Event-Based Kilohertz Eye Tracking using Coded Differential Lighting

Timo Stoffregen tstoff@fb.com
Hossein Daraei daraei@fb.com
Clare Robinson clare.robinson@fb.com
Alexander Fix alexander.fix@fb.com

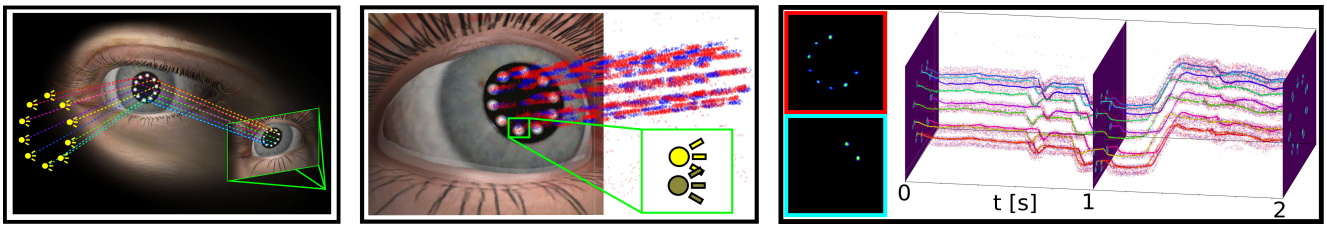


Figure 1: *Left*: Light sources cause specular reflections on the cornea, which appear as visible glints on an image sensor. These glints can be used to locate the corneal sphere for gaze estimation. *Center*: The light sources are pulsed in a binary sequence at high frequency. To prevent stimulating events across the entire scene, each light source is paired with a compensating light, so that the net illumination remains constant, but the specular cornea produces events. *Right:* The binary patterns in the event stream are decoded for kHz glint tracking. Red and cyan boxes show `0` and `1` bit glints for one bit sequence. The glint traces are shown in HSV color in this real-eye saccadic sequence, with events overlayed as red and blue points.

Abstract

Pixels in an event camera operate asynchronously and independently, reporting changes in intensity as events - tuples of (x, y) position, polarity s and timestamp t at microsecond resolution. Event cameras operate at low power ($\approx 5\,\mathrm{mW}$) and respond to changes in the scene with a latency on the order of microseconds. These properties make event cameras an exciting candidate for eye tracking sensors on mobile platforms such as Augmented/Virtual Reality (AR/VR) headsets, since these systems have hard real-time and power constraints. One proven method for eye tracking and gaze estimation is corneal glint detection. We exploit the fact that corneal glint tracking only requires a sparse set of pixels in the image, by making use of the natural sparsity of event cameras, which only detect changes in the scene. To enhance this effect, we design an illumination scheme, Coded Differential Lighting, which enhances specular reflections, suppresses all other events, and solves the light-to-glint correspondence. This is the first purely event-based corneal glint detection and tracking algorithm, which operates on standard hardware at kHz sampling rate.

1. Introduction

Instead of sampling all pixels at a fixed frame rate as in conventional cameras, the pixels of an event camera [17] independently report changes in $\log$ intensity. Events, represented as a tuple of (x, y) position, polarity s and timestamp t, trigger whenever the measured $\log$ intensity changes by more than a preset threshold. This allows event data to be efficient and sparse, since only scene changes are recorded. Event cameras have high dynamic range ($\approx 120\,\mathrm{dB}$), almost no motion blur, draw less power than conventional cameras, and report events at sub-millisecond latency [6].

Eye tracking is a key task in AR/VR headsets, facilitating user interactions, allowing for performance improvements through foveated rendering, and for eye-tracking analytics. State of the Art (SotA) display technologies such as Focal Surface Displays [13] or varifocal displays [15], also rely on eye tracking to determine the appropriate focal plane in real time.

Event cameras are a good fit for eye tracking sensors in AR/VR headsets, since they fulfil key requirements on power and latency. Head-Mounted Displays (HMDs) used in AR/VR must be low power, both to extend the battery life of mobile systems and to reduce the amount of heat generated by the headset. Further, eye tracking needs to operate at high sampling rate, to allow adaptive display technologies to operate seamlessly, and for applications like user authentication which can require up to $1\,\mathrm{kHz}$ sampling [12].

Many modern video based eye-tracking systems use Pupil Center Corneal Reflection (PCCR) [14]. This approach works by shining light sources (usually in the infrared spectrum) at the eye. This induces specular reflections, known as glints, on the surface of the cornea, which

can be detected by the camera as bright peaks and then used to estimate the position of the corneal sphere. At the same time the pupil, which appears as a dark ellipse, is detected. The gaze vector can be estimated by computing a vector between the centers of the pupil ellipse and corneal sphere.

We propose a novel event-based glint detection algorithm which is lightweight, operates at $1\,\mathrm{kHz}$ sampling rate, and efficiently solves the light-to-glint correspondence problem. By pulsing the illumination at high frequency the event camera produces events at the glint reflections, as desired. However, rapidly changing illumination also causes events in the rest of the image (skin, iris, sclera, etc.) which can exceed the event-rate of the camera and eliminate the power benefits of the sensor. We demonstrate that a new lighting scheme for event cameras, Coded Differential Lighting, preserves the events at specular reflections while suppressing events from diffuse parts of the scene. By using a compensatory paired-LED stimulus in which one light in the pair turns off as the other turns on, the net illumination remains approximately constant, while specular reflections move slightly. This enhances the glint signal while suppressing non-glint events.

While increasing the number of corneal glints improves gaze vector estimates [14], it introduces the challenging problem of robustly finding the correspondence between light sources and corneal glints. Our method works by pulsing the light sources for two known periods, with each period encoding either `1` or `0` bits. Each glint is identified through a unique binary pattern of these pulses. By frequency filtering the event stream, we not only remove unwanted sources of noise (such as events caused by changes in background lighting), but unambiguously identify each glint w.r.t. the corresponding light source.

Contributions

- The first fully event-based glint tracker, which is robust to background disturbances, uses only $\approx 35\,\mathrm{mW}$ of power, and has a sampling rate of $1\,\mathrm{kHz}$.
- Coded Differential Lighting, a novel dual-LED design which enhances event camera detection of specular reflections while suppressing non-specular background events, which we apply to corneal glint detection.
- A binary encoding scheme for Active LED Markers (ALMs) that supports arbitrarily many light sources.

