https://doi.org/10.2352/EI.2022.34.7.ISS-242

Accurate Event Simulation using High-Speed Videos

Xiaozheng Mou, Kaijun Feng, Alex Yi, Steve Wang, Huan Chen, Xiaoqin Hu, Menghan Guo, Shoushun Chen, Andreas Suess; OmniVision Technologies Santa Clara CA/USA;

Abstract

Event sensing is a novel modality which is solely sensitive to changes of information. This redundancy reduction can be utilized to achieve high temporal resolution, reduce power consumption, simplify algorithms etc. The hardware-software co-design of event sensors and algorithms requires early simulation of the sensor system. It has been shown that high-speed video is well suited to derive such event data for temporal contrast based event sensors, but the simulators published so far neglect phenomena such as readout latency or refractory period. This paper presents ongoing modeling activities at OmniVision Technologies.

Introduction

Event sensors have recently been gaining lots of popularity [1,2]. The idea of using bio-inspired visual sensing dates back to Prof. Carver Mead's work around 1986-1991 [2,3]. Given the recent advances in CMOS imaging and especially 3D integration technologies such as sensor stacking, event sensing has become an increasingly promising technology for e.g. power efficient high-speed data acquisition or high dynamic range [4,5]. Numerous use-cases have been reported such as gesture recognition, SLAM, surveillance or ADAS [1,2]. Significant research is directed towards novel algorithm and hardware developments by academia as well as industry. Accurate system simulation enables improved understanding of the underlying hardware and enables early algorithm development. Event simulators for generating training data from high-speed video have been reported [6–8]. These simulators employ pre-processing such as re-sampling to a desired resolution and input frame-rate as well as color-space conversion to generate the equivalent of photo-current and "front-end" voltage proportional to log-intensity for each pixel. Both simulators also consider threshold mismatch for individual pixels and can furthermore generate noise events. It is noteworthy that the V2E simulator considers a light-dependent latency model [6,8]. These works treat pixels independently which enables distribution of the calculations to e.g. different cores of a GPU. However, an implicit drawback of this approach is that it neglects the finite latency of the readout by means of peripheral circuitry. Also refractory period is not considered in today's simulators. The simulator in [7] is designed to be a generic simulator offering "typical" event data. This simulator is not linked to an actual hardware implementation and is hence, not calibrated. The V2E simulator has been indirectly calibrated by adjusting e.g. comparator threshold settings to yield similar order-of-magnitude event rates as expected from a known event camera.

This paper presents a simulator that considers realistic system limitations such as finite latency of the peripheral readout circuitry as well as refractory period. The impact of these phenomena on event rate is studied for a set of high-speed video data.

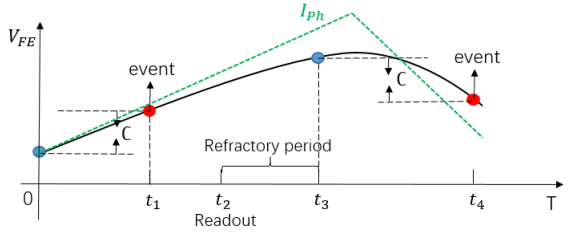


Figure 1. *Schematic pixel operation. (blue circle: reference voltage; red circle: triggered event; C: voltage threshold)*

Methodology

Fig. 1 schematically explains the operation of an event pixel. It assumes that the pixel was activated at $t = 0$ where the log-intensity is sampled and stored as a reference voltage. A difference-detecting circuit continuously tracks the difference of the momentary log-intensity compared to the stored reference level. If at a subsequent time point (t_1) the log-intensity exceeds the sampled log-intensity plus/minus a predefined temporal contrast level C an event is triggered. The event firing exhibits a latency relative to the ideal firing time-point which is determined by a) the front-end latency as well as b) the overall congestion of the readout periphery and hence, is highly scene dependent. Once a pixel has been read (t_2) a programmable refractory period is started. The pixel is then automatically re-activated at the end of the refractory period (t_3) and a new reference log-intensity is sampled. The refractory period can help suppress readout bandwidth occupation by very active or noisy pixels in order to allow less active pixels to communicate at lower latency.

The simulator is composed of three essential modules. The first module estimates photocurrents for each pixel based on a given input video-frame-set at known frame rate. Similarly to [7,8] the optical flow method described in [9] can be used to interpolate the input video. Then, based on an inverse model irradiance spectra are estimated and mapped onto quantum efficiency (QE) curves of a target sensor and scaled according to a desired illuminance. The second module models analog circuit imperfections such as finite pixel latency through bandwidth limitation, noise and mismatch. It converts the photocurrent maps of the first module into front-end voltage maps representing the input of the difference detecting circuit of an event pixel. The third module ultimately models the peripheral readout circuitry such as event scanning operation, refractory period, event formatting etc. The following three sections will highlight key aspects of those modules.

