Science Robotics
AAAS

Supplementary Materials for

Microsaccade-inspired event camera for robotics

Botao He *et al.*

Corresponding author: Fei Gao, fgaoaa@zju.edu.cn; Cornelia Fermüller, fermulcm@umd.edu;
Botao He, botao@umd.edu

Sci. Robot. **9**, eadj8124 (2024)
DOI: 10.1126/scirobotics.adj8124

The PDF file includes:

Other Supplementary Material for this manuscript includes the following:

Supplementary Methods

Data collection platform

We built a custom hardware platform for the event data collection and experiments, as shown in Suppl. Fig. S1. The platform is equipped with an AMI-EV, an S-EV, and an Intel RealSense D435 camera in a parallel setup in order to compare and contrast the results from the different sensor setups. Details of the AMI-EV have been described in Materials and Methods. The event cameras are DVXplorer from iniVation *(70)*, both for the S-EV and the AMI-EV. The RealSense D435 camera is used to provide RGB and grayscale images. To record the data, we used an Intel NUC10i7FNH *(80)* onboard computer, equipped with an Intel i7-10710U *(81)* CPU and 8GB RAM. We also designed an expansion board to accommodate additional sensors, such as a Lidar and an IMU (Inertial Measurement Unit), allowing our platform to integrate other data acquisition devices based on specific r equirements. To facilitate the reproduction of the work, we open-sourced *(34)* all of our schematics and the list of hardware used in this paper.

Perceptual fading effect in event cameras

The perceptual fading effect is a phenomenon that occurs in reconstructing intensity images from input event streams. Reconstruction algorithms (such as E2VID *(43)*) cause fading or blurring in regions where no new events are detected. The effect is caused by the use of memory mechanisms, such as ConvLSTM *(79)*, which learn to fade regions without new event input.

To illustrate this effect, imagine an event camera moving with rigid 3D motion and capturing a static scene. At the time, just after the camera stops moving, the scene is reconstructed. The top row of Suppl. Fig. S2 displays the perceptual fading effect in the grayscale frames reconstructed from the original event camera, and the bottom row showcases frames from our AMI event camera. As can be seen, our AMI event camera introduces additional motion to generate events in static situations, thereby preventing the perceptual fading effect and producing

clearer reconstructed grayscale frames.

This advantage is especially evident when creating high-frame-rate videos. The fading effect in E2VID is influenced by the number of iterations or grayscale frames processed. For a high framerate reconstructed grayscale frame sequence, this effect can occur quickly if no new events are generated in the image region.

Convexity analysis of the fitting error in AMI calibration

According to the Microsaccade Simplification chapter, the added micro-motion is fitted with a circular motion. Here, we formulate the calibration process as a standard unit circular motion fitting problem, in which the fitting error J can be expressed as:

$$J = J_x + J_y, \tag{S1}$$

where

$$\begin{aligned} J_x &= \int_0^{2\pi} (sin(\theta) - k \cdot sin(\theta + b))^2 \, d\theta \\ J_y &= \int_0^{2\pi} (cos(\theta) - k \cdot cos(\theta + b))^2 \, d\theta \end{aligned} \tag{S2}$$

In equation S2, $k = \frac{\delta^{est}}{\delta}$, represents the ratio of the estimated refraction angle δ^{est} to the actual refraction angle δ; $b = \theta_0^{est} - \theta_0$, represents the error of the estimated θ_0.

Now, we prove that both J_x and J_y are convex in a certain domain. Thus J is convex in this domain is also proved.

Theorem: J is convex in the domain $b \in [-\pi/2, \pi/2]$.

Proof: Firstly, we prove J_x is convex when $b \in [-\pi/2, \pi/2]$. Let us first apply the Laplace transform:

$$\begin{aligned} \mathcal{L}\{J_x\} &= \mathcal{L}\{\int_0^{2\pi} (sin^2(\theta) \, d\theta\} - \mathcal{L}\{\int_0^{2\pi} 2k(sin(\theta)sin(\theta + b) \, d\theta\} + \mathcal{L}\{\int_0^{2\pi} sin^2(\theta + b) \, d\theta\} \\ &= (k^2 + 1)\pi - 2k\pi cos(b). \end{aligned} \tag{S3}$$

Then let us write the second-order partial derivative of k and b as:

$$\begin{aligned} \frac{\partial^2 \mathcal{L}\{J_x\}}{\partial k^2} &= 2\pi, \\ \frac{\partial^2 \mathcal{L}\{J_x\}}{\partial b^2} &= 2k\pi cos(b). \end{aligned} \tag{S4}$$

Since k is a positive scale scalar, the $\frac{\partial^2 \mathcal{L}\{J_x\}}{\partial k^2}$ and $\frac{\partial^2 \mathcal{L}\{J_x\}}{\partial b^2}$ are both positive with $b \in [-\pi/2, \pi/2]$, it follows that J_x is convex in this domain.

Similarly, J_y is also proven convex with $b \in [-\pi/2, \pi/2]$. Therefore, J is convex in the domain $b \in [-\pi/2, \pi/2]$.

Quantitative error analysis of the AMI generation simplification.

