Supplymentary Material: Optical Flow Estimation for Spiking Camera

Liwen Hu[1,2]*, Rui Zhao[1*], Ziluo Ding[1], Lei Ma[1,2†], Boxin Shi[1,2,3], Ruiqin Xiong[1], Tiejun Huang[1,2,3]
[1]NERCVT, School of Computer Science, Peking University
[2]Beijing Academy of Artificial Intelligence
[3]Institute for Artificial Intelligence, Peking University

1. Additional Functions of The Simulator

Here, we introduce the additional functions provided by the spiking camera simulator (SPCS).

1.1. Random Trajectories

For each object, we can add a random motion by the function "Random trajectories". Specifically, we generate some random points first. Then, we use these points to fit a spline curve. Finally, discrete point set in the spline curve is as generated trajectory where the distance between discrete points can be controlled by related parameters. Besides, we also provide the user-defined trajectory, i.e., users can define a expected trajectory in TXT and input the TXT to SPCS.

1.2. Random Scenes

SPCS supports generating a random scene by using the funciton "Random scenes". Specifically, the function would randomly select objects and background as the scenes. Then, all objects and camera would be added random trajectories. Note that users need to download relevant materials (objects and background) by themselves, i.e., Obj files, HDR images and so on.

1.3. Noise Simulation

The noise in the spiking camera is mainly caused by dark electric current. The process of brightness accumulation can be updated as,

$$A(i,j,t) = \int_{t_{i,j}^{\mathrm{pre}}}^{t} \left(I(i,j,\tau) + I_{\mathrm{dark}}(i,j,\tau)\right) d\tau, \qquad (1)$$

where $A(i,j,t)$ is the brightness accumulation of pixel (i,j) at time t, $t_{i,j}^{pre}$ is the last time before time t where a spike is fired from pixel (i,j) and $I_{\mathrm{dark}}(i,j,\tau)$ is a noise random variable and expresses dark electric current. Note that no noise is introduced in our proposed datasets due to the lack of measurement of noise variables.

*These authors contributed equally to this work.
†Corresponding author.

1.4. Event Camera Simulation

In addition to spike camera simulation, we also provide event camera simulation. Specifically, when the change of light intensity reaches the predefined threshold ϕ, an event (x,y,t,p) is sent, i.e.,

$$|log(I(x,y,t)) - log(I(x,y,t_{x,y}^{\mathrm{pre}}))| >= \phi, \qquad (2)$$

where $t_{x,y}^{\mathrm{pre}}$ express the time when the last event is fired at the pixel (x,y) and p is the polarity of the event where p = 1 (-1) if the change is positive (negative). All code of SPCS is based on python.

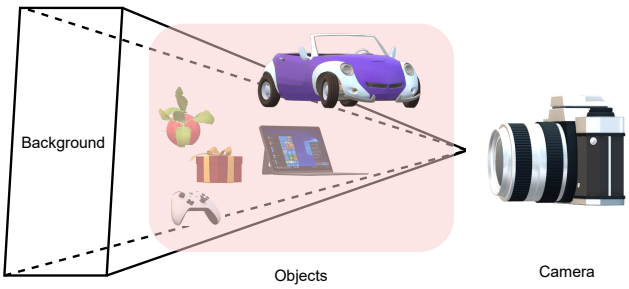

Figure 1. The design of the initial position of objects.

2. Details of Datasets

2.1. SPIFT: Spikingly Flying Things

The whole SPIFT is genrated by the function "Random scenes". Specifically, we collected a total of 30 object models and 50 background images, and randomly selected 8 objects and 1 background from the objects as a random scene each time. In addition, objects and camera are added with motion, including translation and rotation. Furthermore, we use some tricks to control the diversity and effectiveness of the scene. To ensure the effectiveness of the scene, by calculating the imaging range of the camera, we put the initial

Table 1. Average end point error comparison with other methods for estimating optical flow on PHM datasets under "$\Delta t = 10$" and "$\Delta t = 20$" settings. The best results for each scene and the best average results are marked in bold.