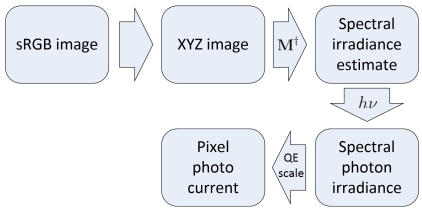


Figure 2. *Program flow explaining the generation of photocurrent transients.*

Photocurrent Estimation

Typically high speed video frames are available in a standard color space, e.g. sRGB. In previous works, a simple sRGB to luma conversion is carried out to obtain the corresponding (linear) luma frames [6]. The luma frames are subsequently treated as pixel irradiance map for further simulation. This scheme, although being convenient and illustrative, doesn't take into account the discrepancy between the luminosity function and the sensor's QE curve. Also it lacks flexibility for custom illumination settings.

To address these issues, we implement the process shown in Fig. 2. Here for each input frame, we first convert it to a linear color space, e.g. XYZ, and then apply a Moore–Penrose pseudo-inverse of the corresponding color matching functions (CMF) to obtain a least-norm estimate of the spectral irradiance map. The spectral irradiance is then converted to spectral photon irradiance according to the photon energy. Finally, we obtain the pixel photocurrent by projecting the spectral photon irradiance onto the QE curve and scaling according to pixel size and illuminance level.

In equations, we first write down the image plane illuminance using the CIE photopic luminosity function [10]:

$$E_\mathrm{v} = \kappa \cdot \hat{E}_\mathrm{e} \int \bar{y}(\lambda) \cdot \tilde{E}_\mathrm{e}(\lambda) \cdot \mathrm{d}\lambda \tag{1}$$

Here, $\kappa = 683.002\,\mathrm{lm\,W^{-1}}$ [11], $\bar{y}(\lambda)$ is the luminosity function, $\hat{E}_\mathrm{e}$ and $\tilde{E}_\mathrm{e}(\lambda)$ are the peak and normalized spectral irradiance on pixels, respectively. Meanwhile, the photocurrent of a pixel can be written as:

$$i_\mathrm{p} = \int e \cdot \Phi_\mathrm{ph}(\lambda) \cdot q(\lambda) \cdot \mathrm{d}\lambda = \frac{a \cdot e}{h \cdot c} \cdot \hat{E}_\mathrm{e} \int \lambda \cdot q(\lambda) \cdot \tilde{E}_\mathrm{e}(\lambda) \cdot \mathrm{d}\lambda \tag{2}$$

Here Φ_ph is photon flux, a is the pixel area, e is the elementary charge, h is the Planck constant and c is the speed of light. Combining Eq. 1 and 2, we obtain:

$$i_\mathrm{p} = \frac{a \cdot e}{h \cdot c} \cdot E_\mathrm{v} \cdot \frac{\int \lambda \cdot q(\lambda) \cdot \tilde{E}_\mathrm{e}(\lambda) \cdot \mathrm{d}\lambda}{\kappa \cdot \int \bar{y}(\lambda) \cdot \tilde{E}_\mathrm{e}(\lambda) \cdot \mathrm{d}\lambda} \tag{3}$$

Now discretizing Eq. 3 with the least-norm estimation for spectral irradiance, we have

$$i_\mathrm{p} = \frac{a \cdot e}{h \cdot c \cdot \kappa} \cdot E_\mathrm{v} \cdot \frac{\mathbf{q}^T \Lambda \mathbf{M}^\dagger \mathbf{p}}{\bar{\mathbf{y}}^T \mathbf{M}^\dagger \mathbf{p}} = \frac{a \cdot e}{h \cdot c \cdot \kappa} \cdot \frac{E_\mathrm{v}}{Y} \cdot \mathbf{q}^T \Lambda \mathbf{M}^\dagger \mathbf{p} \tag{4}$$

Here, $\Lambda \in \mathbb{R}^{n \times n}$ is a diagonal matrix containing the wavelengths; $\mathbf{q} \in \mathbb{R}^n$ represents the sensor's QE curve; $\mathbf{p} \in \mathbb{R}^3$ contains the pixel values in XYZ space and $\mathbf{M}^\dagger \in \mathbb{R}^{n \times 3}$ is the pseudo-inverse of the XYZ CMF's denoted as $\mathbf{M} \in \mathbb{R}^{3 \times n}$.

