Learning to Robustly Reconstruct Dynamic Scenes from Low-light Spike Streams

Liwen Hu[1], Ziluo Ding[2], Mianzhi Liu[3], Lei Ma[1,3⋆], and Tiejun Huang[1]

[1] State Key Laboratory of Multimedia Information Processing, School of Computer Science, Peking University {huliwen, lei-ma, tjhuang}@pku.edu.cn
[2] Beijing Academy of Artificial Intelligence
{ziluoding}@baai.ac.cn
[3] College of Future Technology, Peking University
{liumianzhi}@stu.pku.edu.cn

A Appendix

A.1 Low-light Spike Streams Analysis

In low-light spike streams, the valid information can greatly decrease. It includes two reasons: (a) The information carried by each spike from the input signal is greatly reduced due to the interference of noise. (b) The total number of spikes in spike streams decrease greatly. Based on (1) in our main paper, we can get the valid accumulation in a spike. First, for the pixel $\mathbf{x}$, the time to fire a spike, $t_{\mathbf{x}}$, can be written as,

$$t_{\mathbf{x}} = A_{\mathbf{x}}^{-1}(\phi). \tag{1}$$

Note that $A_{\mathbf{x}}^{-1}(\cdot)$ exists because $A_{\mathbf{x}}(\cdot)$ is monotonically increasing. Especially, when the light intensity is fixed, *i.e.* $I_{in}(\mathbf{x}, \tau) = I$, (11) can be written as,

$$t_{\mathbf{x}} = \phi(I + I_{dark}(\mathbf{x}))^{-1}. \tag{2}$$

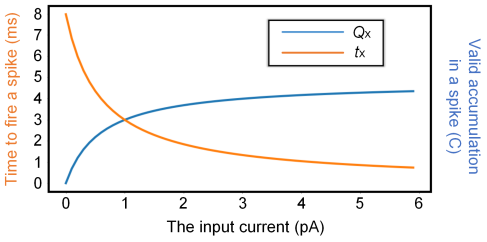

Fig. 1: Influence of input current on spikes. In low-light environment, *i.e.* input current is low, the time to fire a spike is long and the valid accumulation in each spike is small.

⋆ Corresponding author.

Further, we can get the valid accumulation from input current I in a spike, $Q_{\mathbf{x}}(I)$, as,

$$Q_{\mathbf{x}}(I) = It_{\mathbf{x}} = I\phi(I + I_{dark}(\mathbf{x}))^{-1}. \tag{3}$$

The orange curve in Fig. 1 shows that the valid accumulation $Q_{\mathbf{x}}(I)$ increases with increasing input current I which means each spike in low-light environments is more difficult to record information. The blue curve in Fig. 1 shows that time to fire spikes $t_{\mathbf{x}}$ decreases with increasing input current I which means the total information *i.e.* the amount of spikes, in low-light spike stream is sparse. The above two characteristics explains the sparsity of information in low-light spike streams.

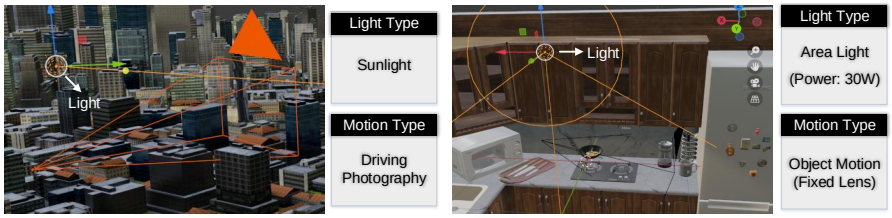

Fig. 2: The Car_{N} (left) and Cook_{L} (right) in LLR. N (L) means normal (low) light.

A.2 Datasets Details

Scene LLR serves as the test set and is designed to be as consistent as possible with the real world in order to effectively evaluate different methods. To achieve this, as shown in Fig. 2, we have carefully designed the light source type and the illumination power for each scene to match the real world. Besides, motion of objects is close to the real world. The motion in Ball, Cook, Fan and Rotate is from [4] while the motion in Car is created based on vehicle speed in real world.
Light source set We set the lighting parameters in the advanced 3D graphics software, blender, to make the lighting conditions as consistent as possible with the real world. The following are the configuration details in Blender. In Blender, various types of lighting simulation functions, including sunlight, point lights, and area lights, have been integrated into the graphical interface. We can adjust lighting parameters to control brightness and darkness. For sunlight in Blender, the watts per square meter can be modified. Typically, 100 watts per square meter corresponds to a cloudy lighting environment. For the Car_L scene, we have set sunlight to 10 watts per square meter, which is deemed sufficiently low. For point lights and area lights, Blender allows modification of radiant Power, measured in watts. This is not the electrical power of consumer light bulbs. A light tube with the electrical power of 30W approximately corresponds to a radiant power of 1W.

