Self-Supervised Learning for Color Spike Camera Reconstruction

Supplementary Material

1. Color Spike Camera Details

1.1. Capture Process

To introduce the working mechanism of CSCs better, we give a brief and rough formulation description of the capture process. During the capture of the dynamic scene, the CSC sensor can record the instantaneous absolute light intensity, which can be modeled as continuous latent light intensity frames $\{\mathbf{F}_1, \mathbf{F}_2, ..., \mathbf{F}_n\}$. Due to multiple noises in the capture process, there are fluctuations for each frame. As a result, the process can be formulated as follows:

$$\{\mathbf{S}_i\}_{i=1}^{n'} = \mathcal{C}(\{\mathbf{F}_i + \mathbf{n}_i\}_{i=1}^{n}, \theta), \tag{1}$$

where $\{\mathbf{S}_i\}_{i=1}^{n'}$ denotes the output Bayer-pattern spike stream with n' binary spike frames, $\mathcal{C}(\cdot)$ denotes the CSC, $\mathbf{n}_i$ denotes the noise of the i-th latent frame, and θ denotes the spike firing threshold.

1.2. Hardware Details

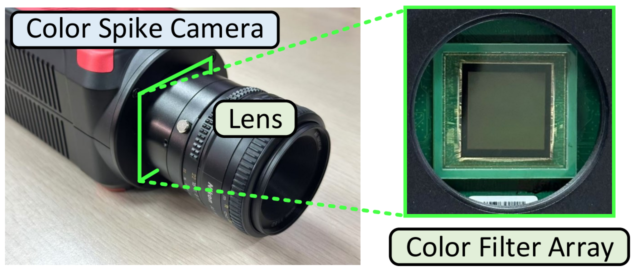


Figure 1. Illustration of color spike camera.

The CSC used to capture real-world Bayer-pattern spike streams of BSS [5] employs an RGGB Bayer-pattern color filter array (see Fig. 1), which is facilitated with a 50mm 1:1.8D lens. According to the sensor manufacturer, a spatial resolution of 1000×1000 is achieved, with a pixel size of $17\mu m \times 17\mu m$. The frequency of its sensor is 20000Hz. The data output speed is 500MHz, while the firing threshold voltage stands at 0.9V, with a reset time of $200ns$.

1.3. CSC Simulator

In our work, we employ a CSC simulator to generate datasets and analyze the quantization noise, which is developed according to the working mechanism of CSC. Fig. 2 presents the pipeline of the simulator. The input of the simulator is color video frames, which are regarded as the dynamic scene to be captured. Considering the frame rate of the video is often limited, the temporal information is not enough for the generation of spike streams with ultra-high temporal resolution. Thus, we use a frame interpolation method [11] to approximate continuous light intensity frames. Then, we convert the three-channel intensity frame to a one-channel Bayer-pattern intensity frame according to the color mask, designed to approximate the CFA on the sensor. After that, we simulate the processes of *accumulate-and fire* and *check-and-reset* based on the working mechanism of spike cameras, resulting in a Bayer-pattern spike stream. To better introduce the simulator, we summarize the pipeline in Algorithm 1. The codes of the CSC simulator will be publicly available.

Algorithm 1: Color spike camera simulator

```
Input: Color video frames, spike firing threshold $\theta$ and pattern of the CFA
Output: A clip of Bayer-pattern spike stream $\{\mathbf{S}_i\}_{i=1}^{N}$
1  Generate $N$ intensity frames denoted as $\{\mathbf{I}_i\}_{i=1}^{N}$ from the input frames by frame interpolation;
2  for $i \leftarrow 1$ to $N$ do
3      Convert the three-channel intensity frame to a one-channel Bayer-pattern intensity frame $\mathbf{I}_i^B$ according to the pattern of CFA;
4      Accumulate light intensity signals from the Bayer-pattern intensity frame, resulting in the accumulated signals $\mathbf{E}_i$;
5      for each pixel $(x, y)$ do
6          if $\mathbf{E}_i(x, y) \geq \theta$ then
7              $\mathbf{S}_i(x, y) = 1$; // Spike fired
8              $\mathbf{E}_i(x, y) = 0$; // Reset
9          else
10             $\mathbf{S}_i(x, y) = 0$; // No spike fired
11         end
12     end
13 end
```

2. Method Details

2.1. Method Summary

We summarize the pipeline of the motion-guided reconstruction method in Algorithm 2 for a better description.

