truth structure y, we define a soft weighted binary cross entropy loss

$$l_{\text{MRM}}(y, \hat{y}) = -\frac{1}{N}(\alpha \sum_{i \in y_+} \log \Pr(\hat{y}_i = 1) + \sum_{i \in y_-} \log(1 - \Pr(\hat{y}_i = 0))), \tag{9}$$

where y_+ is the number of pixels that have structure values in all valid pixels N, and $y_- = N - y_+$. We also define a class weight balancing factor $\alpha = y_-/y_+$. Since the output of MRM is already binary, instead of adding another conv layer and Sigmoid after spikes, we softly set $\Pr(\hat{y}_i = 1) = \sigma(1)$ and $\Pr(\hat{y}_i = 0) = \sigma(-1)$ where $\sigma(\cdot)$ denotes Sigmoid, since $\log(0)$ will bring infinity into our model.

As a result, the overall loss is $l_{\text{total}} = l_{\text{si}} + \lambda_0 l_{\text{grad}} + \lambda_1 l_{\text{FSM}} + \lambda_2 l_{\text{MRM}}$, where λ_0, λ_1, and λ_2 are hyperparameters for balancing different loss terms.

Besides, since the spiking output of the four scales are sequentially occurred, there will be four stage of the spiking features for the MRM supervision and the attention like potential addition for handling each frame. At each stage, both MRM supervising S-IFNeurons and NS-IFNeurons integrate the previously arrived spiking features, leading to the four stage supervision. Specifically, we compute the loss at each stage, and add them together after the entire chronology to derive the final loss.

To further analyze the effectiveness of Feature Supervision Module in our network, here we present the detailed backpropagation process in the encoder. Given pixel-level regression cost l_{pix}, the gradient on weights W^l between layer l and layer $l-1$ based on the chain rule will be

$$\frac{\partial l_{\text{pix}}}{\partial W^l} = \frac{\partial l_{\text{pix}}}{\partial o^l} \frac{\partial o^l}{\partial V^l} \frac{\partial V^l}{\partial W^l}, \tag{10}$$

where $\partial o^l / \partial V^l = g'(V^l - V_{\text{th}})$ due to Eq. 3, and $\partial V^l / \partial W^l = o^{l-1}$ due to Eq. 1. After adding our Feature Supervision Module, defining the l^{th} output of conv-ReLU as a^l, the extra weight gradient on weight in the encoder between layer l and layer $l-1$ in the encoder by feature cost l_{FSM} will be

$$\frac{\partial l_{\text{FSM}}}{\partial W^l} = \frac{\partial l_{\text{FSM}}}{\partial a^l} \frac{\partial a^l}{\partial o^l} \frac{\partial o^l}{\partial V^l} \frac{\partial V^l}{\partial W^l}, \tag{11}$$

where $a^l = ReLU(Conv(o^l))$. Therefore, the whole gradient of the cost on W^l will be $\partial l / \partial W^l = \partial l_{\text{pix}} / \partial W^l + \partial l_{\text{FSM}} / \partial W^l$. The extra gradient, especially $\partial a^l / \partial o^l$ part, will force the output spikes in the encoder to perform more perceptually similar to a feature map instead of random spikes, thus guiding the following layers to pay more attention on import regions.

4 Experiments

Datasets and Evaluation Metrics. The Multi Vehicle Stereo Event Camera Dataset (MVSEC) [47] is a publicly available dataset, which provides stereo event streams (260×346), grayscale images (260×346), IMU readings, and point clouds. Ground truth rectified depth images collected by Lidar every 50ms are also provided. Before training, we perform area closing operation of every depth image to remove dark structures as in [34]. The ground truth disparity is calculated by computing $d = f \times b / z$, where z, f and b denotes depth, focal length, and baseline, respectively. We test Split 1, Split 2 and Split 3 from *Indoor Flying* where $f \times b = 19.94$. Split 4 is not used in our experiments since it only contains scene of little-texture floor which limit the generalization ability of algorithms, as mentioned in [46, 40, 1]. The dataset is splited as in [46, 40]. To quantitatively evaluate the reconstruction performance, we utilize three metrics, i.e. *mean depth error*, *mean disparity error*, and *one pixel accuracy (1PA)*.