For each pixel **u** in the image, there is a corresponding vector **v** in S^2. When the prism rotates, the vector **v** rotates to $\mathbf{v}_i$ represented by the fuction $f : \mathbf{u}, \theta \mapsto \mathbf{v}_i$.

$$l = \{\mathbf{v}_i | \mathbf{v}_i \in f(\mathbf{u}, \theta), \theta \in [0, 2\pi]\} \tag{S5}$$

Given the pixel **u**, the curve l is fitted with a circle in S^2. If the radius of the circle is constant, its error about each pixel is shown in Suppl. Fig. S4A. If the radius of the circle is a quadratic function of the distance r from the pixel to the center of the image, the error about each pixel is shown in Suppl. Fig. S4B.

In our modeling of the light refraction for the rotation wedge-prism, we approximate the trajectory with a circle of radius δ, instead of using different values of δ_i for different rotation angles θ_i. Suppl. Fig. S4 shows the results of a simulation for a wedge prism with a refraction angle of 0.5 degrees. Simulating the trajectories for a 640 by 480 event count image imaging a 90-degree field of view, the maximum error due to approximation was found 0.09 degrees, which is less than 2 pixels for the DVXplorer camera. This approximation error is small enough that we can safely ignore it for real-world robotics application scenarios.

Overview of the released simulator and translator

Suppl. Fig. S5 provides graphically an overview of our simulator and translator. Suppl. Fig. S5A illustrates the simulation platform that can be used to generate high-framerate RGB videos and other image types. The platform is based on WorldGen *(61)* and developed in Blender. Blender is a lightweight simulation platform that is able to generate various image types and allows for easy modification of the environment and the motion model. Moreover, it is easy to modify the optical properties of the camera like aperture size/shape, focus distance, etc, which makes it possible to generate effects like motion blur. Suppl. Fig. S6 illustrates that our platform can provide a variety of visual representations of the same scene alongside our augmented event data. This allows users of our platform to easily collect multi-modal labeled visual learning data for a variety of vision and robotics tasks. Additionally, because the event data is generated from motion information, it is not very sensitive to common issues in transferring simulated to real-world visual data like lighting conditions, object appearance, and camera imaging parameters. Users can simply use these simulated data to train their networks and apply them to real-world applications by using only very little real data for fine-tuning the performance.

Figure S5B illustrates the working principle of our simulator and translator, both of which utilize the same AMI module. The simulator takes as input a high-framerate video via the AMI module, and produces as output both a standard event stream and an AMI event stream. The translator, on the other hand, operates slightly different due to the typically lower framerate of videos in common datasets. It supports various input formats to support different kinds of datasets, including grayscale images, grayscale images combined with events, or events only. Then, low-framerate videos are interpolated into high-framerate videos using appropriate video interpolation algorithms, for example, TimeLens series *(75, 82)* for combined event and image input, SuperSlowMo *(78)* for image input and E2VID *(43)* for event input. Once this is achieved, the rest of the process parallels the simulator's operations. The AMI module com-

bines virtual AMI generation, video-to-events conversion, and AMI compensation. The AMI generation component transforms the input image by applying a specific rotation by emulating the function of our rotating wedge-prism mechanism. Next, the video-to-events conversion algorithmS *(76, 77)*, takes the video as input and translates it into an event stream. Finally, the AMI compensation module accepts the rotated events and compensates for the rotation in the final AMI event stream.

Discussion about the effect of AMI on moving objects.

Event cameras are designed to capture motion information and are most effective in scenarios involving the movement of the camera or scene objects. In situations where either the camera or the objects are moving, the motion detected at each pixel in the image plane is a composite of the introduced AMI motion and the motion within the 3D scene.

Suppl. Fig. S7 qualitatively demonstrates the combined effect of an object's (a ball) motion and the AMI motion. When the speed of the object is zero (i.e., static), the trajectory is circular because the motion is pure rotational AMI motion. As the speed of the object increases, the trajectory transitions from a circular path to a sinusoidal one, and finally, to an almost straight line. This indicates that the proportion of external motion in the overall motion gradually increases to dominate the trajectory profile as the speed of the ball increases.

Since the effect of AMI motion diminishes as the object moves faster, the AMI motion proves more effective in scenarios involving low to medium-speed scene motion, characterized by ratios between the scene motion and the introduced AMI motion. This is typically the case for most robotic applications where the AMI motion can be set to around 20Hz, a frequency significantly higher than the ego motion of objects found in these applications.

Details about compensation error measurement.

The accuracy is assessed by measuring the width of compensated edges in an statistics manner. Under ideal conditions, where there are no calibration or numerical errors, the width of compensated edges should measure precisely one pixel. However, in the real world, errors are inevitable, causing the events of a compensated edge to follow a distribution, as shown in Suppl. Fig. S9. In practice, the width of compensated edges is quantified by calculating the standard deviation of this distribution, as illustrated in Suppl. Fig. S9C.

Specifically, given a compensated edge and its event stream, shown in Suppl. Fig. S9(A and B), as described in Materials and Methods - Microsaccade Calibration and Compensation, we accumulate the events in the sample area to construct an Image of Wrapped Events (IWE), as shown in Suppl. Fig. S9C. In this image, each pixel represents the count of events it has triggered. Subsequently, we apply a Gaussian distribution fit to this data, as shown in Suppl. Fig. S9C. The standard deviation (red dashed lines) from this fit is selected as the measure for the width of the edge, which we refer to as the compensation error in our draft. In practice, we evenly choose ten edges in the event image, and then calculate their width individually. The mean width of these edges are chosen as the final compensation error.

Analysis of bandwidth-related noise

We attribute the noise related to bandwidth into three types: false-positive event trigger, jitter in timestamps and data transmission delay. We analyzed these possible errors one by one below.