2.2. Motion Estimation

In our method, we need the optical flows from the middle time point to the beginning and the end time points, $\mathbf{w}_{j,i}^c$ and $\mathbf{w}_{j,k}^c$. According to the usage of the spike camera motion

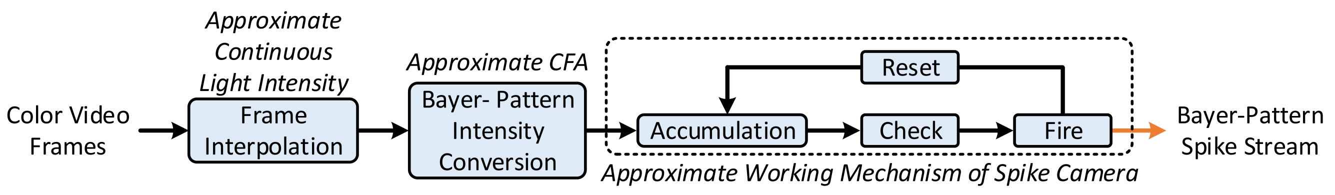

Color Video Frames
Approximate Continuous Light Intensity
Frame Interpolation
Approximate CFA
Bayer- Pattern Intensity Conversion
Reset
Accumulation
Check
Fire
Approximate Working Mechanism of Spike Camera
Bayer-Pattern Spike Stream

Figure 2. The pipeline of the color spike camera simulator.

Algorithm 2: Motion-Guided Reconstruction

```
Input: A clip of Bayer-pattern spike stream
       $\{\mathbf{S}_i\}_{i=1}^N$, color mask
Output: A color image $\mathbf{T}$ from the Bayer-pattern
        spike stream
1  Extract spike signals of each color channel by
   element-wise multiplication of each spike frame
   and the color mask of each color channel, resulting
   in $\{\mathbf{S}_i^R\}_{i=1}^N$, $\{\mathbf{S}_i^G\}_{i=1}^N$ and $\{\mathbf{S}_i^B\}_{i=1}^N$;
2  for each color channel $c \in \{R, G, B\}$ do
3      Transform the spike signals $\{\mathbf{S}_i^c\}_{i=1}^N$ to
       intervals;
4      Estimate the values of the missing pixel
       positions of the intervals by spatial
       interpolation;
5      Estimate optical flows from the middle time
       point to the beginning and end time points by
       the spike camera motion estimation method
       Spike2Flow [14], resulting in $\mathbf{w}_{j,i}^c$ and $\mathbf{w}_{j,k}^c$;
6  end
7  Get joint estimation results $\mathbf{w}_{j,i}$ and $\mathbf{w}_{j,k}$ by
   averaging the optical flows of color channels;
8  for each color channel $c \in \{R, G, B\}$ do
9      Infer the light intensity $\{\mathbf{L}_i^c\}_{i=1}^N$ from $\{\mathbf{S}_i^c\}_{i=1}^N$
       by TFI [15];
10     Fill the missing pixel positions of $\{\mathbf{L}_i^c\}_{i=1}^N$ by
       temporally neighboring pixels along the
       motion trajectory (according to $\mathbf{w}_{j,i}$ and $\mathbf{w}_{j,k}$);
11     Estimate the positions without available
       temporal pixels by spatial interpolation;
12 end
13 Concentrate the three filled color channels, resulting
   in the target color image $\mathbf{T}$.
```

estimation method Spike2Flow [14], the number of spike frames for each time point is 21. The interval between time points should be a multiple of 3. The right boundary of the last time point farthest from the middle time point is $21 + 3 \times 3 + 10 = 40 < N = 41$. Similarly, for the left boundary of the first time point farthest from the middle time point, the calculation yields $21 - 3 \times 3 - 10 = 2 > 1$. Consequently, the indices corresponding to the last time point and the first time point are determined as 30 and 12, respectively, as shown in Fig. 3.

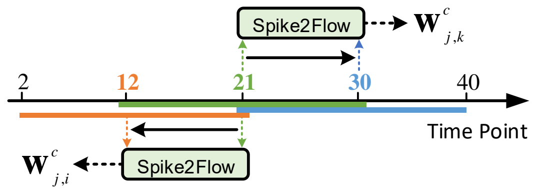


Figure 3. Motion estimation by Spike2Flow [14], $c \in \{R, G, B\}$.

2.3. Assumption of Smooth Motion

In Eqn. (6), we assume that the motion is smooth between the middle time point and the first/last time point. The assumption is a trade-off for computational complexity. As the frequency of the CSC is 20,000Hz, the underlying motion within temporal neighboring frames is slight in many cases. The time interval of two time points for motion estimation is 20/20000s = 1 ms. Thus, we assume the motion is linear during the period for a lower computation cost. For better performance with no consideration of the cost, we can estimate motion between each time point and the middle time point for the cases with complex motion.