	Method	Input	Ball	Cook	Dice	Doll	Fan	Fly	Hand	Jump	Poker	Top	AVG.
$\Delta t = 10$	RAFT	TFP-Win15	0.587	2.539	0.999	0.475	0.571	9.930	4.118	0.273	1.014	2.348	3.013
		TFP-Win25	0.732	2.609	0.958	0.568	0.608	9.687	4.268	0.324	0.949	2.399	3.001
		TFP-Win35	0.658	2.595	0.936	0.363	0.479	10.059	3.916	0.232	0.854	2.297	2.980
		TFI	0.618	2.498	0.930	0.545	0.499	10.189	3.178	0.373	0.795	2.403	2.954
	SCFlow	Raw-Spike	0.671	1.651	1.190	0.266	0.298	8.783	1.692	0.120	1.030	2.166	**2.457**
$\Delta t = 20$	RAFT	TFP-Win15	1.317	4.521	1.796	1.079	0.919	22.105	7.172	0.326	1.679	4.515	6.171
		TFP-Win25	1.442	4.161	1.762	0.570	0.740	22.090	6.866	0.268	1.646	4.330	6.004
		TFP-Win35	1.383	4.417	2.042	0.800	0.709	21.837	6.714	0.268	1.468	4.335	6.033
		TFI	1.402	4.309	1.771	0.959	0.859	22.619	7.546	0.454	1.399	4.981	6.232
	SCFlow	Raw-Spike	1.157	3.430	2.205	0.507	0.578	21.127	4.018	0.267	1.922	4.327	**5.568**

position of all objects within the field of view of the lens as shown in Fig.1. To ensure the diversity of the scene, we randomly change the size of the imported objects, the speed of object motion and the texture information of the objects.

2.2. PHM: Photo-Realistic Motion

Each scene in PHM is carefully designed and has a lot in common with the real world, i.e., the motion of objects follows to the laws of physics. Besides, scenes in PHM contains the following properties:

1) Most areas in scenes have relative motion with sensors.

2) There is shielding between objects.

3) There is much difference in the size and motion of different objects.

3. Details of Training

During training procedure, the size of input spikes is randomly cropped to 480×800. We use Adam optimizer with the initial learning rate of 1e-4 and set the batch size as 4. For "$\Delta t = 10$" setting, we train the network on SPIFT dataset for 40 epochs. We scale the learning rate by 0.7 at every 5th epoch before 10 epochs and every 10th epoch after 10 epochs. For "$\Delta t = 20$" setting, we trian the network on SPIFT dataset for 80 epochs. We scale the learning rate by 0.6 at every 5th epoch before 5 epochs and every 10th epoch after 5 epochs.

For EV-FlowNet and Spike-FlowNet in comparison results in the main body, we warm up the learning rate for 3 epochs for a better convergence. For all the methods except RAFT, we set the weights of ℓ_1 loss constructed by output of each level as 1. For RAFT, we refer to the original training method [1] and decay the ℓ_1 loss by 0.8 from the last output to the first output.

4. Optical Flow from Reconstructed Images

For estimating optical flow from spike stream, an intuitive solution is to reconstruct image sequences from spike stream firstly, and then use frame-based methods to estimate optical flow. However, when the spike stream over a period of time is converted into a two-dimensional image, there is a time offset between the reconstructed image and the real scene which would bring additional errors to optical flow estimation. Here, we compare our SCFlow with traditional frame-based optical flow methods. We choose the state-of-the-art method RAFT [1] to be compared. For each traditional frame-based method, We use two categories of input:

A Gray image reconstructed through spike interval (Texture From Interval, TFI) [6] at each pixel.

B Gray image reconstructed through spike counting (Texture From Playback, TFP) [6] at each pixel with the length of temporal window equals to $\{15, 25, 35\}$.

Though there are more accurate reconstruction methods for the spiking camera [2–5], the computational complexity of the methods are so high that they could cost a few minutes for a frame, which is not appropriate for an end-to-end neural network.

The experimental results with gray images reconstructed from spike in $\Delta t = 10$ and $\Delta t = 20$ settings are shown in Table 1 . As shown in the tables, our SCFlow outperforms all other compared methods in average performance on PHM dataset. The visualization of the comparison with frame-based methods are shown in Fig. 2 and Fig. 3.