False-positive event trigger

According to the noise analysis Suppl. Fig. S10, although our system can generate more noise because of higher bandwidth (Suppl. Fig. S13), our system has a lower ratio of noise, which means our system suffers less from noise because of higher signal-noise ratio.

Jitter in timestamps

From the literature *(30)*, we find that the jitter is dependent on the scene and illumination conditions, which means it is relevant to scene brightness and contrast. We did not find evidence that higher bandwidth can result in drastic higher jitter in timestamps. Moreover, according to *(30)*, since jitter is the ransom noise, post-processing methods, such as median filter, can be applied to eliminate its effects.

Data transmission delay

As we discussed in Section - Choice of deflector angle and rotation speed, with the increase in bandwidth, data transmission delay caused by the software driver can be problematic, as shown in Suppl. Fig. S12. The delay will affect the software synchronization and leads to higher compensation error. Therefore, in Section - Choice of deflector angle and rotation speed and Fig. 7, we carefully discussed how to choose the proper rotation speed and prism to make its influence minimal. Moreover, we find that the issue stems from the software driver and the ROS (Robot Operating System) messaging system, as it is eliminated when we adjust the timestamps in ROS bags during post-processing, using the timestamp from the first event rather than that from the ROS message. Therefore, this problem can be solved by the hardware manufacturers, or we can rectify it by using hardware synchronization between the event camera and the motor encoder.

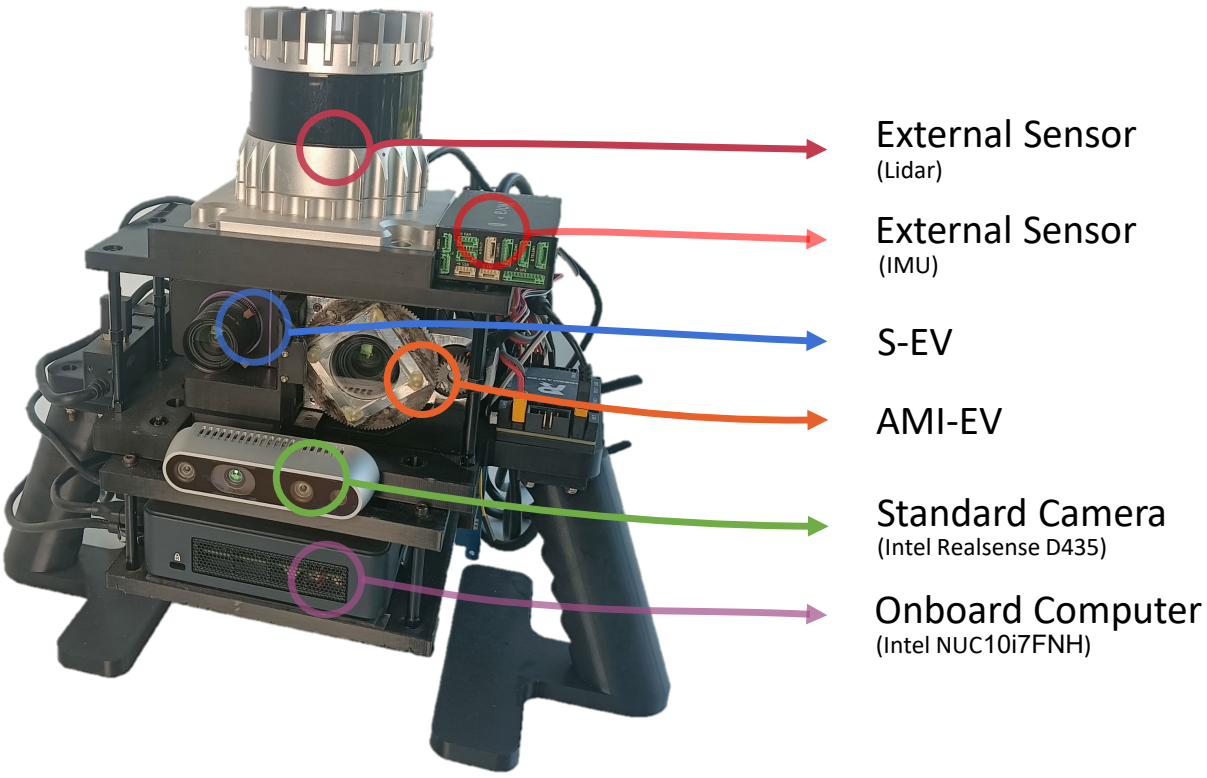


Fig. S1. Overview of the data collection platform.