In the following sections, we review literature on event sensors and eye tracking (Section 2), describe our proposed lighting, filtering and tracking approach (Section 3), and give an experimental evaluation of the method with a prototype hardware implementation (Section 4).

2. Literature

Model-based Eye Tracking Recent work in eye tracking has largely followed one of two approaches: *i)* 3D-model-based eye tracking, where image keypoints corresponding to geometrical features of the image are found and fitted to a 3D eye model using optimisation [7] and *ii)* appearance-based methods, in which the eye is tracked using the raw image of the eye, typically using CNNs or other ML models trained end-to-end to directly output gaze directions [21, 16, 10]. See [8] for a survey on eye tracking methods.

In this work, we follow the 3D-model-based tracking approach: specifically, we track the cornea of the eye using the reflection of a set of known illuminators (LEDs in our prototype) off the front surface of the cornea. Multiple of these reflections, called the *first Purkinje reflection*, or *glints* can be combined with a calibrated camera to estimate the location of the cornea relative to the camera. This is the one-camera, multiple-light sources case of Guestrin [7].

The advantage of model-based eye tracking over end-to-end methods is that model-based trackers can be quite precise. Glints can be estimated to sub-pixel accuracy in the image, and the resulting model fit can achieve $< 1^\circ$ of gaze tracking error [14]. However, compared to ML methods, model-based trackers can be sensitive to outliers, such as blinks and misidentified glints. By removing much of the non-glint signal from the image, we simplify glint detection and minimise the weaknesses of model-based trackers.

Event Based Eye Tracking The only previous work on event-based eye tracking (to our knowledge) combined events and standard frames from a hybrid event camera to perform eye tracking [1]. The frames were used to initialise parametric ellipse, parabola, and circle models to the pupil, eyelid, and a single corneal glint respectively. Events found within a preset distance of the parametric model estimates were then used to asynchronously update the models between frames. Only the pupil estimate was used to estimate the gaze vector. This approach has several key downsides compared to our method. First, it relies on frames (not provided by most SotA event sensors), which eliminates the key event sensor advantages of reduced motion blur and low power. Second, the method is fragile to sensor noise or external changes in brightness: since events used to update the parametric eye model are chosen based on a distance threshold, any unwanted events within that threshold threaten to update the model incorrectly. Our method improves on both of these points - we do not use frames at all and can therefore demonstrate the first purely event-based system. Since we directly filter the events from each glint out of a high frequency signal, our method is resistant to both sensor noise and spurious events caused by unwanted background motions or changes in lighting (Section 4.4).

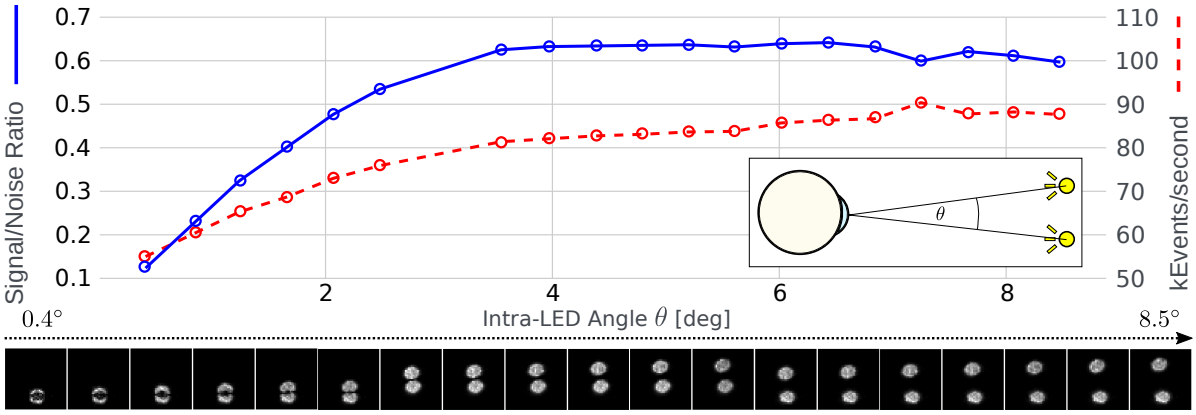


Figure 2: Various intra-LED spacings vs Signal-to-Noise Ratio (SNR) and event rate. Bottom row shows a close up image (formed by integrating events) of the corneal glints for each data point in the plot. Notice that the best SNR occurs at the point where the lower and upper glint begin to separate. For reference, an *unpaired* LED had a SNR of 0.018 and an event-rate of $875\,\mathrm{kev/s}$. Thus, dual LEDs represent substantial savings in spurious events.

Active LED Markers Tracking flashing reflected glints on the cornea is conceptually similar to tracking ALMs which has been done with event sensors for a variety of applications. Censi *et al*. [3] detected LEDs flashing at a fixed, known frequency to perform 3D localization of a drone. By using a lightweight frequency filter (see Section 3.2) for each LED frequency, they were able to track the drone at kHz sample rates. More recently, [4] used the same approach to create a local positioning system which achieves $3\,\mathrm{cm}$ accuracy when the camera is within $1\,\mathrm{m}$ of the light source. [5] used a set of five ALMs to localise LEDs embedded in a glove for hand gesture recognition. Finally, [9] present an event sensor model which they use to discover the high frequency characteristics of event sensors. The authors validate their model by improving the performance of ALMs. ALMs are also commonly used to calibrate event cameras, in products such as the *Propheshield* [19] or the Dynamic Vision Sensor (DVS) calibration software [20]. To achieve this, LEDs are arranged in a predetermined grid pattern and flashed at fixed frequencies. LED locations on the image sensor can be determined through frequency filtering, facilitating calibration.

We could find no examples in the literature of tracking more than four ALMs at $\geq 1\,\mathrm{kHz}$ frequency. We posit that this is due to bandwidth limitations of events sensors (Section 4.5), which motivates our introduction of binary codes to identify stimuli (allowing arbitrarily many ALMs).