In the Cook$_L$ scene, we have set an area light with the radiant Power to 1W (the electrical power of 30W) . It already represents a very dim indoor light source.

Fig. 3: (a) Grayscale histograms of images in low-light scenes, *i.e.* Ball$_L$, Car$_L$, Cook$_L$, Fan$_L$ and Rotate$_L$ with low-light light sources. Each bar represents 5 grayscale levels. (b) Reference images.

Grayscale The brightness is not only determined by the light source, but also by factors such as camera distance, object occlusion, and so on. These factors are ultimately reflected in the grayscale of the rendered images. Therefore, we calculate the grayscale histograms of images in low-light scenes. As shown in Fig. 3, we can see that the grayscale is diverse and in a lower range.

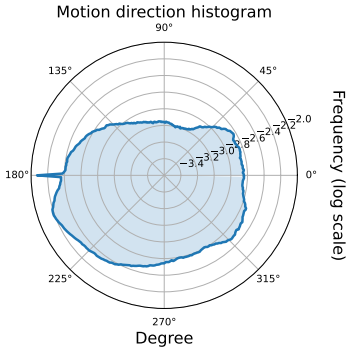

Fig. 4: Motion direction histogram of optical flow in LLR.

Motion The motion in LLR is diverse. We generate a optical flow every 40 frames for LLR. The degree distribution of the optical flow is in Fig. 4. We can find that the motion in LLR covers all kinds of directions.

Impact of Scene Brightness on Performance In fact, the two types of light sources (normal-light and low-light) are sufficient to demonstrate that our method can robustly handle spike streams under different lighting conditions. This is because the distribution of light intensity in scene is diverse (see Fig. 3). To further validate our point of view, based on the scene Car, we modify 6 the

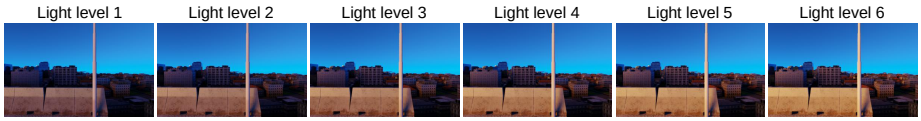

Fig. 5: The scene Car with 6 different light source parameters. From light level 1 to light level 6, brightness is from small to large.

Table 1: PSNR and SSIM of reconstruction results under different light sources. We set sunlight in the scene Car to 30, 50, 70, 90, 110 and 130 watts per square meter to render images respectively.

Light level	30W	50W	70W	90W	110W	130W	Avg.
WGSE(PSNR)	37.12	35.68	34.45	33.88	33.61	33.36	34.68
Ours(PSNR)	**41.52**	**40.69**	**39.99**	**39.51**	**39.15**	**38.91**	**39.96**
WGSE(SSIM)	0.9514	0.9418	0.9324	0.9306	0.9305	0.9307	0.9362
Ours(SSIM)	**0.9772**	**0.9748**	**0.973**	**0.9711**	**0.9701**	**0.9693**	**0.9725**

different light source parameters as shown in Fig. 5. All results are shown in Table 1.

Table 2: Reconstruction results on synthetic dataset, LLR. Retrain$_{idea}$: our method is retrained on the noise-free version of RLLR.

Metric	Our	Retrain$_{idea}$
PSNR	**45.075**	37.679
SSIM	**0.98681**	0.85374

Impact of Spike Camera Noise on Performance In proposed datasets, we have considered noise of spike camera refer to [9]. We further discuss the impact of noisy and noise-free spike streams on the performance of our method as shown in Table. 2. We use an ideal spike camera model in SPCS [4] to synthesize a noise-free version of RLLR and retrain our method using the dataset (written as Retrain$_{idea}$). We can find that our method has better performance than Retrain$_{idea}$. Besides, Fig. 6 shows our method can handle noise in real spike streams better than Retrain$_{idea}$.