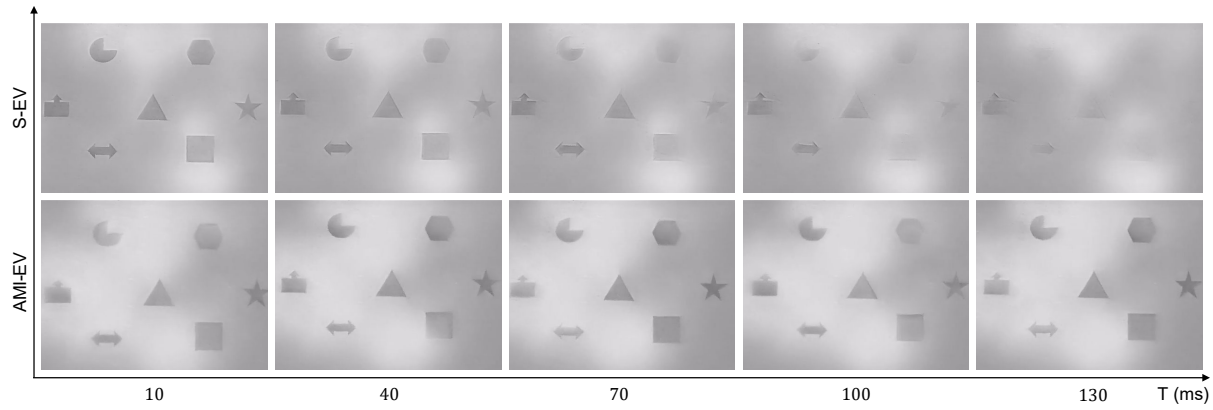


Fig. S2. Demonstration of perceptual fading effect in event cameras. The top row shows reconstructed gray-scale images of the S-EV, and the bottom row shows the corresponding reconstruction from the AMI-EV. The starting point of T is the moment when the platform stops moving and remains stationary.

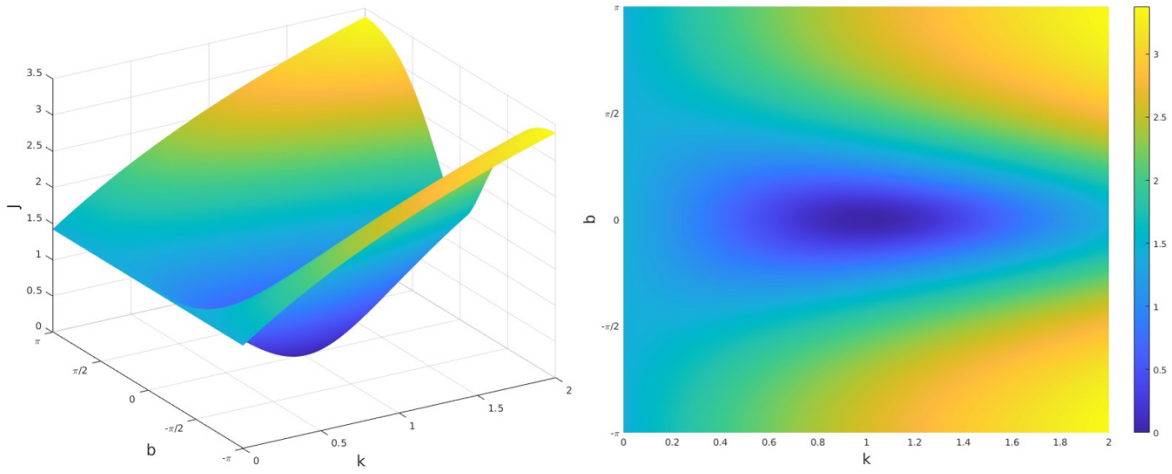


Fig. S3. Illustration of the fitting error in AMI calibration. Normalized Semi-log visulization of the function J, z-axis has log scale. **(A)** Side view. **(B)** Top view.

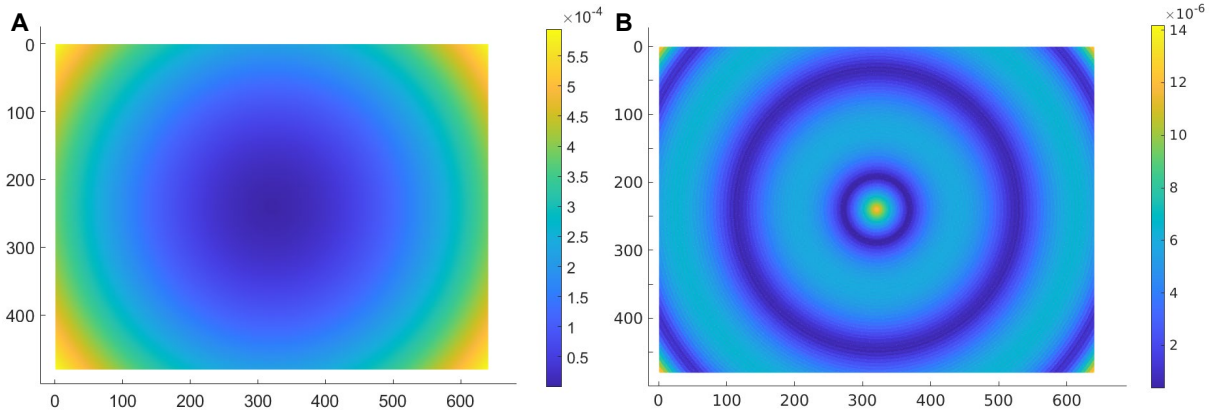


Fig. S4. Illustration of the error caused by AMI calibration and compensation. **(A)** Error resulting from AMI calibration. **(B)** Error introduced by AMI compensation.