2.4. No-Temporal-Pixel Position Filling

For the positions of missing pixels without available temporally neighboring pixels, we propose to estimate the values by spatial interpolation. Supposing that the positions of the four closest non-missing pixels to the missing pixel position (x, y) are (x_1, y_1), (x_1, y_2), (x_2, y_1) and (x_2, y_2) according to the Bayer pattern color layout, we can estimate the value of the position as follows:

$$\begin{aligned}\mathbf{I}(x,y) = \mathbf{I}(x_1,y_1)\cdot\frac{(x_2-x)(y_2-y)}{(x_2-x_1)(y_2-y_1)}\\ +\mathbf{I}(x_2,y_1)\cdot\frac{(x-x_1)(y_2-y)}{(x_2-x_1)(y_2-y_1)}\\ +\mathbf{I}(x_1,y_2)\cdot\frac{(x_2-x)(y-y_1)}{(x_2-x_1)(y_2-y_1)}\\ +\mathbf{I}(x_2,y_2)\cdot\frac{(x-x_1)(y-y_1)}{(x_2-x_1)(y_2-y_1)}.\end{aligned} \tag{2}$$

Dataset	Generation	Base Dataset	Samples	Resolution	Intensity Coeff.	Main Motion Source
RBSS	Simulation	REDS [9]	150	$41 \times 720 \times 1280$	$\{0.6, 0.8, 1.0\}$	Camera's ego motion
DBSS	Simulation	DAVIS [10]	210	$41 \times 720 \times 1280$	$\{0.6, 0.8, 1.0\}$	Motion of captured objects
GBSS	Simulation	GoPro [8]	165	$41 \times 720 \times 1280$	[0.6, 1.0]	Camera's ego motion
BSS [5]	Shoot	-	28	$399 \times 1000 \times 1000$	-	Camera or object motion

Table 1. Details of the Bayer-pattern spike stream datasets for evaluation.

2.5. Discussions on Pseudo-Labels

To generate the pseudo-labels, we collect temporally neighboring pixels along motion trajectories to estimate the spatially missing pixels caused by the color filter array. However, the estimated motion trajectories are not entirely accurate in most cases. There are often some missing pixels without available temporally neighboring pixels for estimation. Though surpassing the SOTA learning-free method 3DRI [3], the quality of the generated pseudo-labels is still imperfect, with some noise and artifacts. Supervised by lots of imperfect pseudo-labels, our self-supervised method can learn to generate clean color images from the input Bayer-pattern spike streams.

2.6. Conv and Residual Block

As shown in Fig. 4, Conv in our figures denotes a 3×3 2D convolutional layer followed by an LReLU activation function. Residual Block refers to the residual block proposed in [7] with the LReLU activation function.

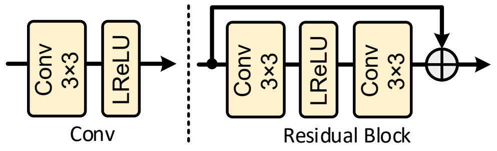


Figure 4. Illustration of Conv and Residual Block.

2.7. Loss Function

Following some network-based works of spike camera reconstruction [12, 13], we use ℓ_1 loss as the loss function. Given m pairs of training data with N spike frames in each sample, the optimization objective can be written by

$$\hat{\Theta} = \arg\min_{\Theta} \frac{1}{m} \sum_{i=1}^{m} \|\mathcal{F}(\{\mathbf{S}_j^i\}_{j=1}^{N}; \mathbf{M}_i; \Theta) - \mathcal{M}(\{\mathbf{S}_j^i\}_{j=1}^{N}; \mathbf{M}_i)\|_1, \tag{3}$$

where Θ denotes the parameters of our model, $\mathcal{F}(\cdot)$ denotes our self-supervised network, $\mathbf{S}_j^i$ denotes the j-th Bayer-pattern spike frame of the i-th sample, $\mathbf{M}_i$ denotes the i-th color mask, and $\mathcal{M}(\cdot)$ denotes our motion-guided reconstruction method for CSCs.

2.8. Limitations

As the sensor frequency of the CSC is 20,000Hz, the inference time is supposed to be less than 0.05ms for a 1000×1000 color image. However, our method is not fast enough to meet the demands of real-time inference. We will study a real-time method in further research.

3. Experimental Details

3.1. Difficulty for Capturing Paired Data

The ideal way of collecting data pairs is using an RGB camera and a CSC to shoot a dynamic scene at the same time. However, this strategy meets the following challenges:

- Compared with the CSC, the temporal resolution of RGB cameras is limited. The output RGB image as ground truth is likely to be blurry for dynamic scenes with high-speed motion. Though employing a high-speed RGB camera, the temporal resolution is also hard to meet at 20,000Hz. Besides, the spatial resolution will be impacted when using higher temporal resolution limited by the bandwidth of the high-speed RGB camera.
- To capture synchronized data, we usually use a beam splitter. However, the beam splitter will halve the incoming light for each camera, resulting in a significant impact on the high-frequency sensors (*i.e.*, CSC and high-speed RGB camera).
- Due to the spatial and temporal resolution difference between the RGB camera and the CSC, the spatial and temporal alignment between RGB ground truth images and Bayer-pattern spike streams is also a challenge, especially for a large number of training pairs.