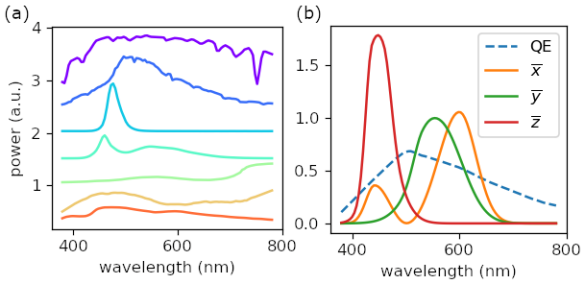


Figure 3. *(a) Selected sample spectra. From top to bottom: sun, sky, blue LED, phosphor, leaf, blue paper, skin. (b) Sample QE curve and XYZ CMFs.*

By convention, we set $Y = 1$ for the white point in the image, and denote the corresponding absolute illuminance as E_v^w. Assuming constant normalization across the pixel array, we have:

$$\frac{E_\mathrm{v}}{Y} = \frac{E_\mathrm{v}^\mathrm{w}}{1} = E_\mathrm{v}^\mathrm{w} \tag{5}$$

Using Eq. 5, Eq. 4 can be further simplified as:

$$i_\mathrm{p} = \alpha \cdot E_\mathrm{v}^\mathrm{w} \cdot \mathbf{q}^T \Lambda \mathbf{M}^\dagger \mathbf{p}\,, \quad \alpha = \frac{a \cdot e}{h \cdot c \cdot \kappa} \tag{6}$$

Here α depends on the sensor pixel size. Eq. 6 gives us a simple linear relation between photocurrent and pixel values and accounts for different sensor parameters. We provide the variable E_v^w for users to specify the global illumination level, as in practice CIS camera settings would adjust according to overall illumination, but EVS sensor's behavior would vary significantly.

To compare with the photocurrent estimation in [6], we collected spectra of some common targets shown in Fig. 3(a). In Fig. 3(b) we plot a sample EVS sensor QE curve along with the XYZ color matching functions. Note that the luma measurement essentially originates from the $\bar{y}(\lambda)$ luminosity curve, which is usually different from the sensor's QE curve.

The recovered photocurrents are shown in Fig. 4. The ground truth values are obtained by directly applying the QE curve to the spectra. For a fair comparison, all photocurrents are normalized to the "sun" photocurrent obtained by the same method. According to the plot, for "phosphor", "leaf" and "skin", the two methods result in similar photocurrent estimate, likely because the X and Z CMFs don't provide extra information of the original spectra. However, for "sky", "blue LED" and "blue paper", there is apparent improvement in the estimation accuracy by using the pseudo-inverse method. This is most likely because large portions of the spectra are not sufficiently measured by the luminosity curve.

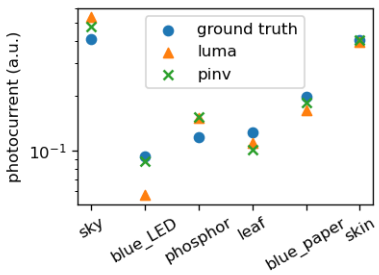


Figure 4. *Comparing photocurrent reconstruction results.*

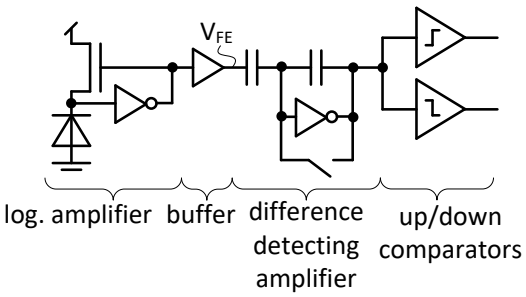


Figure 5. *Simplified event-sensing circuitry after [12]*

Front-End Voltage Simulation

This section addresses the modeling of the latency of a typical event-sensing front-end circuit (c.f. Fig.5). Fundamentally, the readout circuitry converting the incoming photocurrent into a logarithmic voltage v_{FE} may have more than one pole. For low photocurrents the logarithmic amplifier may have a dominating pole due to the small photocurrents charging/discharging the feedback loop. At higher currents the bandwidth limitation of the buffer may start to play a role. Nonetheless, for this work we considered a first order low-pass filter (LPF) behavior with photocurrent dependent time-constant alike [6]. The presented model is empirical but calibrated against cadence circuit simulations.

Fig. 6 depicts the transfer characteristic of the logarithmic amplifier under DC excitation. We express $v_{\mathrm{FE\text{-}DC}}$ as a polynomial fit $v_{\mathrm{FE\text{-}DC}} = \sum_j \alpha_j \cdot [\ln([i_{\mathrm{p}} + i_0])]^j$ in which i_0 represents the dark current. In order to consider latency we update $v_{\mathrm{FE}}(k+1) = v_{\mathrm{FE}}(k) + \Delta v_{\mathrm{FE}}(k)$ as follows:

$$\Delta v_{\mathrm{FE}}(k) = (v_{\mathrm{FE\text{-}DC}}(k) - v_{\mathrm{FE}}(k)) \cdot \left[1 - \exp\left(\frac{-\delta t}{\tau_{\mathrm{FE}}}\right)\right] \overset{\delta t \to 0}{\approx} (v_{\mathrm{FE\text{-}DC}}(k) - v_{\mathrm{FE}}(k)) \cdot \frac{\delta t}{\tau_{\mathrm{FE}}} \tag{7}$$