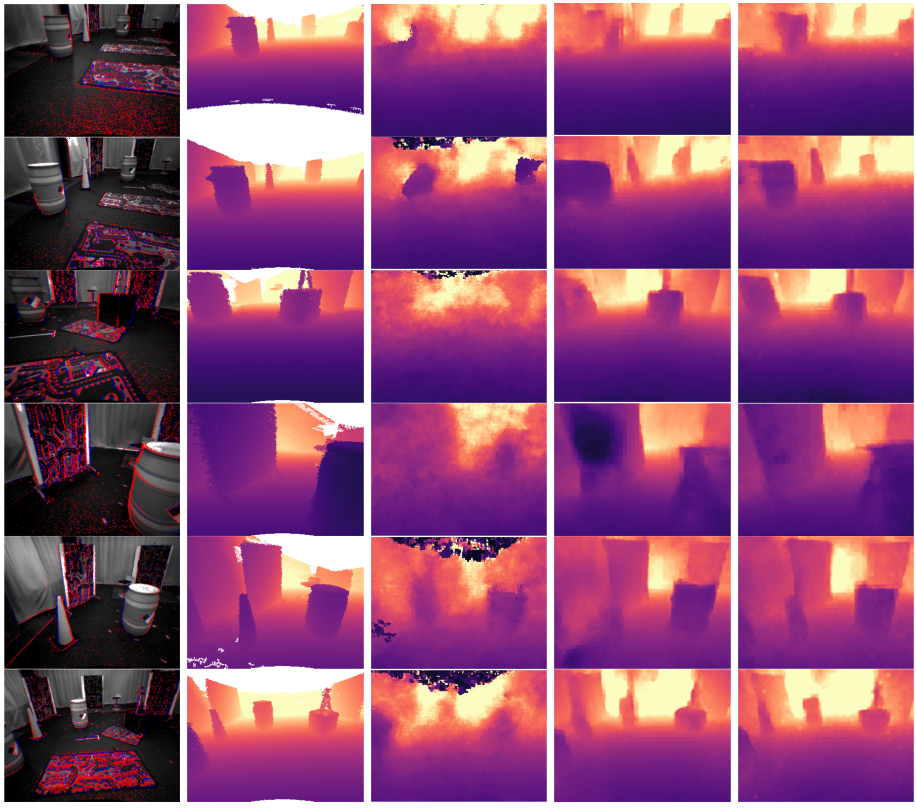

Fig. 2: Depth results from DDES-SNN, StereoSpike [34] and our method. Frame images are overlaid with red positive events and blue negative events. Pixels without disparities in the ground truth are colored white. Scenes from top to bottom are #698 and #778 of Split 1, #358 and #1033 of Split 2, #671 and #1010 of Split 3, respectively. Here we set the range of the output values to $[0, 5]$ and the color map is *magma*.

Implementation Details. The network is trained with PyTorch framework [33] and SpikingJelly [12]. We use Adam [19] as the optimizer with $\beta_1 = 0.9$, $\beta_2 = 0.99$. We train the network for 70 epochs in an end-to-end way, with the initial learning rate set as 0.0002 and decayed by multiplying 0.5 at epoch 8, 42, 60. The batch size is set to 1. The loss parameters for training are $\lambda_0 = 0.5$, $\lambda_1 = 1.0$, and $\lambda_2 = 0.75$. Spiking IFNeurons in the whole network are of the same parameters, with $V_{\text{th}} = 1$ and $V_{\text{reset}} = 0$. We set $\alpha = 2$ in arctan forward function. We adopt VGG11 from PyTorch [33] as the pre-trained VGGNet [39], and the open-source pytorch-hed package [30, 20] as the pre-trained HEDNet [43]. The thresholds in MRM is $\tau_1 = 0$ and $\tau_2 = 25$ respectively. Typically, it takes about 6 hours to train a single split on a NVIDIA RTX 2080 GPU.