Explanation of High PSNR When the scene is so dark, both the ground truth (GT) and the reconstructed images can be at a lower grayscale level. This results in a significantly lower MSE compared to normal-light scenes. Consequently, their PSNR is high. Specifically, the Peak Signal-to-Noise Ratio (PSNR) can be

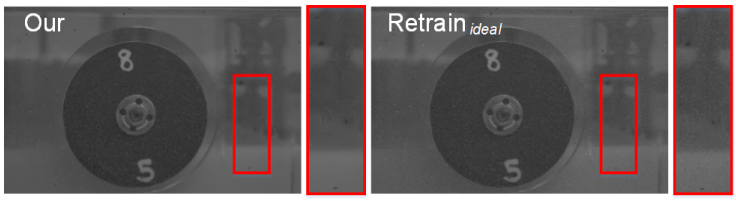

Fig. 6: Reconstruction results on a real spike stream. Please enlarge the figure for more details.

expressed as,

$$PSNR(R, GT) = 10 \cdot \log_{10}\left(\frac{255^2}{MSE(R, GT)}\right), \tag{4}$$

where R (GT) are reconstructed images (ground truth) and $MSE(\cdot, \cdot)$ represents the Mean Squared Error. It can be further defined as, $MSE(R, GT) = \frac{1}{mn}\sum_{i=0}^{m-1}\sum_{j=0}^{n-1}\left(R(i,j) - GT(i,j)\right)^2$,where m (n) is the height (width) of the image, respectively and $R(i,j)(GT(i,j))$ represents the pixel value at (x, y). Hence, when both the Ground Truth (GT) and the reconstructed images are at a lower grayscale level, MSE is lower (PSNR is higher).

A.3 LR-Rep Details

GISI Transform In LR-Rep, we first use the GISI transform to get the global inter-spike interval, $\mathbf{GISI}_{t_i}$, from the input spike stream and the release time of forward and backward spikes. The GISI transform can be summarized as three steps (see Fig. 5 in the main paper): (a). Calculate the local inter-spike interval from input spike stream as [2, 10] and we call it LISI transform for simplicity. (b). Update the local inter-spike interval as global inter-spike interval based on the release time of forward and backward spikes. (c). Maintain the release time of forward (backward) spikes of backward (forward) spike streams. Related details are shown in Algorithm.1. As shown in Fig. 7, GISI (our final method) not only outperforms LISI (Baseline (F) in Table 2 of the main paper) in both PSNR and SSIM on LLR but also have better generalization on real data. More importantly, the cost of using GISI instead of LISI is negligible (we only need to use two 400×250 matrices to store the time of the forward spike and the backward spike, respectively), which does not affect the parameter and efficiency of the network.

A.4 Experiment

Model efficiency Table. 3 demonstrates the training time and inference time of the supervised methods, i.e., Spk2ImgNet, WGSE and our method. Although our method requires more training time compared to Spk2ImgNet and WGSE (Recurrent-based networks typically consume more time during training due to

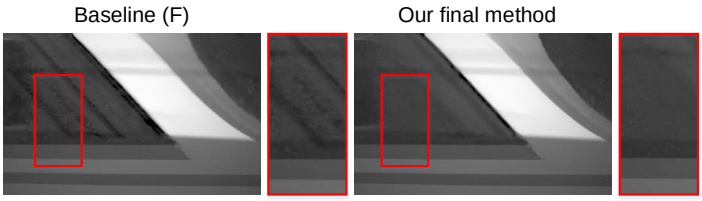

Fig. 7: Reconstruction results on a real spike stream. The scene is a high-speed train that exceeds 200 km/h. The glass in the former result (left) shows obvious artifacts, and our result (right) is very smooth and natural.

Table 3: Comparison between Spk2ImgNet (S2I), WGSE and our method. The input spike stream size is $21 \times 41 \times 250 \times 400$. We test the average of 50 rounds for Inference time.

Method	Para.	Train time	Inference time
S2I	3.91m	2h	1458.45ms
WGSE	**3.63m**	**1h**	1344.06ms
Our	5.32m	17h	**818.03ms**

Fig. 8: The reconstructed results on the real dataset [8].