with timestep δt and LPF time-constant τ_{FE}. This equation corresponds to a step-response of a first order LPF. Using Taylor-series approximation for $\delta t \to 0$ this converges to the solution that could've been derived using a classic *Forward-Euler discretization* of the differential equation representation which is a standard solver used in circuit simulations [13]. The time-constant τ_{FE} is evaluated by simulations of step-response characteristics $i_{\mathrm{p}}(0) \to i_{\mathrm{p}}(1)$. Here, $3 \cdot \tau_{\mathrm{FE}}$ was read from the point where the relative voltage increase meets 95 % as it would for an ideal LPF: $\frac{v_{\mathrm{FE}}(3\cdot\tau_{\mathrm{FE}}) - v_{\mathrm{FE}}(0)}{v_{\mathrm{FE}}(\infty) - v_{\mathrm{FE}}(0)} = 95\,\%$. Hence, by definition τ_{FE} becomes a function of $i_{\mathrm{p}}(0)$ and $i_{\mathrm{p}}(1)$. For a given $i_{\mathrm{p}}(0)$ we model:

$$\tau_{\mathrm{FE}}[i_{\mathrm{p}}(0), i_{\mathrm{p}}(1)] = a_{i_{\mathrm{p}}(0)} + b_{i_{\mathrm{p}}(0)} \cdot i_{\mathrm{p}}(1) + c_{i_{\mathrm{p}}(0)} \cdot [i_{\mathrm{p}}(1)]^2, \tag{8}$$

where $a_{i_{\mathrm{p}}(0)}, b_{i_{\mathrm{p}}(0)}, c_{i_{\mathrm{p}}(0)}$ are stored in a look-up table. The two-dimensional $\tau_{\mathrm{FE}}[i_{\mathrm{p}}(0), i_{\mathrm{p}}(1)]$ approach allows for larger step-size compared to [6] and is flexible to take several poles as well as slewing into account which [6] doesn't address.

Note that the simulator uses frame-based photocurrent maps as described in the photocurrent estimation section above. These are first up-sampled using an optical flow based network [9]. Then linear or cubic interpolation is used generate a time-continuous photocurrent between adjacent frames. Between each set of adjacent frames the photocurrent is resampled at step-size δt to yield $v_{\mathrm{FE\text{-}DC}}(k),\ k \in \{1, \ldots N\}$ used in Eq. 7. After all timesteps between adjacent frames are calculated a linear fit is determined and provided to the event-scanner. $N \gg 1$ is required in order to ensure small error. Fig. 7 demonstrates reasonable resemblance of

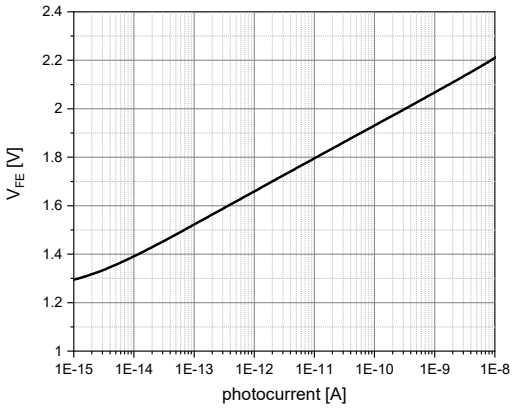


Figure 6. *Logarithmic amplifier transfer characteristic*

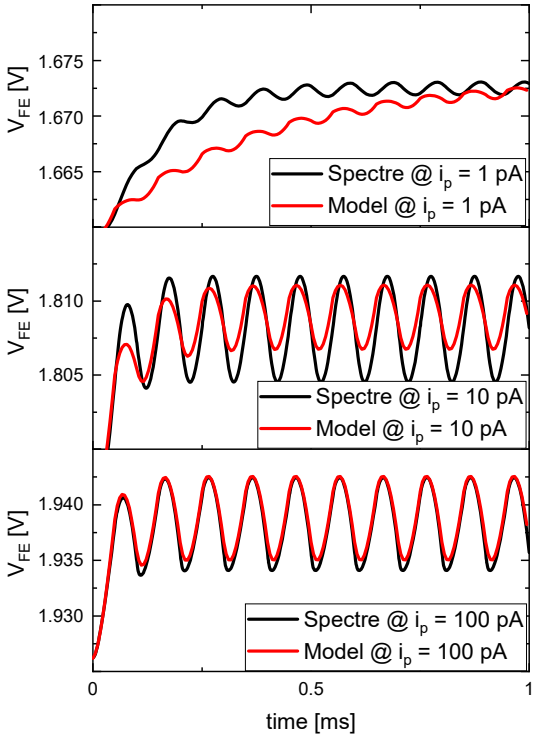


Figure 7. *Verification of first order LPF model with current-dependent time-constant vs.* Spectre *circuit simulations using triangular photocurrent at* 50 % *temporal contrast and* 10 kHz.

the simplified first-order LPF model compared to a full circuit simulation using *Spectre*. The simplified model offers a balance between accuracy and scalability required to execute array-level simulations of entire video streams. We also model noise and FPN which, however, is beyond the scope of this paper.