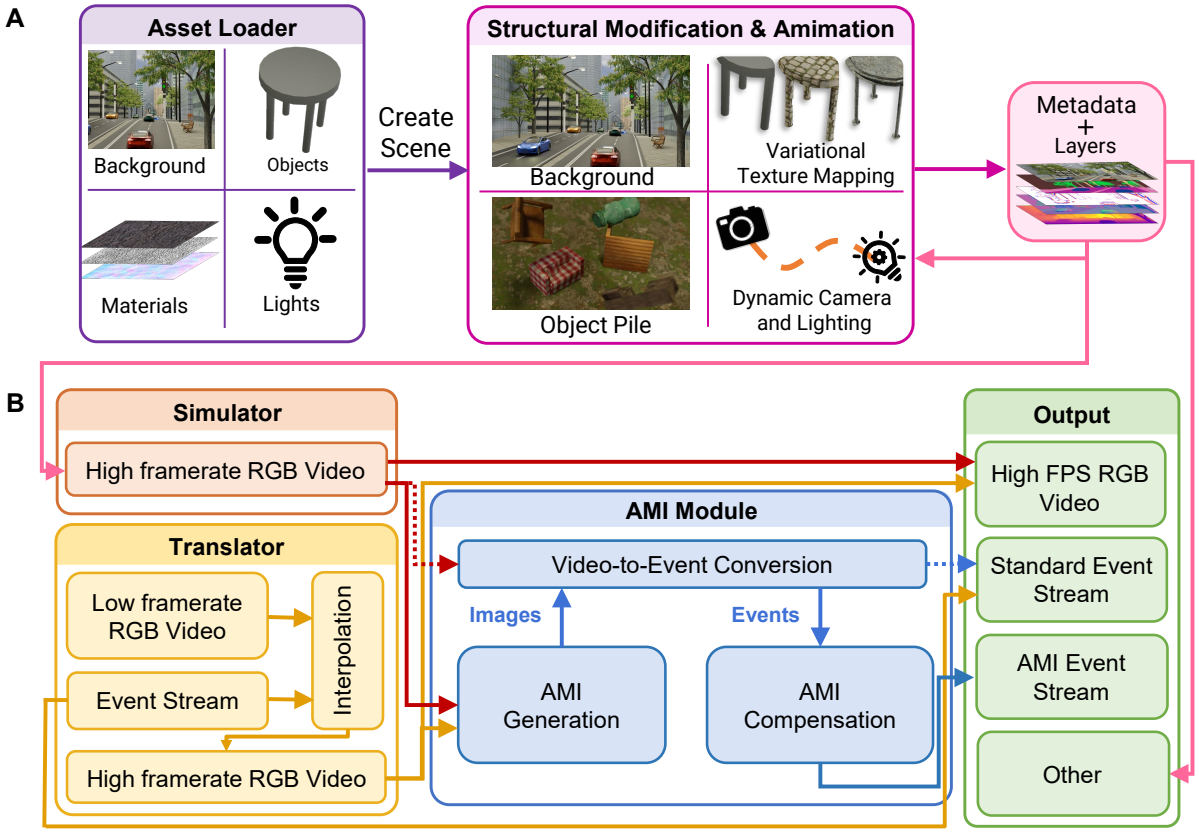


Fig. S5. Overview of the proposed simulator and translator. **(A)** An overview of the World-Gen framework: (left) Loading the assets such as background, objects, materials etc. into the environment. (middle) Modifying the scene and applying physics and motion models on different objects. (right) Generating rich ground truth data with the desired metadata. **(B)** The simulator and translator share the AMI module to transfer high-framerate RGB video into standard and AMI event streams.

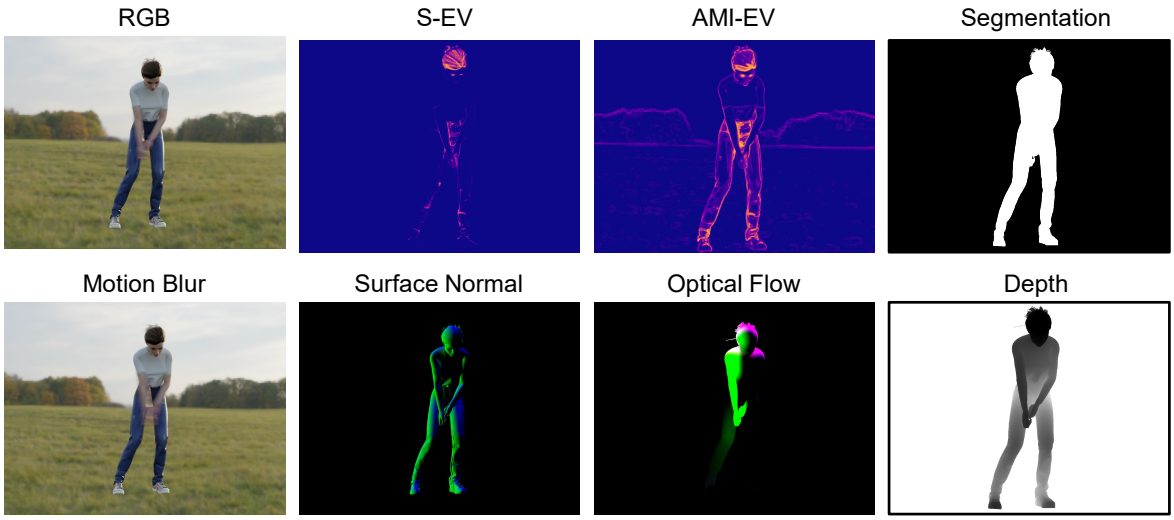


Fig. S6. Illustration of the results produced by the proposed simulator. Our simulator can provide a range of visual representations of the same scene alongside our augmented event data, as shown by examples: **(A)** RGB frame, **(B)** S-EV data, **(C)** AMI-EV data, **(D)** Segmentation map, **(E)** Simulated motion blur effect, **(F)** Surface normal map, **(G)** Optical flow map, **(H)** Depth map.