As a result, it's hard to capture datasets with lots of high-quality pairs. To handle the problem, most existing learning-based methods try to employ spike camera simulators, which generate spike streams from video frames. However, the frame rates of most video datasets cannot meet 20,000Hz. The simulated data pairs based on frame interpolation for higher frame rates are unreal enough. The sensor noise is also hard to simulate. As a result, there are gaps between real-world captured pairs and synthetic pairs.

Due to the difficulty of capturing paired data, we employ synthetic datasets for training and quantitative evaluation. In particular, we use BSS [5] to verify the generalizability of the model trained on the synthetic spike streams to real-world captured Bayer-pattern spike streams.

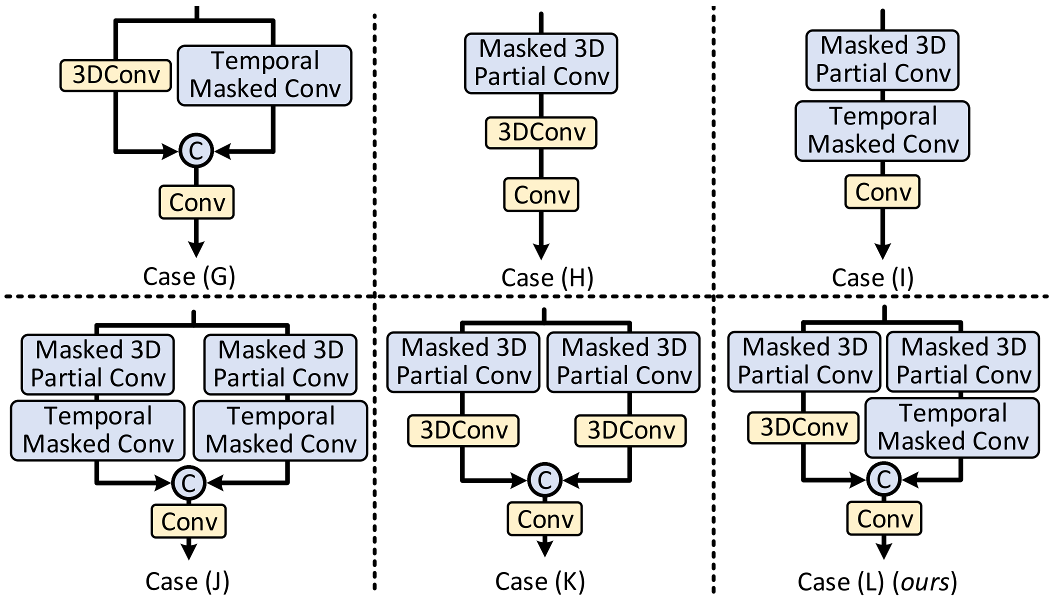


Figure 5. Illustration of ablation study cases for the MSE module.

3.2. Evaluation Dataset Details

As mentioned in the pipeline of the CSC simulator, we employ color frames from video datasets to simulate the dynamic scene. For the SOTA supervised method SJDD-Net [5] and CSpkNet [6], REDS and DAVIS are used for data simulation, which gives us inspiration for video data selection. To fully evaluate the method, we propose to use the video datasets with multiple motion sources as shown in Table 1. Therefore, we select three video datasets (REDS [9], DAVIS [10] and GoPro [8]) for data simulation, which are widely used in video restoration and spike camera reconstruction. Other video datasets with enough color frames are also available for Bayer-pattern spike stream generation.

However, the frame rates of most video datasets are limited. To be specific, the temporal information is not enough for the simulation of spike streams with ultra-high temporal resolution. Thus, we pre-process the data by frame interpolation to approximate continuous light intensity frames. More details of evaluation datasets with Bayer-pattern spike streams are shown in Table 1.

3.3. Light Inference Strategies

TFI [15] and TFP [15] are two basic pixel-level strategies for light inference from spike streams, which are employed in both comparison experiments and ablation studies of our work. The former is based on the interval of neighboring spikes, the latter is based on the number of spike signals within a temporal window. For the first interval-based strategy, we can infer the light intensity of a certain pixel (x, y) based on the reciprocal of the interval as

$$\mathbf{L}_n(x,y) = \frac{\delta}{\mathbf{\Psi}_n(x,y)}, \tag{4}$$

where δ denotes the maximum dynamic range, and $\mathbf{\Psi}_n(x,y)$ denotes the shortest spike interval of two temporally neighboring spikes that covers the n-th spike frame. For the window-based method TFP, the light intensity can be inferred as follows:

$$\mathbf{L}_n(x,y) = \frac{\delta}{w}\sum_{i=n+1}^{n+w} \mathbf{S}_{i-\frac{w+1}{2}}(x,y), \tag{5}$$

where w denotes the length of the temporal window, which is usually an odd number.