3. Method

Our aim is to perform high sampling-rate, low power corneal sphere localization via glint detection with an event camera. Our method combines the following ideas: We produce glints on the cornea with flashing lights (beacons), whose arrangement is designed to enhance specular components of the image (Section 3.1). The events triggered by the beacons are filtered from the background and used to update calculated glint locations at high frequency (Section 3.2). Each beacon is flashed with a unique binary sequence of pulses (Section 3.3). By associating each calculated glint with a particular binary sequence, we find the beacon-to-glint correspondence and infer the position of the corneal sphere w.r.t. the camera on every pulse (Section 3.4).

3.1. Differential Lighting

We want to sample the eye's position at high temporal rate, ideally without relying on the motion of the scene to generate events. One strategy to accomplish this is to flash a light source at high rate. However, a scene illuminated by a flashing light source will typically saturate an event sensor, as brightness changes are reported across the entire scene. So, instead of flashing individual lights, we instead toggle pairs of nearby LEDs (that is, when one light toggles off the other toggles on), with the aim of keeping overall illumination roughly constant.

We make use of the fact that the eye is generally composed of two kinds of surfaces: specular surfaces like the cornea which produce a mirror-like reflection of the light sources; and (approximately) lambertian surfaces like the skin, iris, and sclera which scatter light diffusely. Using the language of BRDFs [2] to describe these surfaces, for a constant viewing angle θ_r, the amount of reflected light as a function of angle θ_i of incoming light for a diffuse surface is approximately $\cos(\theta_i)$, whereas a specular surface will have a sharp peak around $\theta_i = \theta_r$.

Notably, the derivative of the amount of reflected light w.r.t. changing angle of incoming light is much greater

on specular surfaces than diffuse surfaces. In our lighting scheme, when we toggle from one light of a pair being on to the other, we take a small step $\Delta\theta_i$ in the direction of incoming light for each pixel in the image, which produces a correspondingly large or small change in the amount of reflected light for specular or diffuse surfaces respectively.

Experimentally, we confirm that this allows us to reduce the number of events generated and increase the SNR by an order of magnitude (see Figure 2). The reflected glints are a few pixels wide, as opposed to being perfect point reflections, due to the nonzero size of the LED emitters and effects like optical blur. We find that the best intra-LED distance for the pairs of LEDs is at the point where the two glints just touch without overlapping. At lower intra-LED distances, the two glints cancel out where they intersect, causing a net brightness change of zero and reducing the number of 'signal' events at glint locations. Thus, glints at low spacings appear hollow (Figure 2). After the glints separate, the signal ceases to improve with further separation, but the amount of noise due to intra-LED distance induced brightness disparity increases.

There is nothing unique to the cornea in this analysis, except that it is a specular object-of-interest in an otherwise predominantly diffuse scene. We expect this differential illumination scheme to apply more generally to specular object imaging with event cameras.

3.2. Frequency Filtering

Cheap frequency filtering of the event stream is key to our method. Given a stream of events $e = \{(x, y), t, s\}$ from a set of events $\mathcal{E}$, we wish to locate the subset $\mathcal{E}_f$ on the image plane, produced by a beacon switching with period $T = \frac{1}{2f}$. Previous methods [3] detect the transition

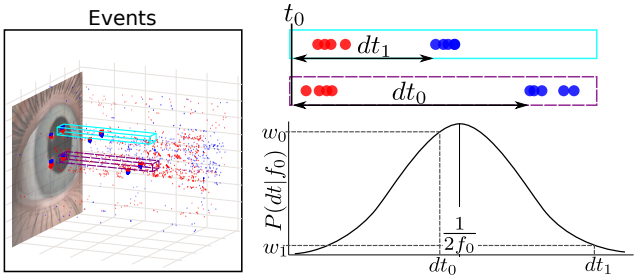


Figure 3: One bit of a four-LED bit sequence. Events in the cyan box are from a short `0` pulse, events in the purple box from a long `1` pulse. For an event in the purple box with time dt_0, the weight for the 0-bit filter, w_0, (equal to the normal pdf $P(dt_0|f_0)$) is much larger than the corresponding weight for the 1-bit filter, w_1, allowing per-pixel filtering of the `0` and `1` frequency bands. Note the delay between lights switching `ON` at t_0 and subsequent event generation.

period at each pixel dt by measuring the time between the first event of polarity s to the first event of opposite polarity $\bar{s}$. The likelihood (or weight) w of each dt being explained by the target frequency is modeled by a normal distribution:

$$w = P(dt|f) = \mathcal{N}(dt; \frac{1}{2f}, \sigma^2) \quad (1)$$

The standard deviation of the distribution σ is a tunable parameter which sets the bandwidth of the frequency filtering and is dependent on the properties of the event camera used. We found a value of 80 Hz to work well (see Section 4.5).

Leveraging electronic synchronisation between LEDs and camera, we modify the formulation for dt as the period between the synchronisation pulse t_0 and the first event of polarity s_p. This allows for more accurate filtering, since the variation in event timestamps from the initial LED state change is eliminated (see Figure 3), and provides a mechanism for separating glints from the primary and compensatory LEDs ($s_p = -1$ for primary and $s_p = 1$ for compensatory glints). For additional robustness to noise, we introduce a threshold such that at least λ_c=2 successive events of polarity s_p need to be detected to count as a transition.

The result is a Frequency Filter (FF) image, formed by summing the transition weights w at each pixel location.

3.3. Binary Glint Encoding and Tracking

Code One option to encode glint (or more generally, beacon) identity is to assign a unique frequency to each [3, 4, 5]. However, in the case of current SotA event cameras, this limits the number of glints that can be robustly tracked at ≥ 1 kHz to ≈ 5. This is because the actual transition periods implied by the event camera fall into a distribution that spans several hundred Hz, while frequencies above 2 kHz exceed sensor capabilities (see Section 4.5).

This motivates our introduction of a binary coding scheme, in which each LED flashes a unique binary sequence in which `0` is represented as a short pulse of period T_0 and `1` as a longer pulse of period T_1 (Figure 5). This allows us to support arbitrarily many beacons while only requiring two frequencies to be filtered, which the 1-2kHz band can easily support (Figure 3). This may appear to reduce the sampling-rate for each beacon, since $\log_2(N)$ bits are needed for N beacons; however, since we track each beacon over time, we can update the location on every bit, once the beacon tracker is initialised. The sampling-rate is equal to the clock frequency, 1 kHz in our case.