Peripheral Event Readout

In our simulator two parallel timelines are maintained. The timeline T_g is used to track event generation and the timeline T_s is used to model the event readout by the peripheral scanning circuitry. Both timelines are progressed alternatingly until new frames have to be loaded and eventually all frames have been processed. Compared to operating T_g and T_s simultaneously at a fix step size this allows for an improved accuracy vs. computational efficiency tradeoff as the scan timeline doesn't need to be processed if no events have been created and T_s can progress completely independent of T_g which advances in increments of Δt. The simulator is initialized based on the first frame: $\mathbf{v}_{\text{ref}} \leftarrow \mathbf{v}_{\text{FE}}(1), \mathbf{t}_{\text{ref}} \leftarrow \mathbf{0}$. Line 3 in Alg. 1 doesn't affect anything initially and its purpose will become clear later. The function $g[\mathbf{v}_{\text{FE}}(i), \mathbf{v}_{\text{FE}}(i-1), \mathbf{t}]$ determines a linear interpolation between the frames $i, i-1$ and evaluates said interpolation at $\mathbf{t}$. In case the scan timeline is equally or more advanced than the event generating timeline the difference between instantaneous and reference voltage is compared per-pixel against the contrast threshold C. If $|\mathbf{v}_{\text{ins}} - \mathbf{v}_{\text{ref}}| > C$ is valid and the reference time lies in the past ($\mathbf{t}_{\text{ref}} < T_g$) and a pixel didn't already trigger an event ($\mathbf{t}_{\text{trg}} = 0$) a new event is generated and stored temporarily ($\mathbf{t}_{\text{trg}} = T_g$). $||\mathbf{t}_{\text{trg}}||_0^0$ tracks if there are unread events. Here, $||\cdot||_0^0$ measures the amount of non-zero entries (notation borrowed from *L0-"norm"* [14]). If $||\mathbf{t}_{\text{trg}}||_0^0 > 0$ the scanner starts. Otherwise the scan time is set to the event generating time and the generator progresses ($T_s \leftarrow T_g$). In case events were found a row scanner at location R_s gradually progresses through all rows and writes events to file. Pixels that have been read receive an updated reference time depending on the refractory period $\mathbf{t}_{\text{ref}}[R_s] \leftarrow T_s + \Delta t_{\text{refrac}}$. $\mathbf{v}_{\text{ref}}$ is updated if $\mathbf{t}_{\text{ref}}$ falls into the same frame: $\mathbf{t}_{\text{ref}} \in \left(\frac{i-2}{f}, \frac{i-1}{f}\right]$. If the refractory period points into future frame intervals $\mathbf{v}_{\text{ref}}$ will be updated upon loading the corresponding frame (Line 3 in Alg. 1). Note that this evaluation and the assignment are computed in parallel for each pixel. The scanner timeline is advanced depending on the amount of events in a particular row $T_s \leftarrow T_s + \Delta t_{\text{col}} \cdot ||\mathbf{t}_{\text{trg}}[R_s]||_0^0 + \Delta t_{\text{row}}$. Here, Δt_{col} and Δt_{row} are the scan cost per column and row. Events that have been read are removed from the list of unread events (Line 19). The event scanner is interrupted if the scan timeline progresses beyond the event fire timeline or if there are no events left to be read. Note that line 23 ensures that the scan timeline is advanced in case all events have already been read and that the generating timeline was more advanced than the scanning timeline.

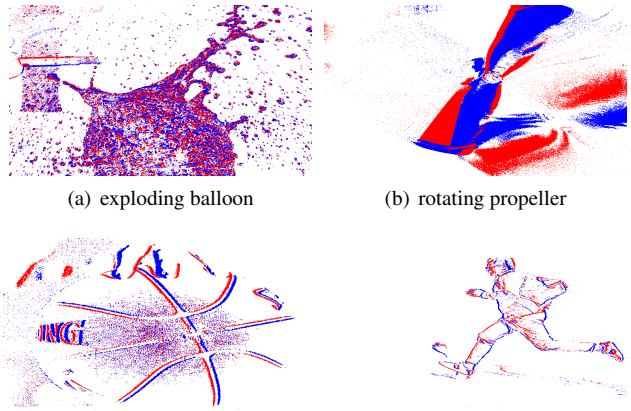
Algorithm 1 Event readout simulator

Input: consecutive front-end voltage maps $\{\mathbf{v}_{\text{FE}}(i), i = 1, 2, ..., n\}$ of a high speed video with frame rate f and frame size $N_{\text{cols}} \times N_{\text{rows}}$.