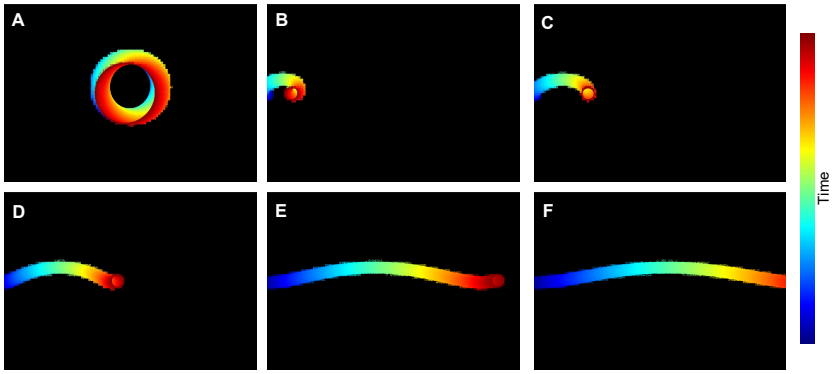


Fig. S7. Coupling effect of the AMI motion and camera motion. Illustration of the coupling effect of introduced AMI motion and the ego-motion on the trajectory of a ball. The ball is translating from left to right at different speeds. Each sub-figure shows the events recorded from one period of AMI motion (one circle). **A** The ball is static. **(B) - (F)** The ball is moving with a translation, with the speed increasing from B to F.

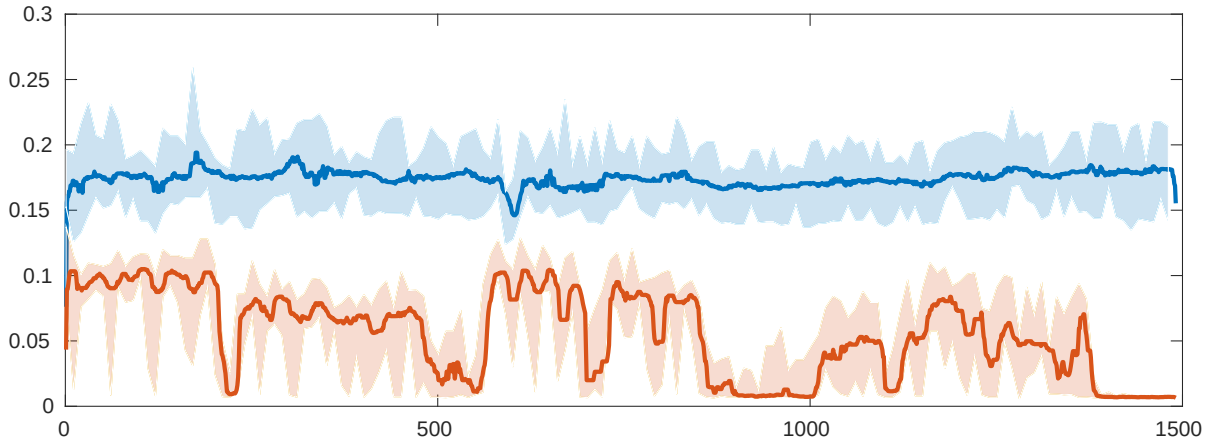


Fig. S8. Entropy of the accumulated event image. Additional experimental results complementing Fig. 3E in the paper.

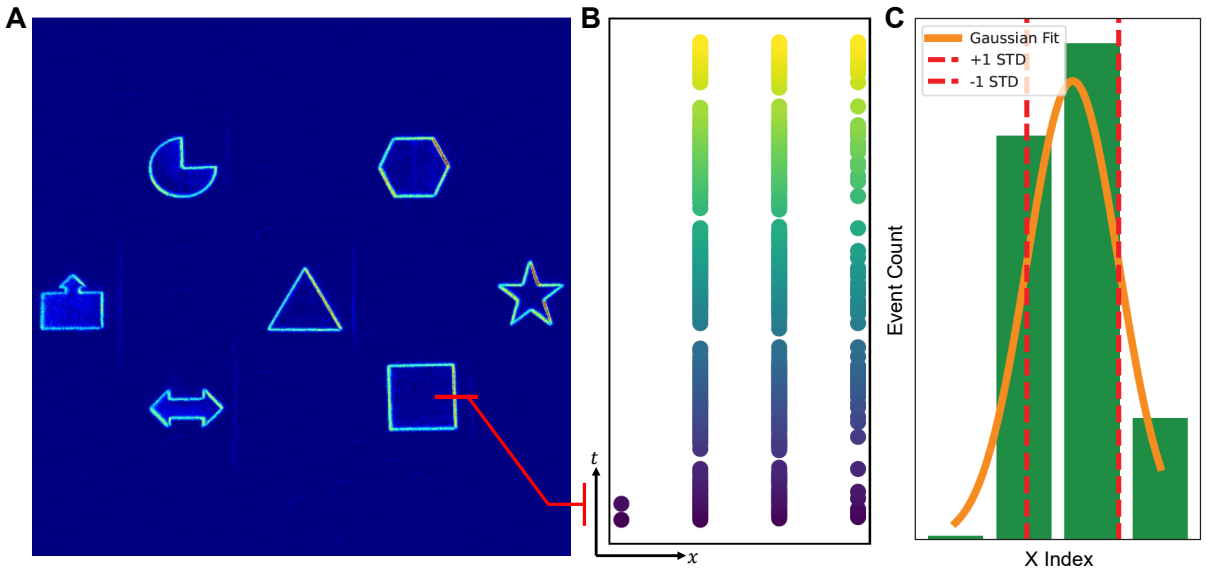


Fig. S9. Calculation of the compensation error. **(A)** Sample along the normal plane of an edge on the compensated event image. **(B)** Event Stream of the sampled area. **(C)** Construct the IWE of the event stream (green bars) and fit it using Gaussian distribution(orange c urve). Red dashed lines indicates the standard deviation of the Gaussian fit.