Tracking For an overview of the tracking algorithm see Figure 4. Using the events in each base clock period, two FF images for the `1` and `0` frequency bands (Section 3.2) and an event-image are formed. By masking the event image (which contains density information) with a FF image

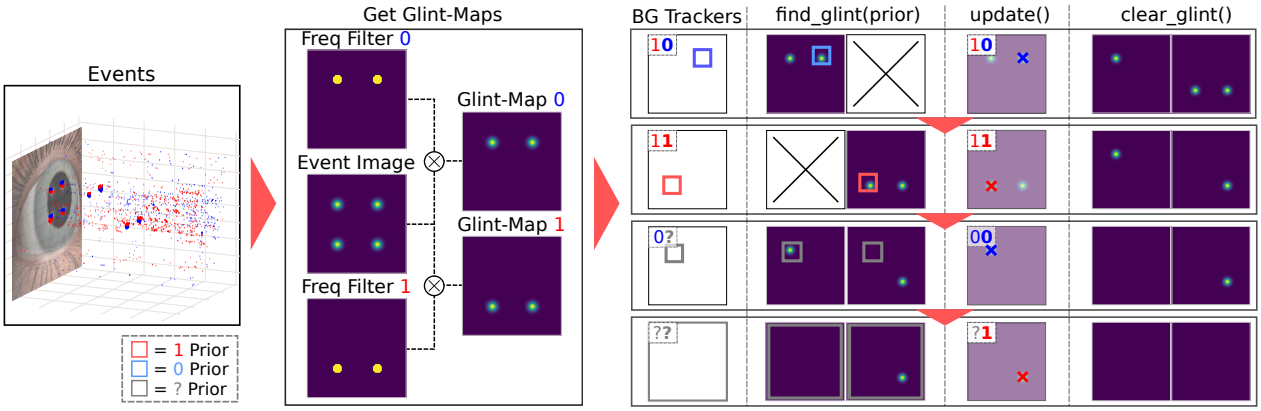


Figure 4: Our method takes as input the events from a single clock cycle: an initial burst of events as all primary LEDs turn on, then a burst as the 0 bit LEDs turn off, then a burst as the 1 bit LEDs turn off. These events feed into two frequency filters for 1 and 0 , which are used to mask an image of the events, giving the glint-maps. Fully initialised trackers (1 0) and (1 1) have a prior on location and bit; half initialised trackers (0 ?) have a prior on location and uninitialised trackers (? ?) have no prior. We iterate through the trackers in order of initialisation, searching the appropriate glint image/s and locations (based on prior) for glints. Located glints are used to update the trackers and then cleared from the glint-maps.

(which indicates which pixels belong to the desired frequency) we gain a glint-map, which contains the event density at the desired frequency. Since glints appear as Gaussian blobs, glint-maps allow more robust centroiding than the FFs.

Each glint is tracked by its own Binary Glint (BG) tracker object. A BG can be in one of 3 states. A *fully initialized* BG has both an expected location x (tracked from the previous clock) and an expected next symbol $b \in \{0, 1\}$ (the next sequentially in the binary pattern for this BG). When tracking is lost or at the beginning, BGs are *uninitialized*, with no prior position or symbol to track. Bright peaks in the event image not corresponding to an already tracked BG are assigned to uninitialized glints, at which point they become *semi-initialized*. Semi-initialized BGs have an expected location from tracking the peak, and fill in one bit of their binary pattern each clock according to the 0- and 1-bit

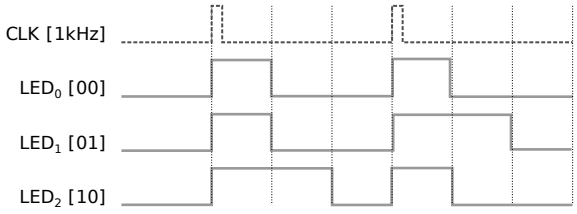


Figure 5: In our method, each LED is encoded by a unique binary code. To represent each bit, the base frequency f (1 kHz) is divided into three segments, with 1 represented by a long pulse of $\frac{2}{3f}$ s and 0 by a short pulse of $\frac{1}{3f}$ s.

FFs. Accordingly we do:

1. For each initialised BG, which tracks location x, bit b
   - Search glint-map b for peak in $\lambda_p \times \lambda_p$ patch around x and update x with the new peak.
2. For each semi-initialised BG, which tracks location x
   - Search $\lambda_p \times \lambda_p$ patch around x in both 0- and 1-maps. Append brighter bit to binary pattern.
3. For each uninitialised BG (tracking nothing)
   - Search entirety of both glint-maps for brightest glint, begin tracking as new x.

In these updates, the centroid of the detected glint is found as the weighted mean pixel coordinates μ_p of the patch and used to update the location and next bit of the BG. The patch around μ_p in the FF is then set to 0, preventing subsequent BGs from finding previously located glints. We considered using a constant-jerk, kinematic Kalman filter to track glints on the image plane, but found that this didn't improve results (Figure 7) relative to raw detections.

3.4. Corneal Sphere Regression

We model the the corneal glints as ideal specular reflections on a perfect sphere. Under these assumptions, there is exactly one ray which passes from each light source to the corresponding glint and from each glint to the corresponding point on the image plane (see Figure 1). Since the light locations in camera coordinates are fixed and known from device calibration, we can determine the corneal sphere dimensions and location by optimising the reprojection error of the lights onto the image plane. That is, we solve the

problem:

$$\boldsymbol{\theta}^* = \arg\max_{\boldsymbol{\theta}} \sum_{n=1}^{n_{\text{led}}} (\mathrm{x}_{rn} - \mathrm{x}_{en})^2 \tag{2}$$

where $\boldsymbol{\theta}$ is the optimization variable represented as a vector $[x_c, y_c, z_c, r_c]^\intercal$ of the cornea sphere position x_c, y_c, z_c and radius r in camera coordinates, n_{led} is the number of LEDs, x_{rn} is the reprojected location on the camera plane of LED n, and x_{en} is the detected location of the glint caused by LED n. We solve this problem using line-search based gradient decent using numeric derivatives.

4. Experiments

The following experiments are designed to demonstrate key aspects of our method and prove the claims made in the introduction. 4.1 demonstrates that our algorithm provides sub-pixel accurate glint estimates at rotational velocities far beyond the capabilities of human eyes. We show the performance of our method on real eye motions, as well as the model eyes used in other experiments. 4.2 demonstrates the frequency limits of our particular sensor and validates our choice of $1\,\mathrm{kHz}$ base frequency. 4.3 demonstrates the low power requirements of the camera and our setup. 4.4 shows that our method is robust against background sources of events and sensor noise. 4.5 illustrates the importance of our binary coding scheme by demonstrating the limitations of SotA one-frequency-per-beacon encoding.