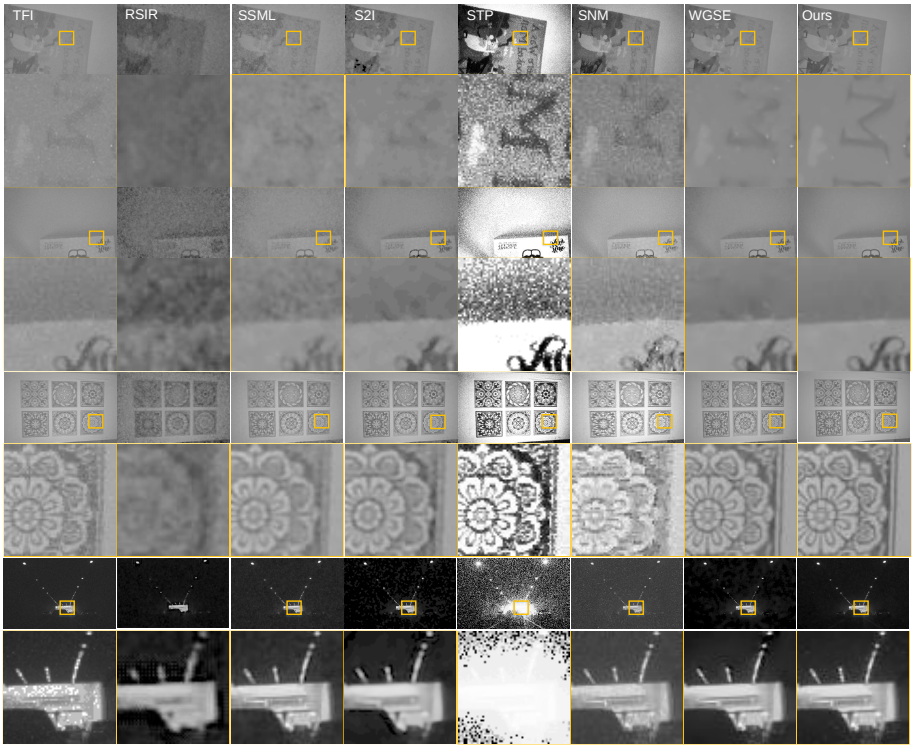

Fig. 9: The reconstructed results on the real dataset [3].

Algorithm 1 GISI Transform.

Require: The spike streams at different time $\{S_{t_i} \mid i = 1, 2, \ldots, K\}$, K is the number of Continuous spike streams.

Initialize forward state $\text{Spike}^f_{t_1} = 0$.
Initialize backward state $\text{Spike}^b_{t_K} = 2K\Delta t$.
for i from 1 to K **do**
  Calculate LISI_{t_i} based on S_{t_i}.
end for
for i from 2 to K **do**
  Forward search the recent release time of spike to t_{i+1}, $\text{Spike}^f_{t_i}$ based on S_{t_i}.
  if $\text{Spike}^f_{t_i}$ is None **then**
    Set $\text{Spike}^f_{t_i} = \text{Spike}^f_{t_{i-1}}$.
  end if
  Update GISI_{t_i} based on S_{t_i} and $\text{Spike}^f_{t_{i-1}}$.
end for
for i from $K-1$ to 1 **do**
  Backward search the recent release time of spike to t_{i-1}, $\text{Spike}^b_{t_i}$ based on S_{t_i}.
  if $\text{Spike}^b_{t_i}$ is None **then**
    Set $\text{Spike}^b_{t_i} = \text{Spike}^b_{t_{i+1}}$.
  end if
  Update GISI_{t_i} based on S_{t_i} and $\text{Spike}^b_{t_{i+1}}$.
end for
Return $\{\text{GISI}_{t_i} \mid i = 1, 2, \ldots, K\}$

Backpropagation Through Time (BPTT)), our method outperforms Spk2ImgNet and WGSE in terms of inference speed. Besides, due to the need to fuse both forward and backward temporal features, our method is offline, i.e., After spike camera collects spike stream for a long period of time, the data can be reconstructed. In future work, we would extend our method so to online reconstruct.

Real data Here, we show more results on two real datasets. Fig. 8 and Fig. 9 show more reconstructed images. We find that for traditional methods, TFI performs better on low-light data than TFP, SNM and TFSTP. For deep learning-based methods, SSML introduces a large amount of motion blur while Spk2ImgNet and WGSE may introduces some loss in dark backgrounds. Our method restores texture details in low-light scenes clearly more than other methods.

Synthetic data Here, we show more results on synthetic dataset LLR. Fig. 10 shows more reconstruction results on proposed dataset LLR. We find that for traditional methods, TFI performs better on low-light data than TFP, SNM and TFSTP. For deep learning-based methods, SSML introduces a large amount of motion blur while Spk2ImgNet and WGSE may introduces some loss in dark backgrounds. Our method restores texture details in low-light scenes clearly more than other methods.