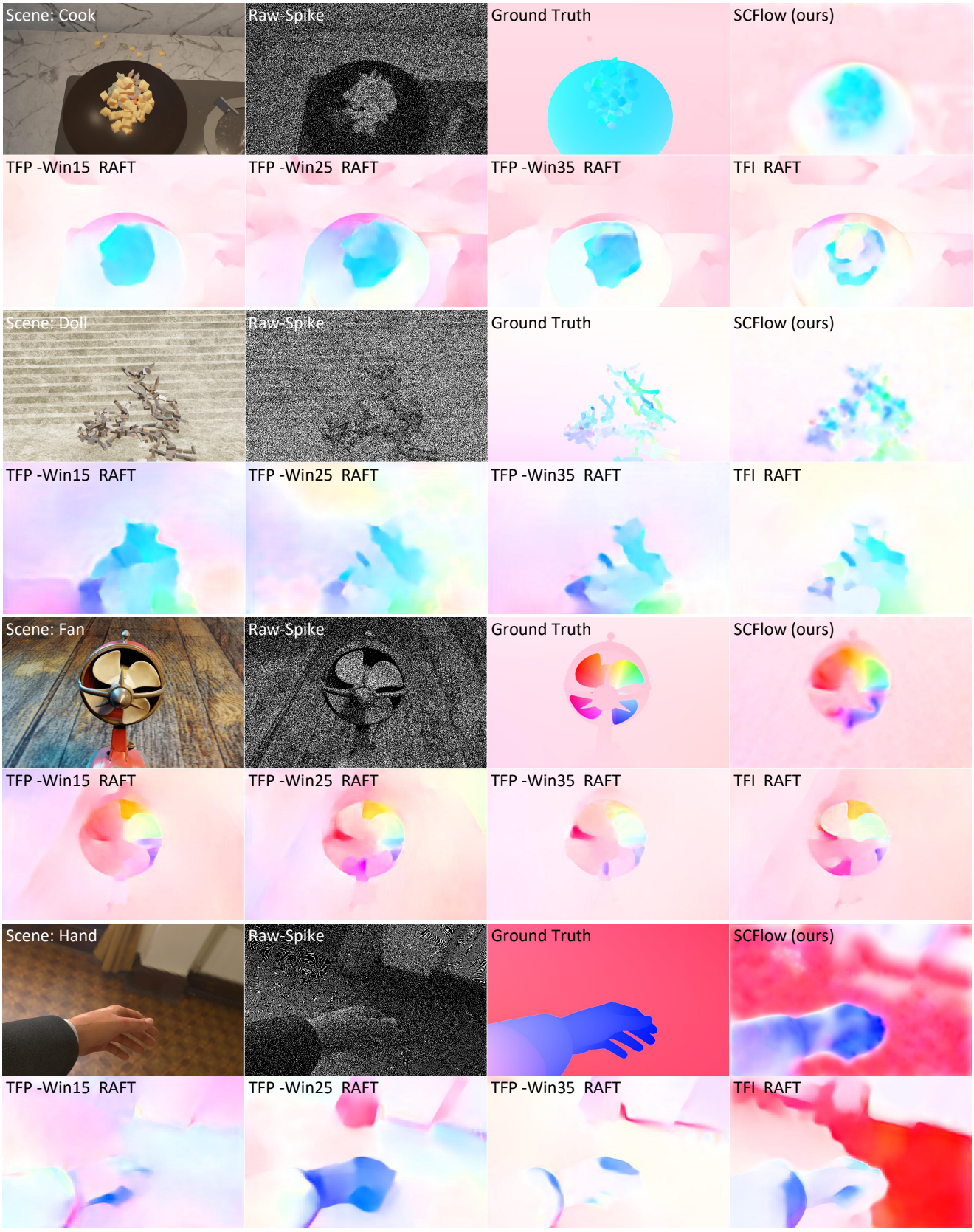

Figure 2. Visualization comparison with frame-based methods in $\Delta t = 10$ setting.