Experiments were performed using a 640x480 resolution Prophesee EVK Gen3.1 event camera. The camera biases were tuned to suit our application; all figures quoting event rates need to be considered in this light. Biases are listed in Table 1. For the following experiments, 10 LED pairs were placed on an annular PCB (see Figure 6) around the camera lens ($16\,\mathrm{mm}$ f/2.8 C-Mount). Experiments involving ocular motions are capped to $1000\,^\circ\,\mathrm{s}^{-1}$, informed by the fact that human eye motions rarely go above $500\,^\circ\,\mathrm{s}^{-1}$ [11]. Unless otherwise indicated, we show results using raw glint detections *without* additional filtering

4.1. Detection Accuracy

In order to measure detection accuracy vs angular velocity, we place an eye model on a rotation stage and rotate it at various angular velocities. Ground truth glint positions are found by moving the eye model to discrete $1°$ intervals between $[-30°, 30°]$ and recording static sequences to form event images of the eye at each location. Glint locations

Table 1: Camera biases [18] used throughout experiments.

diff	_off	_on	fo	hpf	pr	refr
299	185	404	1438	1300	1250	1450

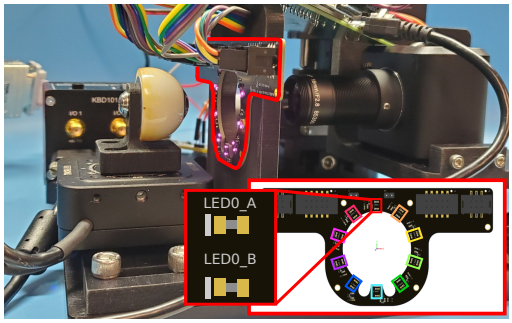


Figure 6: In our experimental setup, the event camera views the model eye through an annular arrangement of paired LEDs (intra-pair distance = $3\,\mathrm{mm}$). Each pair bordered by a different color, with closeup of one pair.

for these static locations are then interpolated using cubic splines, to fill the range between samples. As can be seen in Figure 7, our method is able to detect corneal glints with below $0.5\,\mathrm{pix}$ error even at the upper limit of human ocular motions. Results also show that raw detections are competitive with Kalman filtering of the detections, and raw detections are used in all other experiments.

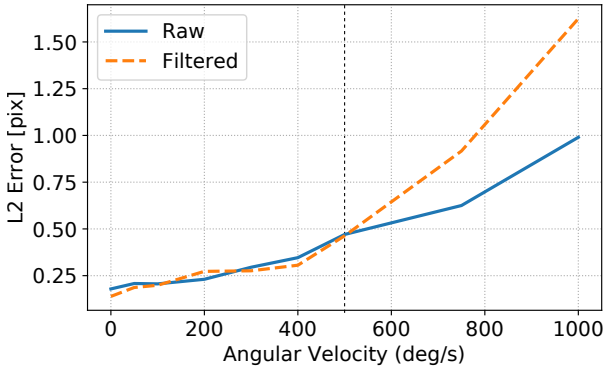


Figure 7: Mean of glint detection L2 error at various angular velocities of the model eye (raw and Kalman filtered). Vertical line indicates limit of human ocular velocity ($500\,^\circ/\mathrm{s}$).

Real Eye Motions To demonstrate the capabilities of our method on real eyes, we recorded sequences of both saccadic and smooth-pursuit motions from a human. The participant was seated in front of the device, with head stationary relative to the device, and asked to perform random saccades and smooth pursuits. Ground truth was collected by randomly selecting event frames from the event sequence and hand-labelling 100 sets of glint centroids (1000 labels overall) per sequence. We measured a mean L2 glint detection error of $0.342\,\mathrm{pix}$ for saccadic motions and $0.497\,\mathrm{pix}$

for smooth-pursuit. While these results reflect positively on our method, errors in the hand-annotated ground truth labelling may increase the size of the errors. For qualitative results of eye tracking at 1 kHz, please see the accompanying video.

4.2. Sampling Rate

The sampling-rate of our method is determined by the base clock frequency. As a result, a higher sampling-rate can be easily achieved by pulsing lights at higher frequencies. However, high frequencies may begin to exhaust the capabilities of the event sensor and its particular biases. At the limit, the sensor may produce few or no events at all. The trade-off between sampling-rate and glint detection accuracy is dependent on hardware and biases, however for our camera there is a sharp deterioration in detection accuracy above 1 kHz (Figure 8). Note that we were unable to find biases that would permit a faster sampling-rate.

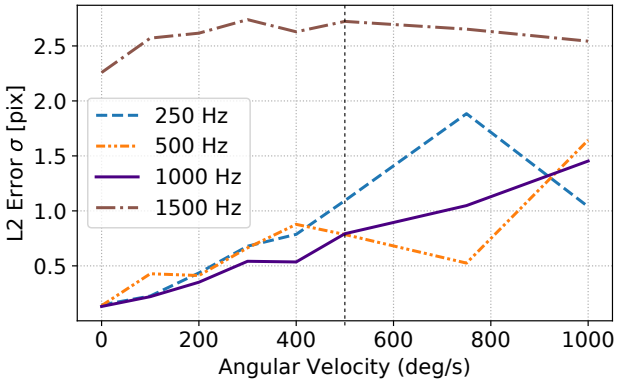


Figure 8: Mean glint detection L2 error [pixels] at various angular velocities of the model eye over a range of base frequencies. Detection error increases with the base frequency, since faster motion causes transitions to fail to generate sufficient events given the camera capabilities. Note that 1 kHz performs best - this is because the camera biases were tuned for this base frequency. In general though, higher base frequency results in lower latency and higher error. Vertical line indicates limit of human ocular velocity ($500\,°/s$).