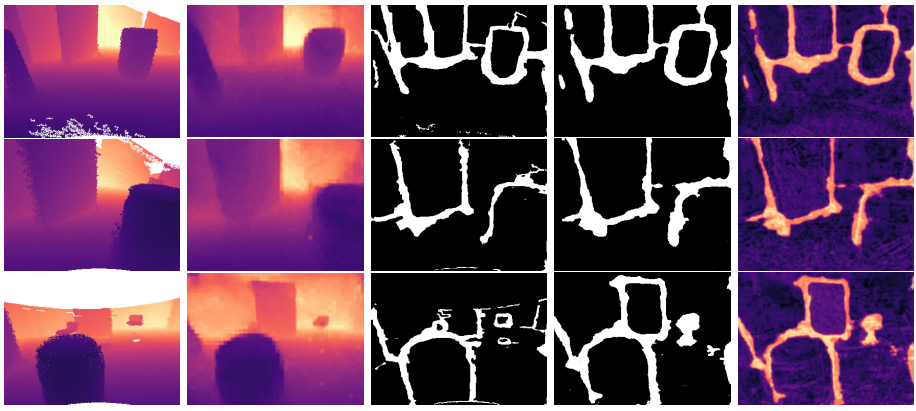

Fig. 3: Visualization of MRM on Split 1 #870, Split 2 #1057, and Split 3 #1309.

Table 1: Results on the *dense* MVSEC *Indoor Flying* dataset. The 2nd and 3rd row correspond to the CNN models and SNN models. The best and secondary performance are marked with bold and underlined text for CNN and SNN methods respectively.

	Mean Depth Error [cm]			Mean Disp. Error [pix]			One Pixel Accuracy [%]		
	S1	S2	S3	S1	S2	S3	S1	S2	S3
DDES [40]	15.93	26.47	20.38	0.58	**1.17**	**0.70**	90.65	69.14	86.74
DESN [1]	14.2	–	19.4	**0.55**	–	0.75	**92.1**	–	**89.6**
EIS [28]	**13.74**	**18.43**	22.36	–	–	–	89.0	**85.2**	88.1
HDES [50]	16.0	28.0	**18.0**	–	–	–	86.41	49.70	80.08
DDES-SNN	31.87	49.47	34.91	1.30	2.73	1.38	66.49	33.68	65.14
StereoSpike [34]	19.99	29.94	25.30	0.88	2.28	1.40	76.75	48.68	70.57
proposed	**17.99**	**26.90**	**22.54**	**0.76**	**2.04**	**1.33**	**81.13**	**52.23**	**75.44**

4.1 Comparisons with State-of-the-art Methods

We compare the proposed method with state-of-the-art algorithms, including four CNN-based methods, e.g., DDES [40], DESN [1], EIS [28] and HDES [50] and a SNN-based method StereoSpike [34]. Besides, to further demonstrate our method, we convert the CNN-based algorithm DDES [40] to the SNN version called "DDES-SNN" by replacing every conv-InstanceNorm-ReLU block with conv-InstanceNorm-IFNeuron combination. (Please refer to the supplementary material for more details.) Both DDES [40] and StereoSpike [34] are reimplemented according to their open-source code.

Quantitative Results. As shown in Tab. 1 and Tab. 2, there is a big gap on mean depth error and one pixel accuracy between DDES-SNN and the CNN-based methods. The proposed method outperforms the state-of-the-art SNN-based methods, approaching the performance of the CNN-based algorithms.

As for the sparse MVSEC dataset, results are shown in Tab. 2. It is worth noting that the ground truth depth are only captured for only around 15% pixels, making it difficult to generate faithful edge masks. Hence, we do no utilize the MRM for the sparse dataset and set $\lambda_1 = 1.25$ to guarantee stronger feature supervision on SNN due to the absence of MRM.

Qualitative Results. Fig. 2 shows the qualitative comparisons between the proposed method and the state-of-the-art SNN-based algorithms. The experiments show that DDES-SNN, even though with much deeper and more complicated structure than our UNet-like structure, is unable to reconstruct smooth depth map and contains many outlier pixels. On the contrary, StereoSpike [34] and our proposed method can still predict continuous values, and our method produces more accurate and fine details in edge regions.