3.4. Quantitative Gaps between SL and SS Methods

The final target of CSC reconstruction is to restore high-quality color images from real-world captured spike streams. In low-level vision works, SS methods are often inferior to SL methods in terms of quantitative evaluation by synthetic data, including the existing SS spike camera reconstruction methods SSML [1] and SJRE [2]. Supervised by ground truth images, SL methods make it easier to achieve better quantitative results on synthetic data.

As shown in the table of comparison results on synthetic datasets, the SL methods tend to achieve better quantitative results. As there are gaps between real-world captured data and synthetic data, better quantitative performance does not mean better performance on real-world captured data for

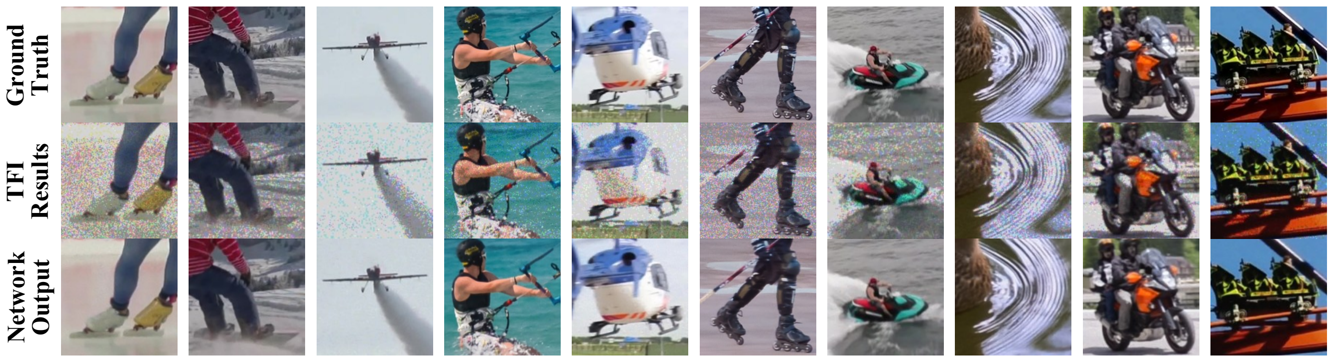


Figure 6. Visual comparison of the ground truth images, TFI results, and outputs of our network. Zoom in for better visualization.

sure. The better quantitative performance of the SL methods comes from the supervision of ground truth. According to experiments on real-world captured data, the performance of our SS method is not inferior to the SL methods. The quantitative evaluation on real-world captured data with the ground truth will effectively demonstrate the phenomenon. Considering there are no real-world CSC datasets with ground truth, it's significant to construct such a dataset in the following work.

3.5. Case Details of Ablation Study

To better introduce the ablation study, we provide more case details. For Case (B) in the first ablation study table, motion estimation is performed on pixels split from each color channel without interpolation. As the spatial resolution is $H/2 \times W/2$ due to channel splitting, we upsampled the estimated optical flows. Without motion-guided temporal pixel search, the missing pixels of each color channel are estimated by spatial interpolation in Case (C). For Cases (D), (E) and (F), the lengths of the short, medium and long temporal windows are set to 11, 21 and 31, which are the same as Cases (B), (C) and (D) in the second ablation study table. For Case (H), the values of the no-temporal-pixel positions are set to 0. The structures of Cases in the second ablation study table for the MSE module are shown in Fig. 5.

3.6. Evidence for Relieving Quantization Noise

In our experiments, the synthetic spike streams can be regarded as only involving quantization noise. If the quantization noise does not exist, we can infer nearly exact intensity by TFI [15]. By comparing TFI-based results and targets, we can visualize the effects of the quantization noise. To show the visual evidence that our method can relieve the noise, we show ground truth images, TFI results and outputs of our network in Fig. 6.

3.7. More Visual Results

Due to the data gaps, better performance on synthetic data does not mean better practical performance on real-world captured data. At present, BSS [5] is the most widely used real-world dataset captured for CSC reconstruction [4–6]. To further demonstrate the practical performance of our method, we supplement more complete visual results on the Bayer-pattern spike streams from BSS, which are shown in the following figures. These samples involve both object and camera motion. Compared to other methods, our method achieves better texture details. Please enlarge the figures for better visual comparison.