4.3. Power Usage

Camera Power Low power consumption is a primary concern in many applications of near-eye gaze tracking. Event cameras typically consume less power than conventional cameras, with typical die-level power consumption around 10 mW and some prototypes achieving less than 10 µW [6]. The EVK Gen3.1 camera consumes 26 mW static power and has a dynamic consumption of $3\,\mathrm{nW/ev}$. By finding the event rate at various ocular velocities (Figure 9), we can use the power model of the event camera to determine power usage for our method. At the limit of human ocular motion ($\approx 500\,°/s$), sensor power usage for our method is ≈ 35 mW.

The power to process each event is dependent on the chosen processor hardware. The complexity of our method is dominated by the frequency filtering component, which is called millions of times per second. By contrast, glint detection is executed merely at 1 kHz. Frequency filtering requires around 3 Floating-Point Operations (FLOPs) per event on average (see supplementary material). On a modern CPU this would correspond to $\approx 0.5\,\mathrm{nW/ev}$ (≈ 1.75 mW total at $500\,°/s$ ocular velocity).

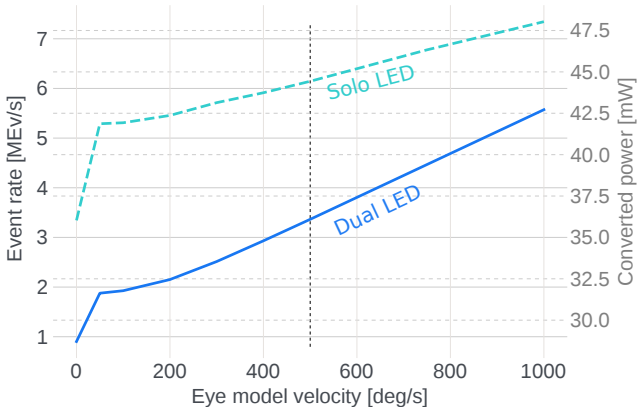


Figure 9: The event rate at various ocular angular velocities for both paired LEDs and a solo LED (dotted line). Conversion to power for our camera system via $P_c = 26 + 3\mathrm{e}{-6} \times \text{events}$ mW. Vertical line indicates limit of human ocular velocity ($500\,°/s$).

LED Power One means of reducing the power usage of our proposed method is reducing the current to the signal LEDs. To investigate this, we reduced the current to the LEDs and measured the incident optical power at the model eye as well as the glint detection error. The results in Figure 10 suggest that around 5 mW per LED are required for our method to work reliably, although we still achieve sub-pixel accuracy at lower power. Current LEDs are wide-angle, so better performance can likely be achieved through focusing and targeting LEDs at the eye.

4.4. Background and Noise Rejection

A limitation of event based sensors is that unexpected changes of brightness can produce many unwanted (spurious) events, which may cause errors in downstream tasks. Some examples are flickering halogen lights, PWM dimmed monitors, lens flare, or unmodelled camera/background motions. We claim that since we filter out these rela-

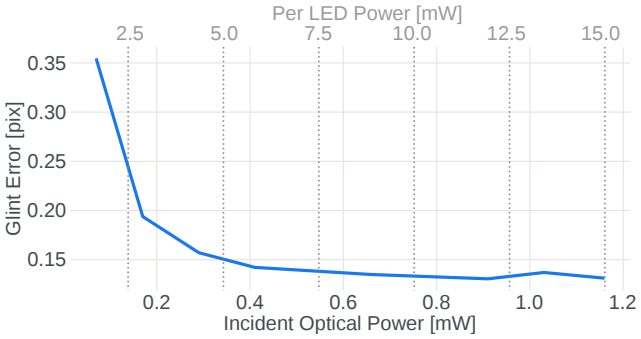


Figure 10: Incident optical and LED power vs. glint detection error. Optical power was measured using an 11 mm aperture optical power meter at the cornea.

tively low-frequency sources of spurious events, our method is unaffected by invasive light sources or facial movements which might induce spurious events in a HMD. We demonstrate this by recording the corneal glints of a realistic eye model embedded in a model head. A bright light source illuminates the scene at a fixed frequency, causing large brightness shifts in the surrounding eye and facial region being recorded. The results of this experiment in Table 2 show that our method retains sub-pixel accuracy even when scene noise dominates the signal.

Our method is also robust to sensor noise. In agreement with [3], we find that $> 99\,\%$ of sensor noise operates in the 0-250 Hz band, well outside the range of our frequency filters. For more details, see supplementary materials.

4.5. Bandwidth

A limiting factor of previous ALM approaches (which map each light to a unique frequency), is limited bandwidth, motivating our introduction of binary sequence encoding. Due to limitations of event camera hardware, there is variation in the response time of event camera pixels. Since these variations are independent of scene frequency, they have a larger effect on timing at higher frequencies (since they are larger relative to the smaller timing differences). This causes the observed transition periods to fall within a distribution D, whose support is the required bandwidth for

Table 2: Effect of background events generated by an external light source flashing at f_e Hz on the event rate (Mev/s), SNR and glint detection error (pix).

f_e [Hz]	1	5	10	20	50	100
Mev/s	5.19	5.20	5.21	5.61	11.3	23.2
SNR	5345	4611	401	8.83	0.63	0.18
Err [pix]	0.13	0.13	0.14	0.16	0.78	4.97

that frequency. Placing frequency filters too close together causes these distributions to overlap and therefore misidentification of frequencies in the scene. For example, Figure 11 shows the distribution of transition periods for a light flashing at 1 kHz, 1.25 kHz and 1.5 kHz. From the graph, transition periods implying 1.1 kHz are equally likely to be explained by a 1 kHz source as a 1.25 kHz pulse; the two frequency bands are too close together and overlap substantially. 1 kHz and 1.5 kHz overlap too, yet the large majority of transitions may be identified unambiguously.

It should be noted that the sampling function (Equation 1), reduces the required bandwidth, since it weights values that are far from the actual frequency as essentially zero. One way to think of this, is that the sampling function is multiplied with D to produce a distribution W with a smaller support ($W = \mathcal{N}(\frac{1}{2f}, \sigma_s) \times D$). However, this comes at the cost of throwing away information from D. For more on this, see the supplementary materials. We found $\sigma_s = 80$ Hz to be the smallest sample function standard deviation to give robust results, allowing ≈ 4 unique frequencies in the 1-2kHz band (in agreement with the literature). In contrast to the above discussion, our binary encoding scheme is unaffected by issues of limited bandwidth.