To further evaluate the effectiveness of the MRM, we present the ground truth structure mask, the spikes of neurons, and potentials in Fig. 3. The results show that MRM learns the coarse mask and forwards meaningful structure information effectively to the decoder. Interestingly, although the ground truth structure mask does not contain mask information of invalid pixels, the module still manages to predict the full mask with the guidance of FSM.

4.2 Ablation Studies

Qualitative Results. In order to qualitatively compare the effect of different components in our network, we visualize the recovered depth under the guidance of each module in Fig. 4 on Split 2 #1074. The blue box denotes a continuous region while the yellow one denotes a discontinuous region. For better visualization, we increased the brightness of the part in the blue box by 40%. We can see that the baseline output has an obvious black part in the middle and suffers from edge distortion. FSM can efficiently recover the characteristic of non-edge part but the edge gets blurry. On the contrary, MRM regulates the edge sharper and more explicit but still has bad pixels in the continuous part. With the combination of these two modules, the proposed method can retrieve high-fidelity structure as the ground truth.

Quantitative Results. We conduct ablation experiments of different modules in the network on dense MVSEC and the results are shown in Tab. 3. As is

Table 2: Results on the *sparse* MVSEC Indoor Flying dataset. EIS [28] is not in the table since they don't provide test results on sparse MVSEC.

	Mean Depth Error [cm]			Mean Disp. Error [pix]			One Pixel Accuracy [%]		
	S1	S2	S3	S1	S2	S3	S1	S2	S3
DDES [40]	13.06	**16.37**	17.59	0.53	**0.91**	0.66	**92.18**	**79.18**	**88.76**
DESN [1]	**11.3**	–	**15.2**	**0.49**	–	**0.63**	–	–	–
HDES [50]	14.1	24.2	16.4	–	–	–	–	–	–
DDES-SNN	27.28	45.75	33.50	1.30	3.20	1.50	65.08	23.10	56.43
StereoSpike [34]	15.58	22.71	20.29	0.77	2.07	1.18	80.78	42.95	72.41
proposed	**14.73**	**20.91**	**19.86**	**0.72**	**1.96**	**1.10**	**82.74**	**47.51**	**75.18**

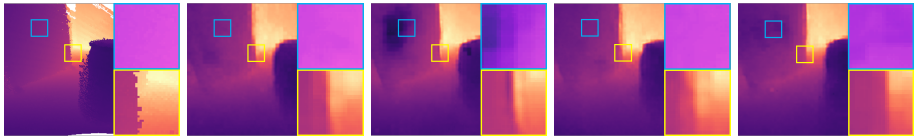

Fig. 4: Visual comparisons of ablation study.

shown, although two modules may have slightly different compacts on different splits, the combination of them can bring satisfactory results.

Feature Visualization. We also visualize the multi-scale features in the encoder to verify the contribution of FSM, as shown in Fig. 5, from Scale 0 (130×173) to Scale 3 (17×22). We present baseline spikes, w/ FSM spikes, w/ FSM features, and supervising VGG features. For feature channels of each scale, we select the first channel, middle channel, and last channel for more comprehensive comparison. It is obvious that with FSM, the location of the output spikes from the encoder shift from relatively random to important regions. For example, for Scale 3, the middle channel and the last channel of the baseline are blank and non-blank respectively, and after adding feature supervision module, the spiking mode is much more similar to the ground truth features, with non-blank middle channel and blank last channel. Additionally, the features after the conv-ReLU block is visually similar to the features extracted from VGGNet [39] especially in deeper layers, which further proves the efficiency of FSM.

Spiking Rates Ablation. As shown in Tab. 4, we test the spiking rates of different modules in our network on Split 1. The spiking rates of all modules increases with the introduction of FSM (Row 2), while the spiking rates of encoder and bottleneck fall when MRM is inserted into the network (Row 3) due to partial spiking assignment to MRM. With joint FSM-MRM combination, the spiking rates become more balanced.

Ablation on Structure for Extracting Guidance Features. To compare the effect of different types of guidance feature extraction network in FSM, we conduct experiments of various VGG-type network on dense MVSEC Indoor Flying Split 1 as shown in Tab. 5. For a fair comparison, we train these ablations with the same initializations. The results present that different types of VGGNet [39] only slightly influence the performance. That is because all these pre-trained models can well represent multi-scale powerful features.