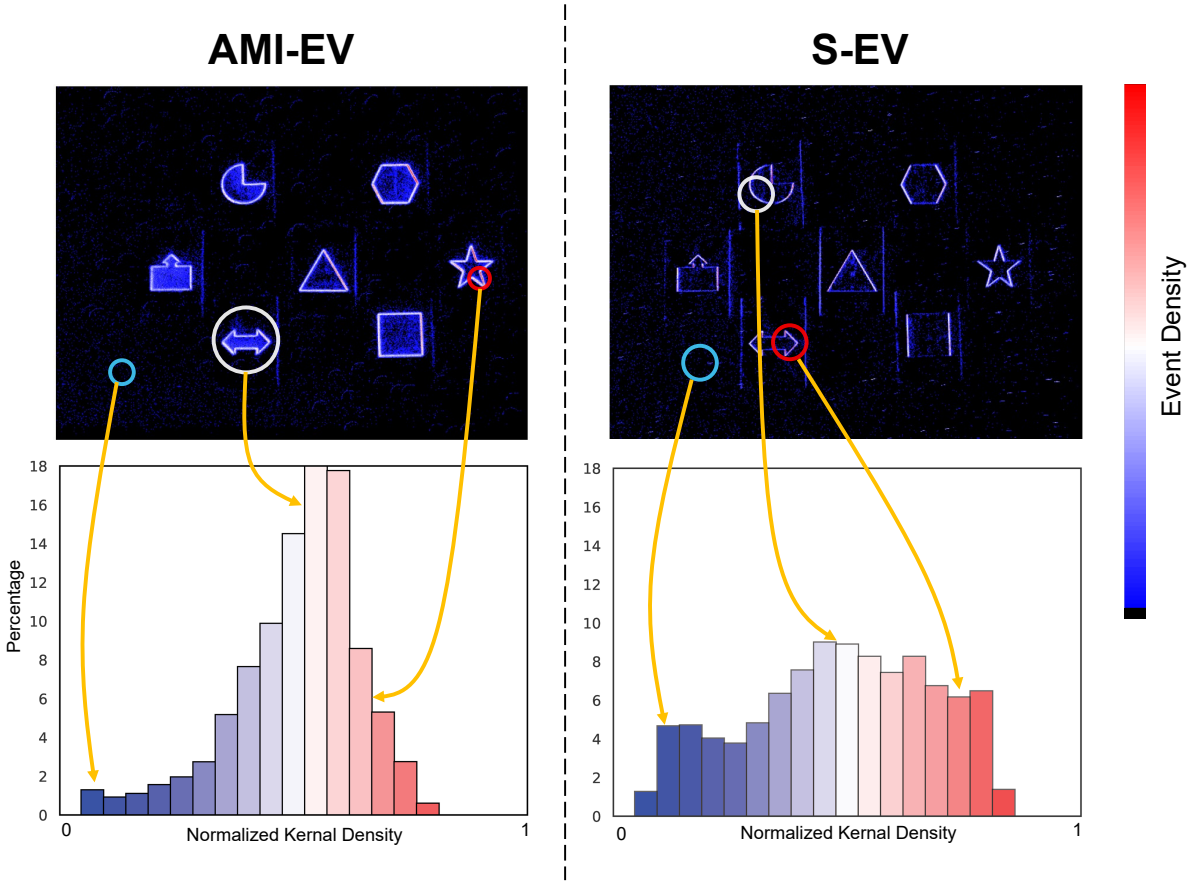
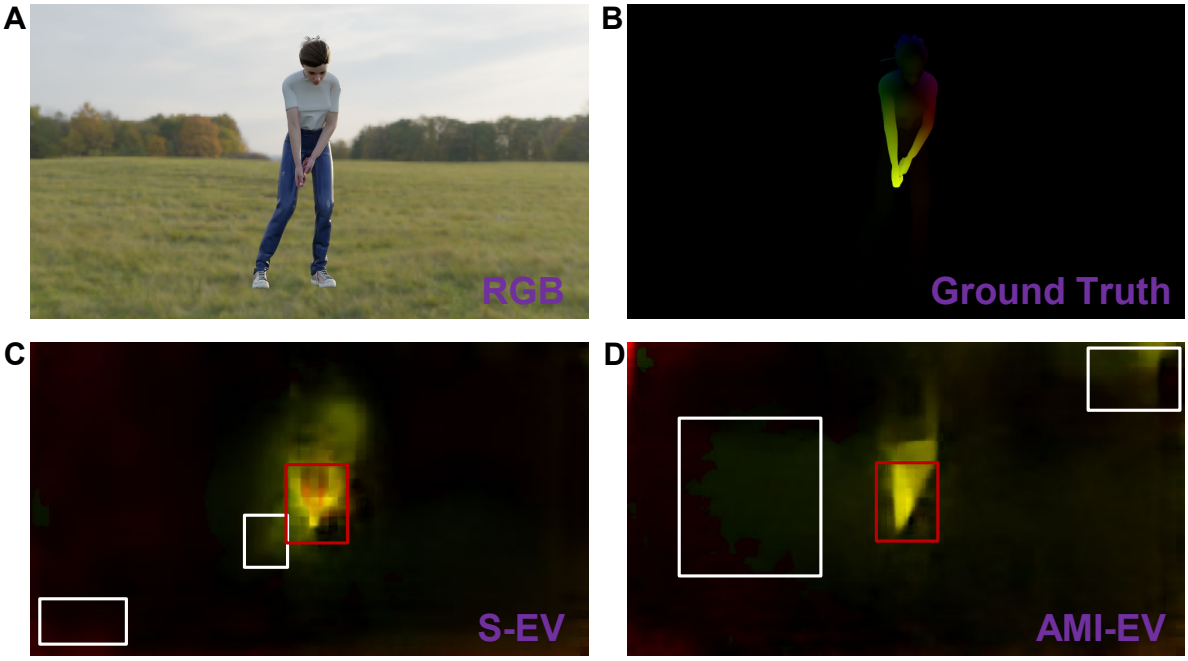