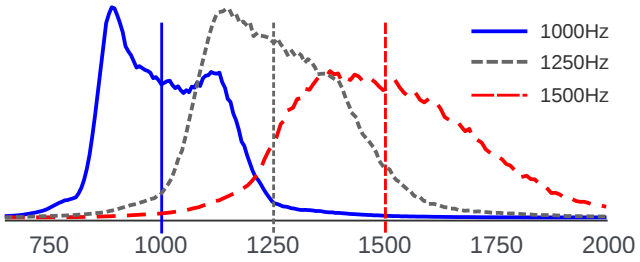


Figure 11: Histograms of the frequencies implied by the transition periods measured by an event camera observing a light flashing at 1 kHz, 1.25 kHz and 1.5 kHz.

5. Conclusion

In this paper we present a novel method for detecting corneal glints using an event camera. By pulsing the glint stimuli in binary patterns in the 1-2kHz range, we are able to achieve sampling-time of 1 ms on glint updates as well as constructing an unambiguous correspondence between stimuli and glint locations on the image plane. By placing glint stimuli in complementary pairs, we are able to counteract the saturation of the event buffer one might expect from recording flashing light sources on a scene. The result is a low-power, sub-pixel accurate corneal glint detector which robustly provides updates at kHz rates. By demonstrating our method both on controlled experiments as well as on real users, we hope to inspire the use of event sensors in actual eye tracking solutions.

References

[1] A.N. Angelopoulos, J.N.P. Martel, A.P.S. Kohli, J. Conradt, and G. Wetzstein. Event based, near-eye gaze tracking beyond 10,000Hz. *IEEE Transactions on Visualizations and Graphics*, 2021.

[2] FO Bartell, EL Dereniak, and WL Wolfe. The theory and measurement of bidirectional reflectance distribution function (brdf) and bidirectional transmittance distribution function (btdf). In *Radiation scattering in optical systems*, volume 257, pages 154–161. International Society for Optics and Photonics, 1981.

[3] Andrea Censi, Jonas Strubel, Chistian Brandli, Tobi Delbruck, and Davide Scaramuzza. Low-latency localization by active LED markers tracking using a dynamic vision sensor. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2013.

[4] Guang Chen, Wenkai Chen, Qianyi Yang, Zhongcong Xu, Longyu Yang, Jörg Conradt, and Alois Knoll. A novel visible light positioning system with event-based neuromorphic vision sensor. *sj*, 20(17):10211–10219, 2020.

[5] Guang Chen, Zhongcong Xu, Zhijun Li, Huajin Tang, Sanqing Qu, Kejia Ren, and Alois Knoll. A novel illumination-robust hand gesture recognition system with event-based neuromorphic vision sensor. *tase*, 18(2):508–520, 2021.

[6] Guillermo Gallego, Tobi Delbruck, Garrick Orchard, Chiara Bartolozzi, Brian Taba, Andrea Censi, Stefan Leutenegger, Andrew Davison, Jörg Conradt, Kostas Daniilidis, and Davide Scaramuzza. Event-based vision: A survey. *IEEE Transactions Pattern Analysis and Machine Intelligence (PAMI)*, 2020.

[7] E.D. Guestrin and M. Eizenman. General theory of remote gaze estimation using the pupil center and corneal reflections. *IEEE Transactions on Biomedical Engineering*, 53(6):1124–1133, 2006.

[8] Dan Witzner Hansen and Qiang Ji. In the eye of the beholder: A survey of models for eyes and gaze. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(3):478–500, 2010.

[9] Damien Joubert, Mathieu Hébert, Hubert Konik, and Christophe Lavergne. Characterization setup for event-based imagers applied to modulated light signal detection. *ao*, 2019.

[10] Petr Kellnhofer, Adria Recasens, Simon Stent, Wojciech Matusik, and Antonio Torralba. Gaze360: Physically unconstrained gaze estimation in the wild. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019.

[11] John R. Leigh and David S Zee. *The Neurology of Eye Movements*, page 111. Oxford University Press, Oxford, UK, 4th edition, 2007.

[12] Dillon J. Lohr, Henry K. Griffith, and Oleg V. Komogortsev. Eye know you: Metric learning for end-to-end biometric authentication using eye movements from a longitudinal dataset. *CoRR*, abs/2104.10489, 2021.

[13] Nathan Matsuda, Alexander Fix, and Douglas Lanman. Focal surface displays. *ACM Transactions on Graphics*, 36:1–14, July 2017.

[14] Clara Mestre, Josselin Gautier, and Jaume Pujol. Robust eye tracking based on multiple corneal reflections for clinical applications. *Journal of Biomedical Optics*, 23(3):1 – 9, Mar. 2018.

[15] Nitish Padmanaban, Robert Konrad, Tal Stramer, Emily A. Cooper, and Gordon Wetzstein. Optimizing virtual reality for all users through gaze-contingent and adaptive focus displays. *Proceedings of the National Academy of Sciences*, 114(9):2183–2188, 2017.

[16] Cristina Palmero, Javier Selva, Mohammad Bagheri, Mohammadali Ca, and Sergio Escalera. Recurrent cnn for 3d gaze estimation using appearance and shape cues. 09 2018.

[17] Christoph Posch, Teresa Serrano-Gotarredona, Bernabe Linares-Barranco, and Tobi Delbruck. Retinomorphic event-based vision sensors: Bioinspired cameras with spiking output. *Proceedings of the IEEE*, 102(10):1470–1484, Oct. 2014.

[18] Prophesee.ai. Biases manual. `https://docs.prophesee.ai/stable/hw/manuals/biases.html`. Accessed: 2021.05.04.

[19] Prophesee.ai. Propheshield description. `https://docs.prophesee.ai/stable/metavision_sdk/tutorials/Blinking_lights_detector.html`. Accessed: 2021.07.29.

[20] Robotics and Perception Group. Rpg dvs calibration toolkit. `https://github.com/uzh-rpg/rpg_dvs_ros/tree/master/dvs_calibration`. Accessed: 2021.07.29.

[21] Xucong Zhang, Yusuke Sugano, Mario Fritz, and Andreas Bulling. Appearance-based gaze estimation in the wild. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4511–4520, 2015.