Table 3: Quantitative results of ablation study on dense MVSEC dataset.

FSM	MRM	Mean Depth Error [cm]			One Pixel Accuracy [%]		
		S1	S2	S3	S1	S2	S3
		19.99	29.94	25.30	76.75	48.68	70.57
✓		18.86	<u>27.73</u>	23.59	79.62	**52.88**	73.25
	✓	<u>18.59</u>	27.82	<u>23.09</u>	<u>79.89</u>	51.60	<u>73.92</u>
✓	✓	**17.99**	**26.90**	**22.54**	**81.13**	<u>52.23</u>	**75.44**

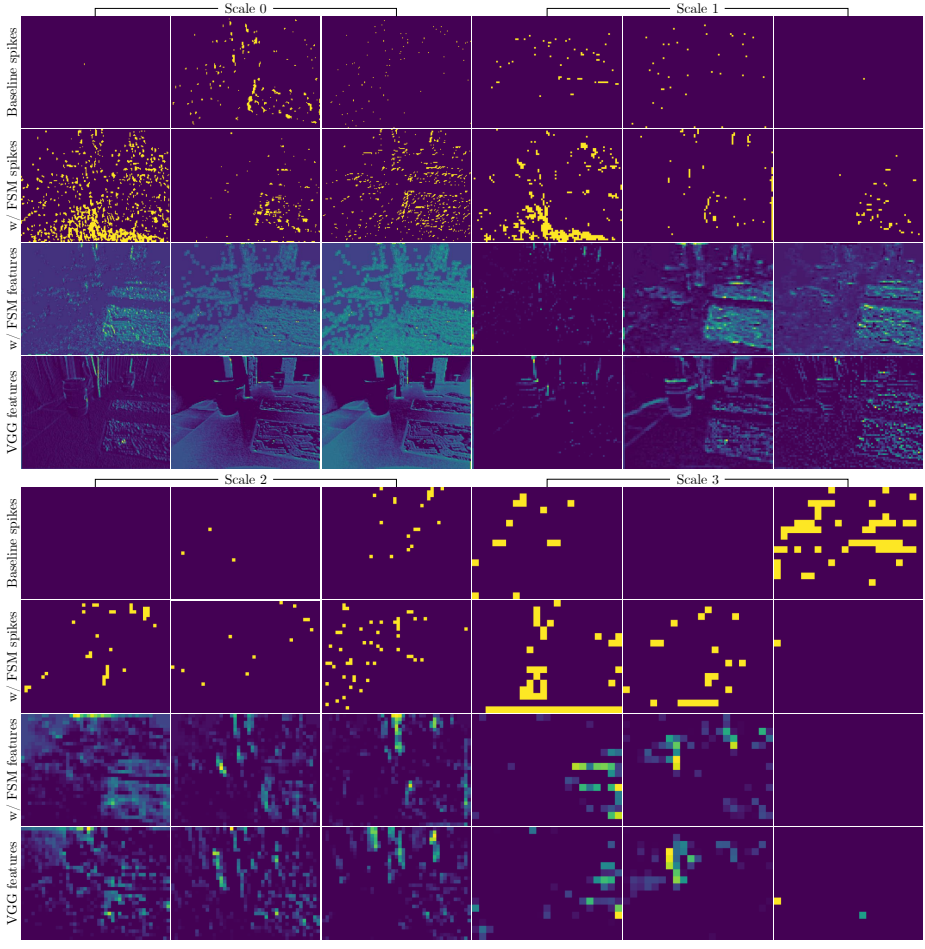

Fig. 5: Visualization of multi-scale features in network encoder of Split 1 #749.

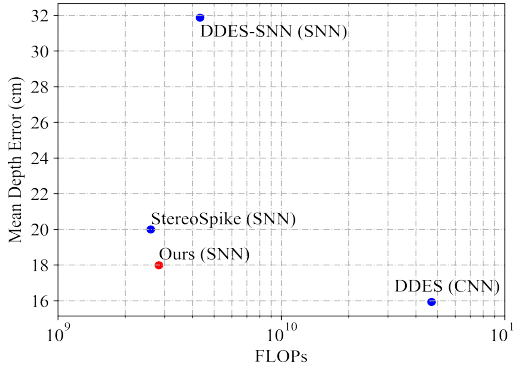

Fig. 6: Comparison in efficiency and performance of different methods.