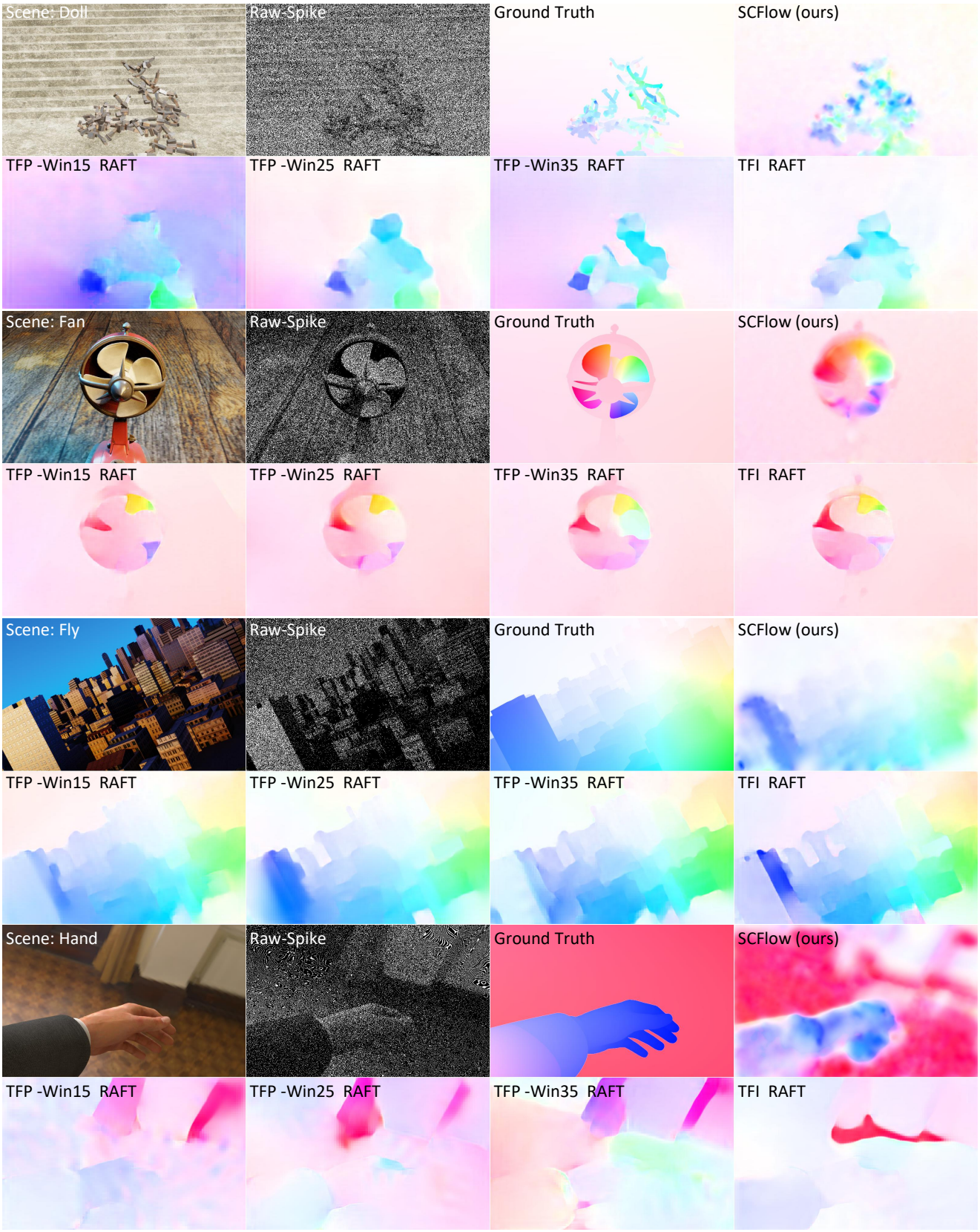

Figure 3. Visualization comparison with frame-based methods in $\Delta t = 20$ setting.

References

[1] Zachary Teed and Jia Deng. Raft: Recurrent all-pairs field transforms for optical flow. In *European Conference on Computer Vision (ECCV)*, pages 402–419, 2020. 2

[2] Jing Zhao, Jiyu Xie, Ruiqin Xiong, Jian Zhang, Zhaofei Yu, and Tiejun Huang. Super resolve dynamic scene from continuous spike streams. In *IEEE International Conference on Computer Vision (ICCV)*, pages 2533–2542, 2021. 2

[3] Jing Zhao, Ruiqin Xiong, and Tiejun Huang. High-speed motion scene reconstruction for spike camera via motion aligned filtering. In *IEEE International Symposium on Circuits and Systems (ISCAS)*, pages 1–5, 2020. 2

[4] Jing Zhao, Ruiqin Xiong, Jiyu Xie, Boxin Shi, Zhaofei Yu, Wen Gao, and Tiejun Huang. Reconstructing clear image for high-speed motion scene with a retina-inspired spike camera. *IEEE Transactions on Computational Imaging (TCI)*, 8:12–27, 2021. 2

[5] Yajing Zheng, Lingxiao Zheng, Zhaofei Yu, Boxin Shi, Yonghong Tian, and Tiejun Huang. High-speed image reconstruction through short-term plasticity for spiking cameras. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6358–6367, June 2021. 2

[6] 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, 2019. 2