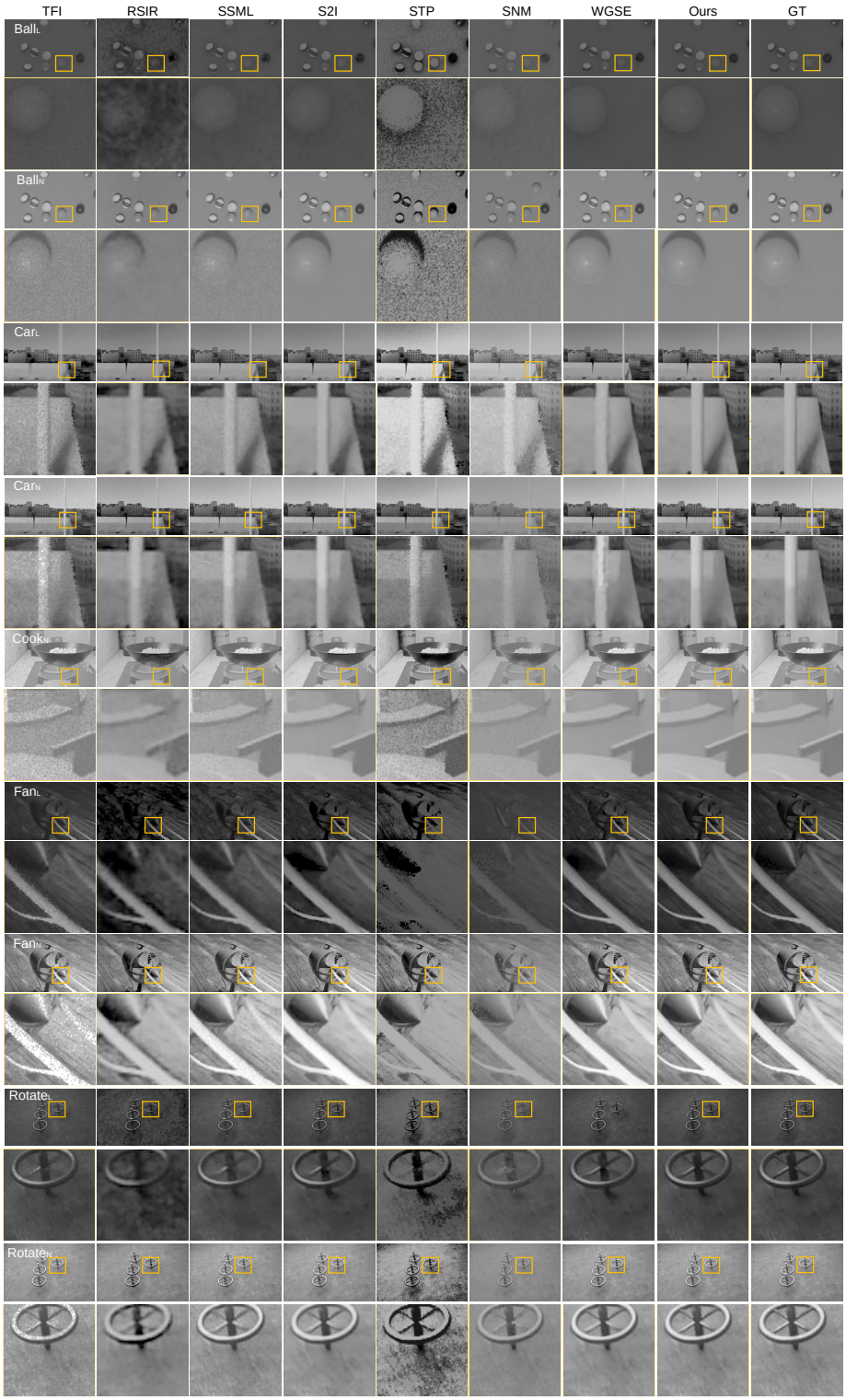

Fig. 10: The reconstructed results on LLR. N (L) means normal (low) light.

Table 4: Reconstruction results on a synthetic dataset, LLR. We compare the open-source Single Photon Avalanche Diode method, 3DCNN [1] (ICCP 2019) which is retrained on RLLR.

Method	PSNR	SSIM
Our	**45.075**	**0.98681**
3DCNN	34.507	0.93506

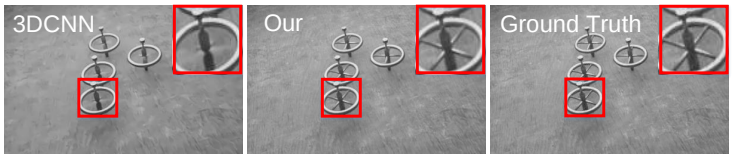

Fig. 11: The reconstructed results of Rotate_{N}. Our method has clearer images.