Table 4: Average of spiking rate on Split 1 test set on dense MVSEC.

FSM	EPM	Encoder	Bottleneck	Decoder	MRM
		0.15	0.27	0.43	–
✓		0.19	0.32	0.47	–
	✓	0.13	0.25	0.39	0.21
✓	✓	0.16	0.28	0.41	0.20

Table 5: Ablation study of VGGNet type on dense MVSEC Split 1.

	Mean Depth Error [cm]	Mean Disp. Error [pix]	One Pixel Accuracy [%]
VGG11	17.99	0.76	81.13
VGG13	18.05	0.79	79.68
VGG16	18.08	0.77	81.36
VGG19	18.04	0.77	80.91

4.3 Computational Consumption

The main benefit of SNN is its passive response to spike stimulation, which means that it can significantly reduce the amount of calculation and energy consumption when deployed to neuromorphic hardware. As clarified in [21], the total number of operations for SNN in the l^{th} layer is $M_l \times C_l \times F_l \times N$, where M_l is the the number of neurons, C_l is the number of synaptic connections, F_l is the firing rate, and N is the number of time steps. The other difference between SNN and CNN is that SNN only requires accumulation (AC) while CNN requires multiply-accumulation (MAC). Fig. 6 shows the mean depth error and FLOPs in baseline CNN model DDES [40] and three SNN models. For CNN models, we generate a random input and calculate the MACs, while for SNN, we feed test data of Split 1 into the network and calculate the average ACs. We can find out that the FLOPs of our proposed method is 20× less than that of the DDES and 2× less than DDES-SNN, while still achieving competitive results. Besides, AC is 5.1× more energy-efficient than MAC, which means that our SNN will achieve much more energy saving on dedicated hardware.

5 Conclusions

In this paper, we propose a knowledge distillation mechanism based method to teach the SNN with a well trained CNN for dense depth estimation with stereo event cameras. The proposed SNN is composed of the encoder and the decoder part. For the encoder part, a feature supervision module is proposed to strengthen the expressiveness and generalization capability. And for the decoder part, a mask regularization module is proposed to realize elegant edge reconstruction performance at changing regions. We demonstrate the effectiveness and efficiency of the proposed method through comparing to state-of-the-art SNN methods, with competitive quantitative and qualitative results.