References

[1] Shiyan Chen, Chaoteng Duan, Zhaofei Yu, Ruiqin Xiong, and Tiejun Huang. Self-supervised mutual learning for dynamic scene reconstruction of spiking camera. IJCAI, 2022. 4

[2] Shiyan Chen, Zhaofei Yu, and Tiejun Huang. Self-supervised joint dynamic scene reconstruction and optical flow estimation for spiking camera. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 350–358, 2023. 4

[3] Yanchen Dong, Jing Zhao, Ruiqin Xiong, and Tiejun Huang. 3D residual interpolation for spike camera demosaicing. In *IEEE International Conference on Image Processing (ICIP)*, pages 1461–1465. IEEE, 2022. 3

[4] Yanchen Dong, Ruiqin Xiong, Jian Zhang, Zhaofei Yu, Xiaopeng Fan, Shuyuan Zhu, and Tiejun Huang. Super-resolution reconstruction from bayer-pattern spike streams. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 24871–24880, 2024. 5

[5] Yanchen Dong, Ruiqin Xiong, Jing Zhao, Jian Zhang, Xiaopeng Fan, Shuyuan Zhu, and Tiejun Huang. Joint demosaicing and denoising for spike camera. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 1582–1590, 2024. 1, 3, 4, 5

[6] Yanchen Dong, Ruiqin Xiong, Jing Zhao, Jian Zhang, Xiaopeng Fan, Shuyuan Zhu, and Tiejun Huang. Learning a deep demosaicing network for spike camera with color filter array. *IEEE Transactions on Image Processing*, 33:3634–3647, 2024. 4, 5

[7] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016. 3

[8] Seungjun Nah, Tae Hyun Kim, and Kyoung Mu Lee. Deep multi-scale convolutional neural network for dynamic scene deblurring. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3883–3891, 2017. 3, 4

[9] Seungjun Nah, Sungyong Baik, Seokil Hong, Gyeongsik Moon, Sanghyun Son, Radu Timofte, and Kyoung Mu Lee. NTIRE 2019 challenge on video deblurring and super-resolution: Dataset and study. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 1996–2005, 2019. 3, 4

[10] Jordi Pont-Tuset, Federico Perazzi, Sergi Caelles, Pablo Arbeláez, Alex Sorkine-Hornung, and Luc Van Gool. The 2017 davis challenge on video object segmentation. *arXiv preprint arXiv:1704.00675*, 2017. 3, 4

[11] Hyeonjun Sim, Jihyong Oh, and Munchurl Kim. Xvfi: extreme video frame interpolation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 14489–14498, 2021. 1

[12] Xijie Xiang, Lin Zhu, Jianing Li, Yixuan Wang, Tiejun Huang, and Yonghong Tian. Learning super-resolution reconstruction for high temporal resolution spike stream. *IEEE Transactions on Circuits and Systems for Video Technology*, 2021. 3

[13] Jing Zhao, Ruiqin Xiong, Hangfan Liu, Jian Zhang, and Tiejun Huang. Spk2ImgNet: Learning to reconstruct dynamic scene from continuous spike stream. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11996–12005, 2021. 3

[14] Rui Zhao, Ruiqin Xiong, Jing Zhao, Zhaofei Yu, Xiaopeng Fan, and Tiejun Huang. Learning optical flow from continuous spike streams. *Advances in Neural Information Processing Systems*, 35:7905–7920, 2022. 2

[15] Lin Zhu, Siwei Dong, Tiejun Huang, and Yonghong Tian. A retina-inspired sampling method for visual texture reconstruction. In *IEEE International Conference on Multimedia and Expo (ICME)*, pages 1432–1437. IEEE, 2019. 2, 4, 5