Comparison of Quanta Image Sensor We would like to discuss Quanta Image Sensors (QIS). Spike camera [11] and QIS [5] (including CIS-QIS and SPAD-QIS) share some similar characteristics, such as high temporal resolution and 1-bit (0 or 1) data. Besides, they also have differences in principles and circuits. For one sampling (one frame), QIS records whether a photon has arrived during the sampling, with a corresponding pixel output of 1 if photons arrive, and 0 otherwise [6]. Different from QIS, spike camera continuously accumulates photons [11], and if the accumulated value reaches a fixed threshold, the pixel outputs 1 and the accumulation is reset. Otherwise, it outputs 0, and the accumulation value. The different principles result in distinct meanings of two data (QIS data and spike streams). In QIS, 1 reflects the information of a specific sampling. In contrast, in spike camera, 1 contains the information from previous multiple sampling, and adjacent spikes are interdependent. This also leads to differences in the data patterns. This characteristic brings both advantages and disadvantages. In terms of advantages, in spike cameras, the influence of photon shot noise on each spike is reduced as multiple samples of photons are dynamically accumulated together, while QIS is sensitive to poisson shot noise [5]. In terms of disadvantages, spike cameras face more challenges in low-light conditions due to difficulties in reaching the accumulation threshold (see limitation in [10]). Furthermore, the pixel circuits of two cameras are also different. A spike camera continuously accumulates photons in the form of voltage and the voltage can be kept for next sampling. QIS cameras (using SPAD-QIS [7] as an example) amplify the signal through the avalanche multiplication mechanism to detect the presence or absence of individual photons. Besides, we test 3DCNN [1] (a reconstruction method for QIS). To ensure fairness, we retrain 3DCNN using RLLR with spike streams as inputs. Table. 4 demonstrates the reconstruction evaluation on LLR. As shown in Fig. 11, our method removes motion blur better.

References

1. Chandramouli, P., Burri, S., Bruschini, C., Charbon, E., Kolb, A.: A bit too much? high speed imaging from sparse photon counts. In: 2019 IEEE International Conference on Computational Photography (ICCP). pp. 1–9. IEEE (2019)
2. chen, S., Duan, C., Yu, Z., Xiong, R., Huang, T.: Self-supervised mutual learning for dynamic scene reconstruction of spiking camera. In: Proceedings of the International Joint Conference on Artificial Intelligence, IJCAI. pp. 2859–2866 (2022)
3. Dong, Y., Zhao, J., Xiong, R., Huang, T.: High-speed scene reconstruction from low-light spike streams. In: 2022 IEEE International Conference on Visual Communications and Image Processing (VCIP). pp. 1–5. IEEE (2022)
4. Hu, L., Zhao, R., Ding, Z., Ma, L., Shi, B., Xiong, R., Huang, T.: Optical flow estimation for spiking camera. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 17844–17853 (2022)
5. Ma, J., Chan, S., Fossum, E.R.: Review of quanta image sensors for ultralow-light imaging. IEEE Transactions on Electron Devices **69**(6), 2824–2839 (2022)
6. Ma, S., Gupta, S., Ulku, A.C., Bruschini, C., Charbon, E., Gupta, M.: Quanta burst photography. ACM Transactions on Graphics (TOG) **39**(4), 79–1 (2020)
7. Qian, X., Jiang, W., Elsharabasy, A., Deen, M.J.: Modeling for single-photon avalanche diodes: State-of-the-art and research challenges. Sensors **23**(7), 3412 (2023)
8. Zhao, J., Xiong, R., Liu, H., Zhang, J., Huang, T.: Spk2imgnet: Learning to reconstruct dynamic scene from continuous spike stream. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 11996–12005 (2021)
9. Zhao, J., Zhang, S., Ma, L., Yu, Z., Huang, T.: Spikingsim: A bio-inspired spiking simulator. In: 2022 IEEE International Symposium on Circuits and Systems (ISCAS). pp. 3003–3007. IEEE (2022)
10. Zhao, R., Xiong, R., Zhao, J., Yu, Z., Fan, X., Huang, T.: Learninng optical flow from continuous spike streams. In: Proceedings of the Annual Conference on Neural Information Processing Systems (NeurIPS) (2022)
11. Zhu, L., Dong, S., Huang, T., Tian, Y.: A retina-inspired sampling method for visual texture reconstruction. In: IEEE International Conference on Multimedia and Expo (ICME). pp. 1432–1437 (2019)