References

1. Ahmed, S.H., Jang, H.W., Uddin, S.N., Jung, Y.J.: Deep event stereo leveraged by event-to-image translation. In: Proceedings of the AAAI Conference on Artificial Intelligence. pp. 882–890 (2021)
2. Andreopoulos, A., Kashyap, H.J., Nayak, T.K., Amir, A., Flickner, M.D.: A low power, high throughput, fully event-based stereo system. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 7532–7542 (2018)
3. Barbier, T., Teuliere, C., Triesch, J.: Spike timing-based unsupervised learning of orientation, disparity, and motion representations in a spiking neural network. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshop. pp. 1377–1386 (2021)
4. Berner, R., Brandli, C., Yang, M., Liu, S.C., Delbruck, T.: A 240×180 10mw 12us latency sparse-output vision sensor for mobile applications. In: Proceedings of the Symposium on VLSI Circuits. pp. 186–187 (2013)
5. Bohte, S.M., Kok, J.N., La Poutre, H.: Error-backpropagation in temporally encoded networks of spiking neurons. Neurocomputing **48**(1-4), 17–37 (2002)
6. Brandli, C., Berner, R., Yang, M., Liu, S.C., Delbruck, T.: A 240×180 130 db 3 µs latency global shutter spatiotemporal vision sensor. IEEE Journal of Solid-State Circuits **49**(10), 2333–2341 (2014)
7. Cannici, M., Ciccone, M., Romanoni, A., Matteucci, M.: Asynchronous convolutional networks for object detection in neuromorphic cameras. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (2019)
8. Cao, Y., Chen, Y., Khosla, D.: Spiking deep convolutional neural networks for energy-efficient object recognition. International Journal of Computer Vision **113**(1), 54–66 (2015)
9. Caporale, N., Dan, Y.: Spike timing–dependent plasticity: a hebbian learning rule. Annual Review of Neuroscience **31**(1), 25–46 (2008)
10. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 248–255 (2009)
11. Dikov, G., Firouzi, M., Röhrbein, F., Conradt, J., Richter, C.: Spiking cooperative stereo-matching at 2 ms latency with neuromorphic hardware. In: Proceedings of the Conference on Biomimetic and Biohybrid Systems. pp. 119–137 (2017)
12. Fang, W., Chen, Y., Ding, J., Chen, D., Yu, Z., Zhou, H., Tian, Y., other contributors: Spikingjelly. `https://github.com/fangwei123456/spikingjelly` (2020), accessed: 2021-11-18
13. Gallego, G., Delbruck, T., Orchard, G.M., Bartolozzi, C., Taba, B., Censi, A., Leutenegger, S., Davison, A., Conradt, J., Daniilidis, K., et al.: Event-based vision: A survey. IEEE Transactions on Pattern Analysis and Machine Intelligence **44**(1), 154–180 (2020)
14. Gehrig, M., Aarents, W., Gehrig, D., Scaramuzza, D.: DSEC: A stereo event camera dataset for driving scenarios. IEEE Robotics and Automation Letters **6**(3), 4947–4954 (2021)
15. Haessig, G., Berthelon, X., Ieng, S.H., Benosman, R.: A spiking neural network model of depth from defocus for event-based neuromorphic vision. Scientific Reports **9**(1), 1–11 (2019)

16. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 770–778 (2016)
17. Hidalgo-Carrió, J., Gehrig, D., Scaramuzza, D.: Learning monocular dense depth from events. In: Proceedings of the International Conference on 3D Vision. pp. 534–542 (2020)
18. Huang, Z., Hu, X., Xue, Z., Xu, W., Yue, T.: Fast light-field disparity estimation with multi-disparity-scale cost aggregation. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 6320–6329 (2021)
19. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
20. Lanza, D.: The pytorch-hed python package. `https://github.com/Davidelanz/pytorch-hed` (2020)
21. Lee, C., Kosta, A.K., Zhu, A.Z., Chaney, K., Daniilidis, K., Roy, K.: Spike-flownet: event-based optical flow estimation with energy-efficient hybrid neural networks. In: Proceedings of the European Conference on Computer Vision. pp. 366–382 (2020)
22. Lee, C., Sarwar, S.S., Panda, P., Srinivasan, G., Roy, K.: Enabling spike-based backpropagation for training deep neural network architectures. Frontiers in Neuroscience **14**, 119 (2020)
23. Lichtsteiner, P., Posch, C., Delbruck, T.: A 128×128 120db 30mw asynchronous vision sensor that responds to relative intensity change. In: Proceedings of the IEEE International Solid State Circuits Conference-Digest of Technical Papers. pp. 2060–2069 (2006)
24. Luan, F., Paris, S., Shechtman, E., Bala, K.: Deep photo style transfer. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 4990–4998 (2017)
25. Ma, C., Rao, Y., Cheng, Y., Chen, C., Lu, J., Zhou, J.: Structure-preserving super resolution with gradient guidance. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 7769–7778 (2020)
26. Maqueda, A.I., Loquercio, A., Gallego, G., García, N., Scaramuzza, D.: Event-based vision meets deep learning on steering prediction for self-driving cars. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 5419–5427 (2018)
27. Marr, D., Poggio, T.: Cooperative computation of stereo disparity: A cooperative algorithm is derived for extracting disparity information from stereo image pairs. Science **194**(4262), 283–287 (1976)
28. Mostafavi, M., Yoon, K.J., Choi, J.: Event-intensity stereo: Estimating depth by the best of both worlds. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 4258–4267 (2021)
29. Neftci, E.O., Mostafa, H., Zenke, F.: Surrogate gradient learning in spiking neural networks: Bringing the power of gradient-based optimization to spiking neural networks. IEEE Signal Processing Magazine **36**(6), 51–63 (2019)
30. Niklaus, S.: A reimplementation of HED using PyTorch. `https://github.com/sniklaus/pytorch-hed` (2018)
31. Osswald, M., Ieng, S.H., Benosman, R., Indiveri, G.: A spiking neural network model of 3d perception for event-based neuromorphic stereo vision systems. Scientific Reports **7**(1), 1–12 (2017)
32. Pan, L., Liu, M., Hartley, R.: Single image optical flow estimation with an event camera. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 1669–1678 (2020)

33. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al.: Pytorch: An imperative style, high-performance deep learning library. arXiv preprint arXiv:1912.01703 (2019)
34. Rançon, U., Cuadrado-Anibarro, J., Cottereau, B.R., Masquelier, T.: Stereospike: Depth learning with a spiking neural network. arXiv preprint arXiv:2109.13751 (2021)
35. Rathi, N., Srinivasan, G., Panda, P., Roy, K.: Enabling deep spiking neural networks with hybrid conversion and spike timing dependent backpropagation. arXiv preprint arXiv:2005.01807 (2020)
36. Risi, N., Calabrese, E., Indiveri, G.: Instantaneous stereo depth estimation of real-world stimuli with a neuromorphic stereo-vision setup. In: Proceedings of the IEEE International Symposium on Circuits and Systems. pp. 1–5 (2021)
37. Rueckauer, B., Lungu, I.A., Hu, Y., Pfeiffer, M., Liu, S.C.: Conversion of continuous-valued deep networks to efficient event-driven networks for image classification. Frontiers in Neuroscience **11**, 682 (2017)
38. Shrestha, S.B., Orchard, G.: Slayer: spike layer error reassignment in time. In: Proceedings of the International Conference on Neural Information Processing Systems. pp. 1419–1428 (2018)
39. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014)
40. Tulyakov, S., Fleuret, F., Kiefel, M., Gehler, P., Hirsch, M.: Learning an event sequence embedding for dense event-based deep stereo. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 1527–1537 (2019)
41. Tulyakov, S., Ivanov, A., Fleuret, F.: Practical deep stereo (pds): Toward applications-friendly deep stereo matching. In: Proceedings of the International Conference on Neural Information Processing Systems. pp. 5871–5881 (2018)
42. Wu, Y., Deng, L., Li, G., Zhu, J., Shi, L.: Spatio-temporal backpropagation for training high-performance spiking neural networks. Frontiers in Neuroscience **12**, 331 (2018)
43. Xie, S., Tu, Z.: Holistically-nested edge detection. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 1395–1403 (2015)
44. Zhang, X., Liao, W., Yu, L., Yang, W., Xia, G.S.: Event-based synthetic aperture imaging with a hybrid network. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 14235–14244 (2021)
45. Zhou, Y., Gallego, G., Shen, S.: Event-based stereo visual odometry. IEEE Transactions on Robotics **37**(5), 1433–1450 (2021)
46. Zhu, A.Z., Chen, Y., Daniilidis, K.: Realtime time synchronized event-based stereo. In: Proceedings of the European Conference on Computer Vision. pp. 433–447 (2018)
47. Zhu, A.Z., Thakur, D., Özaslan, T., Pfrommer, B., Kumar, V., Daniilidis, K.: The multivehicle stereo event camera dataset: An event camera dataset for 3d perception. IEEE Robotics and Automation Letters **3**(3), 2032–2039 (2018)
48. Zhu, A.Z., Yuan, L., Chaney, K., Daniilidis, K.: Unsupervised event-based learning of optical flow, depth, and egomotion. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 989–997 (2019)
49. Zhu, M., Pan, P., Chen, W., Yang, Y.: Eemefn: Low-light image enhancement via edge-enhanced multi-exposure fusion network. In: Proceedings of the AAAI Conference on Artificial Intelligence. pp. 13106–13113 (2020)
50. Zuo, Y.F., Cui, L., Peng, X., Xu, Y., Gao, S., Wang, X., Kneip, L.: Accurate depth estimation from a hybrid event-rgb stereo setup. In: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems. pp. 6833–6840 (2021)