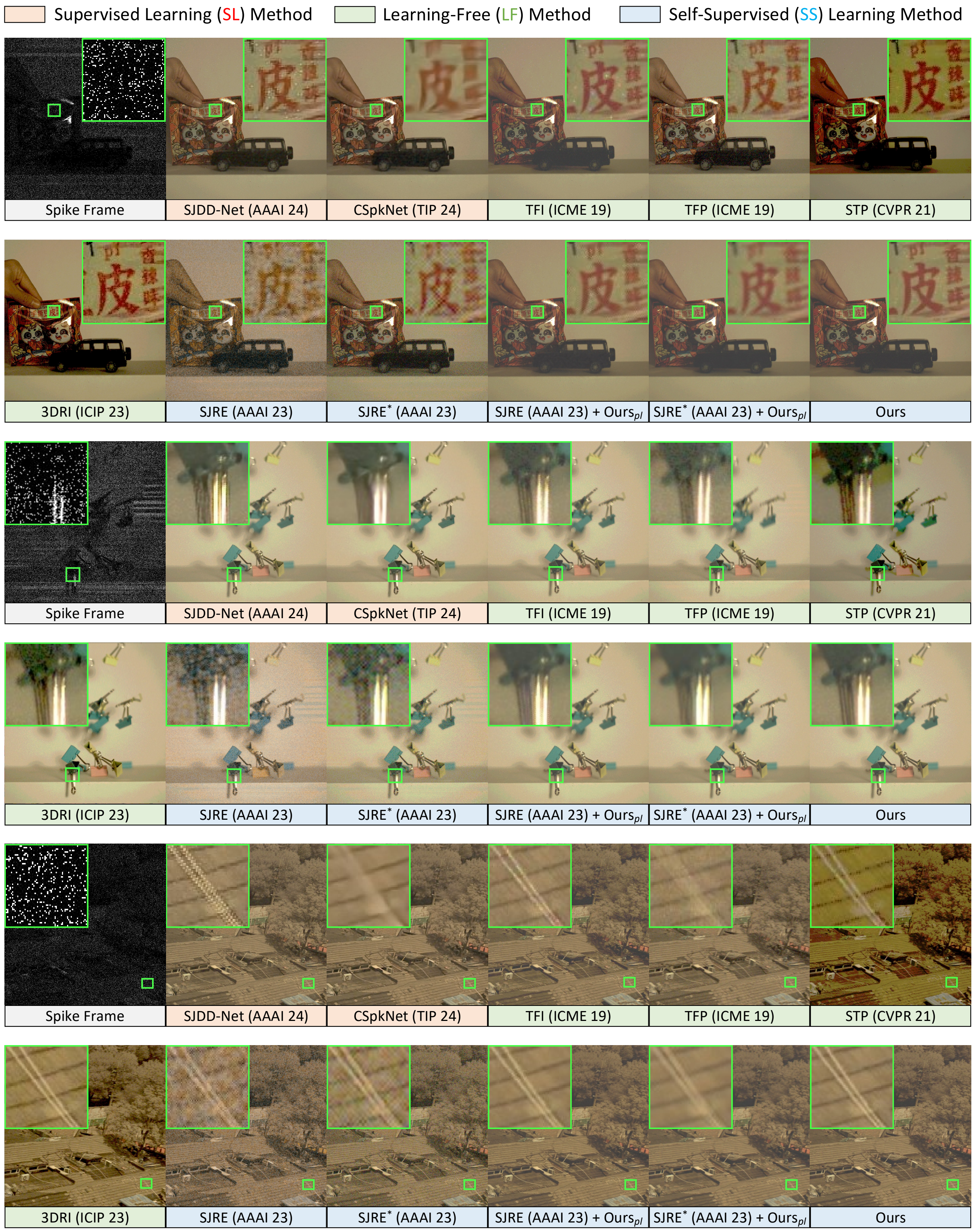
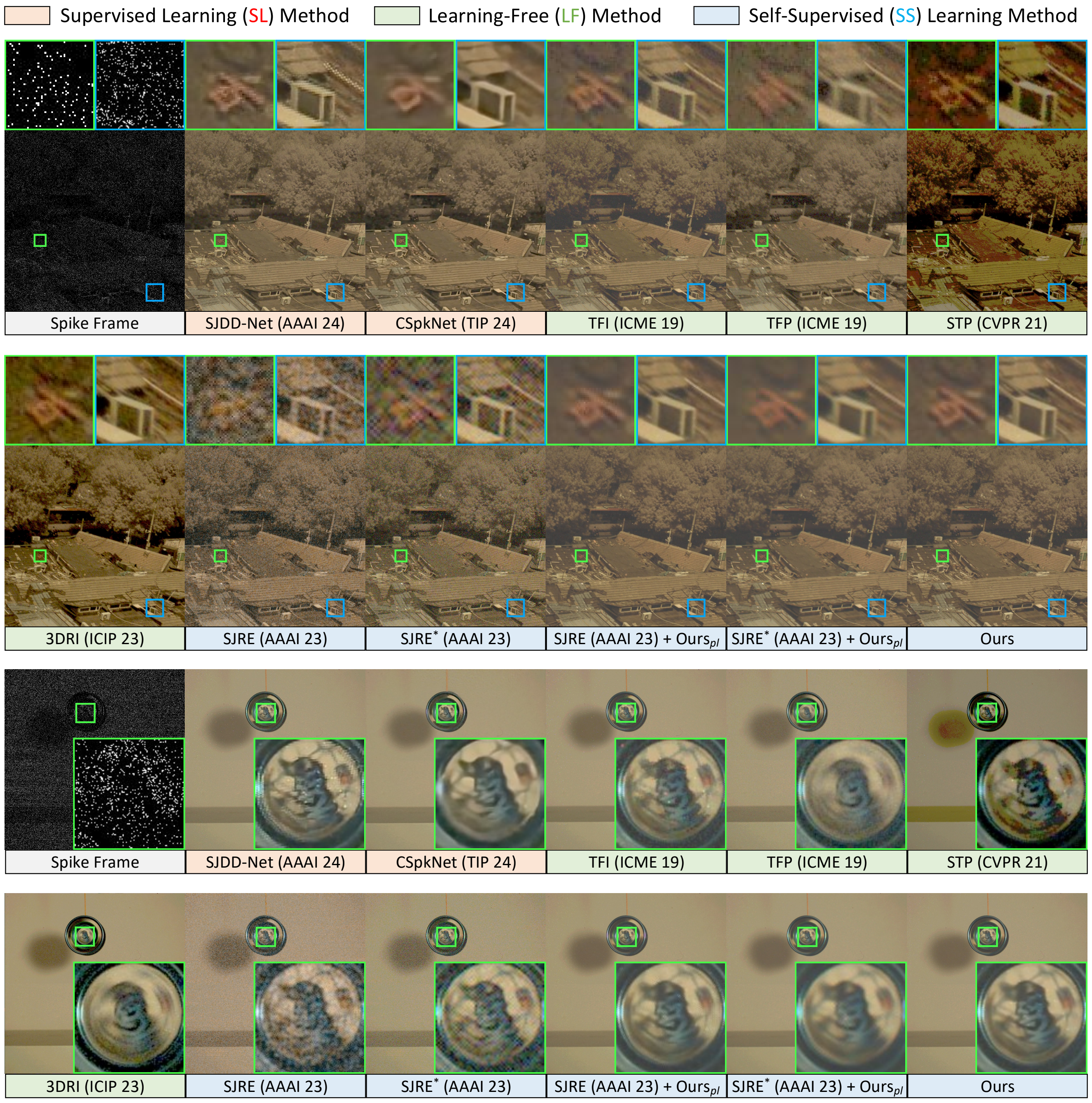