Initialization:

time step $\Delta t \in \mathbb{R}_+$
timeline of event generator $T_g \leftarrow \Delta t \in \mathbb{R}_+$
timeline of event scanner $T_s \leftarrow \Delta t \in \mathbb{R}_+$
event triggering time $\mathbf{t}_{\text{trg}} \leftarrow \mathbf{0} \in \mathbb{R}^{N_{\text{cols}} \times N_{\text{rows}}}$
reference time array $\mathbf{t}_{\text{ref}} \leftarrow \mathbf{0} \in \mathbb{R}^{N_{\text{cols}} \times N_{\text{rows}}}$
reference voltage array $\mathbf{v}_{\text{ref}} \leftarrow \mathbf{v}_{\text{FE}}(1) \in \mathbb{R}^{N_{\text{cols}} \times N_{\text{rows}}}$
instantaneous voltage array $\mathbf{v}_{\text{ins}} \leftarrow \mathbf{v}_{\text{FE}}(1) \in \mathbb{R}^{N_{\text{cols}} \times N_{\text{rows}}}$
voltage threshold $C \in \mathbb{R}_+$
row pointer in the scanner $R_s \leftarrow 0 \in \mathbb{N}$
row switching time $\Delta t_{\text{row}} \in \mathbb{R}_+$
scanning time per event $\Delta t_{\text{col}} \in \mathbb{R}_+$
refractory period $\Delta t_{\text{refrac}} \in \mathbb{R}_+$
$i \leftarrow 1$

1: **while** $i < n$ **do**
2: $i \leftarrow i + 1$ ▷ *load new frame*
3: **if** $\mathbf{t}_{\text{ref}} \in \left(\frac{i-2}{f}, \frac{i-1}{f}\right]$ **then** $\mathbf{v}_{\text{ref}} \leftarrow g[\mathbf{v}_{\text{FE}}(i), \mathbf{v}_{\text{FE}}(i-1), \mathbf{t}_{\text{ref}}]$
4: **while** $T_g \leq \frac{i-1}{f}$ **or** $T_s \leq \frac{i-1}{f}$ **do**
5: **if** $T_g \leq T_s$ **then** ▷ *event generator*
6: $\mathbf{v}_{\text{ins}} \leftarrow g[\mathbf{v}_{\text{FE}}(i), \mathbf{v}_{\text{FE}}(i-1), T_g]$
7: **mask** $=$ **if**$|\mathbf{v}_{\text{ins}} - \mathbf{v}_{\text{ref}}| > C$ **&** $\mathbf{t}_{\text{ref}} < T_g$ **&** $\mathbf{t}_{\text{trg}} = 0$
8: $\mathbf{t}_{\text{trg}}[\textbf{mask}] \leftarrow T_g$ ▷ *event fired*
9: $T_g \leftarrow T_g + \Delta t$
10: **else** ▷ *event scanner*
11: **if** $||\mathbf{t}_{\text{trg}}||_0^0 > 0$ **then**
12: **while** true **do**
13: $R_s \leftarrow$ the next row with events
14: write events (x,y,t,p) on R_s to file
15: $\mathbf{t}_{\text{ref}}[R_s] \leftarrow T_s + \Delta t_{\text{refrac}}$
16: **if** $\mathbf{t}_{\text{ref}}[R_s] \in \left(\frac{i-2}{f}, \frac{i-1}{f}\right]$ **then**
17: $\mathbf{v}_{\text{ref}}[R_s] \leftarrow g[\mathbf{v}_{\text{FE}}(i), \mathbf{v}_{\text{FE}}(i-1), \mathbf{t}_{\text{ref}}[R_s]]$
18: $T_s \leftarrow T_s + \Delta t_{\text{col}} \cdot ||\mathbf{t}_{\text{trg}}[R_s]||_0^0 + \Delta t_{\text{row}}$
19: $\mathbf{t}_{\text{trg}}[R_s] \leftarrow \mathbf{0}$
20: **if** $T_s \geq T_g$ **then**
21: **break**
22: **if** $||\mathbf{t}_{\text{trg}}||_0^0 = 0$ **then**
23: $T_s \leftarrow T_g$
24: **break**
25: **else**
26: $T_s \leftarrow T_g$



(a) exploding balloon (b) rotating propeller

(c) spinning basketball (d) running people

Figure 8. *Generated event frames with different input videos. Each frame is formed by accumulating events of* 1 ms *(red: positive event, blue: negative event).*