Fig. S10. Comparison of Event Density Distribution. Events with density lower than a threshold (blue circles in both images) are more likely to be considered noise. AMI-EV reports a lower ratio of both low- and high-density events (blue and red circles, respectively), and a higher ratio of medium-density events (white circles), compared to S-EV. Consequently, AMI-EV exhibits a narrower and more unimodal density distribution, which leads to a higher uniformity in the event stream and thus a more stable representation of scene features.



E

	AMI-EV	S-EV
Static & Slow Motion	0.37	0.18
Fast Motion	2.65	2.98

Fig. S11. Comparison of optical flow estimation results. The experiment is conducted in the simulated scene that contains static backgrounds, slow motion, and fast motion scenarios. We compare and contrast the optical flow estimation in S-EV and AMI-EV along with the ground truth optical flow. We apply the same optical flow algorithm (E-RAFT *(83)*) to both S-EV and AMI-EV. **(A)** Experiment scene. **(B)** Ground truth optical flow. **(C and D)** Optical flow estimation for one frame. White boxes indicate the static noise in estimation. Red boxes indicate the fast-motion area. **(E)** End-Point Error (EPE) comparison, a lower value means better performance.

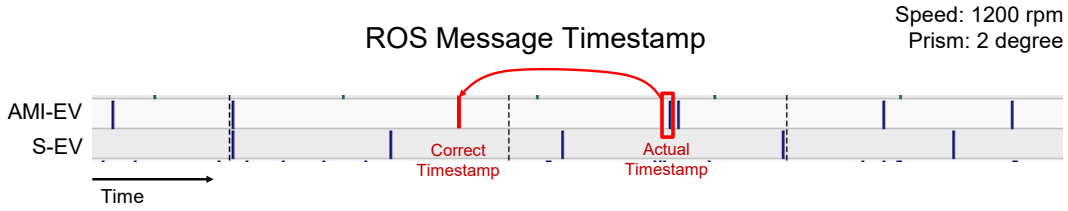


Fig. S12. Demonstration of data transmission delay.

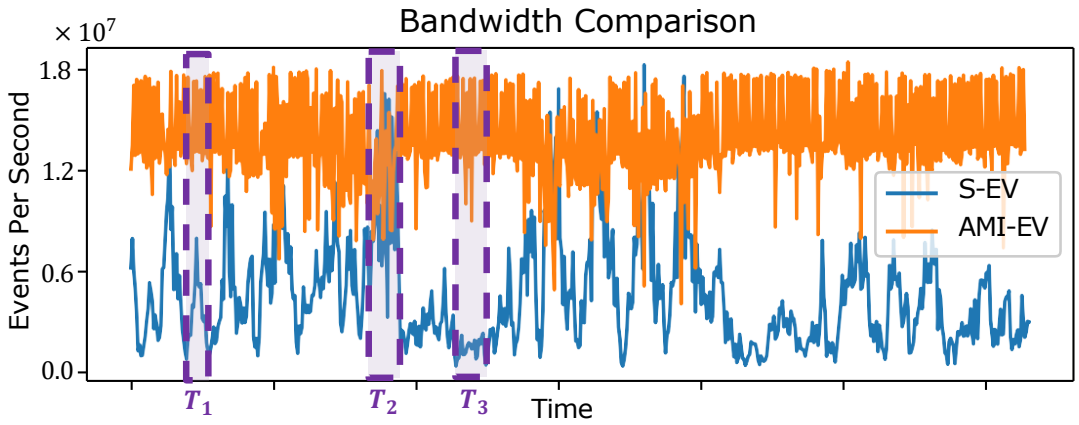


Fig. S13. Bandwidth comparison between S-EV and AMI-EV. Generally, AMI-EV reports higher bandwidth than S-EV. During the T_1, the platform moves slowly, the bandwidth of AMI-EV is $2.5-8$ times higher than S-EV's. During the T_2, the platform moves faster, the bandwidth of AMI-EV is $1-2.5$ times higher than S-EV's. During the T_3, the platform is static or nearly static, the the bandwidth of AMI-EV is over 8 times higher than S-EV's. The bandwidth of AMI-EV also reports higher independence with camera motion.

Movie. S1. Demonstration of Artificial Microsaccades (AMI) generation and compensation mechanism.

Movie. S2. Demonstration of texture enhancement in various event representations.

Movie. S3. Demonstration of feature detection and tracking.

Movie. S4. Demonstration of pose detection and pose estimation.