Figure 7. Visual comparison on the real-world captured Bayer-pattern spike streams. The first one records a ***fast-shaking*** pattern and a ***fast-moving*** toy car. The second one records some ***fast-dropping*** clips. The last one is recorded by a ***fast-shaking*** camera.



Supervised Learning (SL) Method
Learning-Free (LF) Method
Self-Supervised (SS) Learning Method
Spike Frame
SJDD-Net (AAAI 24)
CSpkNet (TIP 24)
TFI (ICME 19)
TFP (ICME 19)
STP (CVPR 21)
3DRI (ICIP 23)
SJRE (AAAI 23)
SJRE* (AAAI 23)
SJRE (AAAI 23) + Ours$_{pl}$
SJRE* (AAAI 23) + Ours$_{pl}$
Ours

Figure 8. Visual comparison on the real-world captured Bayer-pattern spike stream. The first sample is recorded by a ***fast-shaking*** camera. The second one records a ***fast-rotating*** yoyo-ball.

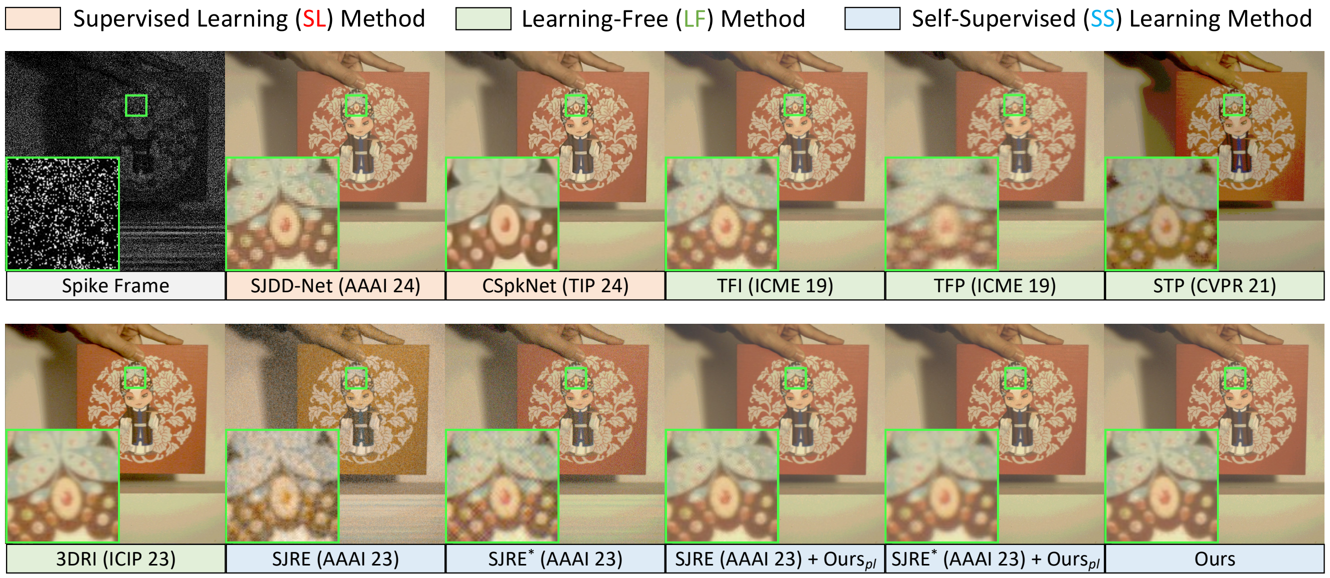


Figure 9. Visual comparison on the real-world captured Bayer-pattern spike stream, which records a ***fast-shaking*** board.