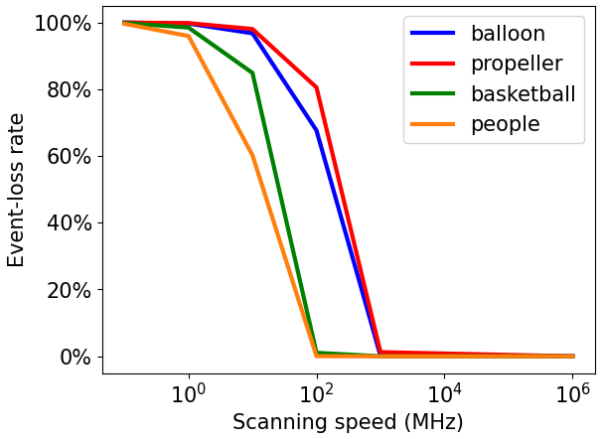


Figure 9. *Event-loss rate VS. scanning speed.*

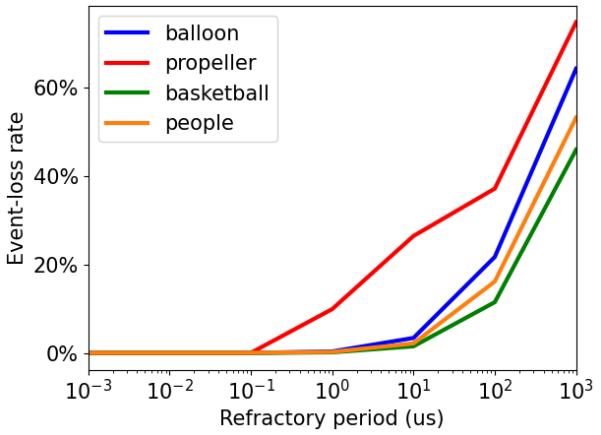


Figure 10. *Event-loss rate VS. refractory period.*

Results

We evaluated the impact of the scanner as well as the refractory period on event rate and readout latency using the proposed simulator on our own dataset.

Dataset

The testing dataset includes 4 videos recorded with a high speed camera "Photron NOVA S12" [15] at frame-rate of 10 kHz and image size 1024×1024 (exploding balloon, rotating propeller, spinning basketball, and running people). The experiments below were evaluated at a resolution of 960×540 and with a contrast threshold of 24 %. Fig. 8 shows sampled event frames from this dataset.

Event-Loss Rate

Due to refractory period or readout latency, a pixel may not yet be ready to trigger a new event shortly after one has already been registered. This causes an event-loss rate. In this section we evaluate event-loss with the impact factors of scanning speed and refractory period. The event-loss rate is computed by the total number of lost events due to the process of scanning or refractory period divided by the total number of triggered events when neither scanning nor refractory period are enabled.

Throughout the study of the scanning speed impact, the refractory period Δt_{refrac} was set to 0 and $\Delta\tau = \Delta t_{\text{col}} = \Delta t_{\text{row}}$. The scanning speed, here, is defined as $\frac{1}{\Delta\tau}$. Fig. 9 depicts the variation of the event-loss rate with regard to the scanning speed. At faster scanning speed the event-loss rate is reduced. Clearly, scenes with higher information density such as the exploding balloon and the rotating propeller exhibit increased loss-rates compared to moderately challenging scenes such as the spinning of a basketball which generates events only locally.

For the analysis of the impact of the refractory period, we set the column switching time Δt_{col} to 1 ns and the row switching time Δt_{row} to 80 ns. Fig. 10 shows the relation between the event-loss rate and the refractory period. Similarly, larger refractory period can also inhibit the event triggering. Also here, the exploding balloon and rotating propeller prove most challenging.

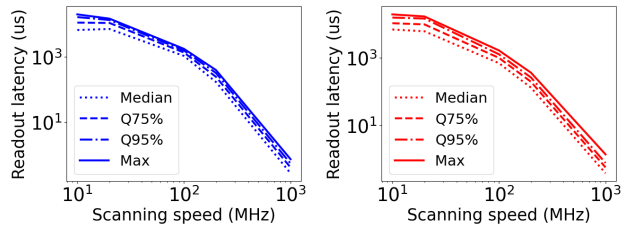
Readout Latency

The readout latency stands for the time interval between an event being triggered and being read, which can be formulated by $\Delta t_{\text{delay}} = t_{\text{scan}} - t_{\text{trg}}$, where Δt_{delay} represents the time delay, t_{scan} and t_{trg} are the timestamps when an event is being read and being triggered, respectively. We evaluated the readout latency for each event within the first 20 ms of the input videos.

Fig. 11 shows the change of the readout latency with regard to the scanning speed $\frac{1}{\tau}$. For each input video, we calculated the latency for each event and computed quantiles (median, $Q_{75\,\%}$, $Q_{95\,\%}$ and maximum). Note that in general it is not guaranteed that there is a simple monotonous relationship between increased readout speed and latency. This can be explained as follows. Consider that at a given time an event fires and coincidentally the event scanner is ready to read said event shortly after it is being fired. Increasing the scan speed now may cause the scanner to arrive at said pixel before the event is being fired. Thus a scan cycle has to be completed before the event can be read. This explains an increase of latency despite increased scan speed. Assuming a constant event-rate uniformly distributed in an array, the

latency, however, is expected to monotonously improve at faster scan speed.

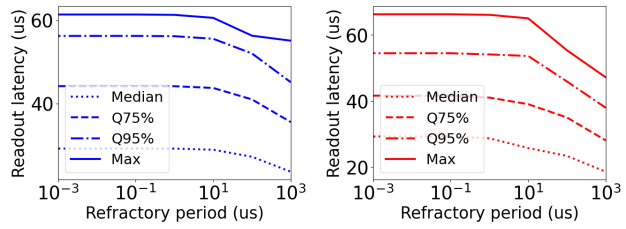
Fig. 12 evaluates the relation between the readout latency and the refractory period. Interestingly, an increase of refractory period can reduce the latency of events that are not lost. This is due to the excess event-loss rate at longer latency which reduces congestion of peripheral circuitry (as shown in Fig. 10).



(a) exploding balloon (b) rotating propeller

(c) spinning basketball (d) running people

Figure 11. *Readout latency VS. scanning speed ($\Delta t_{refrac} = 0$).*



(a) exploding balloon (b) rotating propeller

(c) spinning basketball (d) running people

Figure 12. *Readout latency VS. refractory period ($\Delta t_{col} = 1\,\text{ns}, \Delta t_{row} = 80\,\text{ns}$).*

Conclusion

The presented work offers several advancements compared to prior art. A physics based inverse model to estimate photocurrents has been shown to outperform luma based models - especially for spectra with stronger blue content. For the first time a latency model is presented for event-sensors that is calibrated and evaluated against circuit simulations. Moreover the presented work outlines the importance of the peripheral latency. A study has been presented highlighting the relationships between event-loss rates, latency and peripheral readout speed as well as refractory period. The acceptable loss-rates and latency are application specific and the presented simulator enables the derivation of sensor requirements to meet application targets.

References

[1] R. Benosman, "Event Computer Vision 10 years Assessment: Where We Came From, Where We Are and Where We Are Heading To," IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW) (2021).

[2] G. Gallego, T. Delbruck, G. Orchard, C. Bartolozzi, B. Taba, A. Censi, S. Leutenegger, A. Davidson, J. Conradt, K. Daniilidis, and D. Scaramuzza, "Event-based vision: A survey," IEEE Transactions on Pattern Analysis and Machine Intelligence 44(1), pp. 154-180, (2020).

[3] M. Mahowald, and C. Mead, "The silicon retina," Scientific American 264(5), pp. 76–83, (1991).

[4] H. E. Ryu, "Industrial DVS Design; Key Features and Applications," IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPR) (2019).

[5] T. Finateu, A. Niwa, D. Matolin, K. Tsuchimoto, A. Mascheroni, E. Reynaud, P. Mostafalu, F. Brady, L. Chotard, F. LeGoff, H. Takahashi, H. Wakabayashi, Y. Oike, and C. Posch, "A 1280x720 Back-Illuminated Stacked Temporal Contrast Event-Based Vision Sensor with 4.86um Pixels, 1.066GEPS Readout, Programmable Event-Rate Controller and Compressive Data-Formatting Pipeline," IEEE International Solid-State Circuits Conference (2020).

[6] T. Delbruck, Y. Hu, and Z. He, "v2e: From video frames to realistic DVS event camera streams," arXiv: 2006.07722 (2020).

[7] D. Gehrig, M. Gehrig, J. Hidalgo-Carrio, and D. Scaramuzza, "Video to Events: Recycling Video Datasets for Event Cameras," IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPR) (2020).

[8] Y. Hu, S.C. Liu, and T. Delbruck, "v2e: From video frames to realistic DVS event camera streams," IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW) (2021).

[9] H. Jiang, D. Sun, V. Jampani, M. Yang, E. Learned-Miller, and J. Kautz. "Super SloMo: High quality estimation of multiple intermediate frames for video interpolation". IEEE/CVF Conference on Computer Vision and Pattern Recognition (2018).

[10] Commission internationale de l'Eclairage proceedings. Cambridge University Press, Cambridge (1924).

[11] Bureau international des poids et mesures, "On the revision of the International System of Units (SI)", Proceedings of the 26th meeting of the General Conference on Weights and Measures (2018).

[12] P. Lichtensteiner, and T. Delbruck, "A 64x64 AER logarithmic temporal derivative silicon retina," Research in Microelectronics and Electronics, PhD (2005).

[13] K. Kundert, "Designer's Guide to SPICE and Spectre", Kluwer (1995).

[14] S. Foucart, and H. Rauhut, "A Mathematical Introduction to Compressive Sensing", Springer (2013).

[15] `https://photron.com/wp-content/uploads/2021/02/NOVA_4models_2021.02.10.pdf` (2021).