A. Supplementary Materials

A.1. Unique Frequency Encoding

The simplest method of associating glints with their respective light sources is using a unique frequency per light. This is the method used in previous works on ALMs and is illustrated in Algorithm 1. We propose integrating the weight of each event transition w by a lowpass filter, in a similar manner to lowpass image reconstruction [22]. Essentially, each time a transition is registered, the previous value in the filter is decayed proportionally to the time since the last update, via the update formula

$$E(w_n) = E(w_{n-1})k + P(dt|f) \tag{3}$$

$$k = e^{-dt*f*\tau} \tag{4}$$

where τ is the time constant of the lowpass filter (cutoff frequency = $\frac{1}{\tau}$).

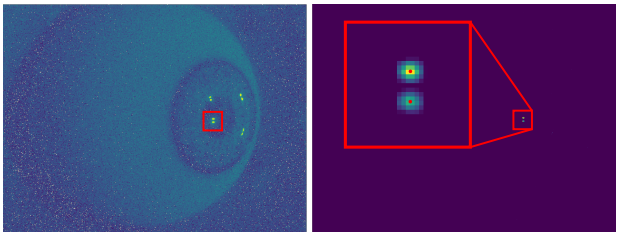
Frequency filter normalization In order to make our frequency filters more interpretable than a 'raw' expectation map as in [3], we can normalize the filter images using the ideal maximum value of the filter. Concretely, supposing that the event filter was observing an ideal pulse with frequency f, the update weight w for each pulse would be the mean likelihood of the sampling distribution $\mu_f = \mathcal{N}(0; 0, \sigma)$. Under this circumstance, the closed form definition for 3 is

$$E(n) = k^n E_0 + \frac{\mu_f(k^{n+1} - 1)}{k - 1}. \tag{5}$$

Algorithm 1: Frequency filtering algorithm (ec=event count, ts=timestamp, pol=polarity).

Input: Events $\mathcal{E}$, f, σ, λ_c, s_p
Output: I_f

```
forall e = {x, y, t, p} ∈ E do
    if p == curr_pol[x, y] then
        curr_ec[x, y] += 1;
    else
        next_ts[x, y] = t;
        next_pol[x, y] = p;
        next_ec[x, y] += 1;
    end
    if next_ec[x, y] > λc next_pol[x, y] == sp then
        dt=next_ts[x, y]- curr_ts[x, y];
        If[x, y] = N(dt; 1/2f, σ²);
        curr_pol[x, y] = p;
        curr_ec[x, y] = next_ec[x, y];
        curr_ts[x, y] = next_ts[x, y];
        next_pol[x, y], next_ec[x, y], next_ts[x, y] = 0;
    end
end
```



(a) Scene (b) Frequency Filter

Figure 12: In 12a four glint pairs flash at various frequencies. 12b shows the frequency filter response for the glints. A 2-mean GMM is applied to detect glint centers (red points).

This equation has a limit

$$\lim_{n\to\infty} E(n) = \frac{\mu_f}{1-k}, \tag{6}$$

which gives us the maximum value the filter can take. We divide by this maximum value to scale expectation maps to the range $[0, 1]$.

Glint centroiding Because the frequency filter doesn't rely on the synchronization pulse, it needs an additional step to distinguish the 2 glints within a glint pair (which operate at the same frequency, as they compensate each other). We use a 2-mean Gaussian Mixture Model (GMM) to find the centers of each glint pair in the expectation maps with subpixel accuracy (see Figure 12).

Operations per Event Note that the large majority of events never passes the `if` statement on line 9 of Algorithm 1. Only when an event of the *opposite* polarity to the current polarity is observed, is a transition registered and this condition triggered. For a single pulse of events, this should only occur *once* and only for those pixels observing the beacon stimulating the pulse. In our experiments, glints were always $\leq 120\,\mathrm{pix}$ in size. A typical pulse of events contained around $3500\,\mathrm{ev}$ at an upper bound of $500\,^\circ/\mathrm{s}$ ocular motion (see Figure 9, where an event-rate of $\approx 3.5\,\mathrm{Mev/s}$ is measured with 1000 pulses/s). Most of these events only require 3 FLOPs as they do not pass the second `if` statement, which is only passed $\approx$ 120 times. The second `if` statement requires 10 FLOPs, so on average $\approx \frac{3\times3400+10\times120}{3500} = 3.3$ FLOPs/ev

A.2. Binary Coded Glint Tracking Algorithm

A breakdown of the algorithm illustrated in 3.3 is presented in Algorithm 2.

Algorithm 2: Update algorithm for Binary Glint tracker (gm0=glint-map `0` , gm1=glint-map `1`).

Input: gm0, gm1, x, b, λ_p
if x == *none* **then**
  patch = `get_patch` (topleft=(0, 0), size=gm0.shape);
else
  patch = `get_patch` (center=x, size=(λ_p,λ_p));
end
if b == *none* **then**
  new_x_0, w0 = `find_glint` (gm0, patch);
  new_x_1, w1 = `find_glint` (gm1, patch);
  if *w0* > *w1* **then**
    return new_x_0;
  else
    return new_x_1;
  end
else
  if b == *0* **then**
    new_x, w = `find_glint` (gm0, patch);
    return new_x;
  else
    new_x, w = `find_glint` (gm1, patch);
    return new_x;
  end
end

A.3. Sensor Noise

We claim that our proposed method is resistant to sensor noise. To demonstrate this, we present a histogram of transition periods for a 60 s recording with the lens cap on (Figure 13). It is clear from this experiment that typical camera noise operates almost entirely in the 0-250 Hz range.

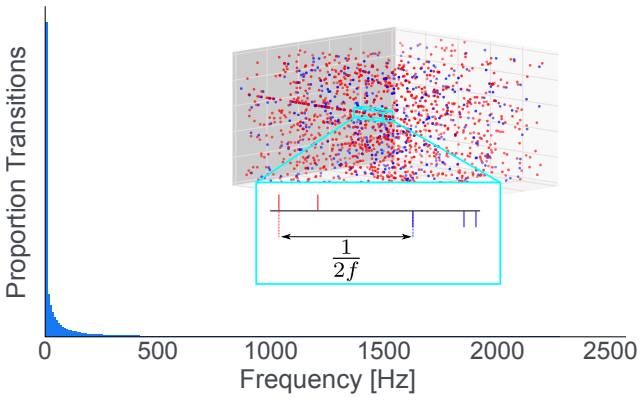


Figure 13: Dark blue bars show the histogram of the frequencies implied by the transitions of lens-cap noise events.

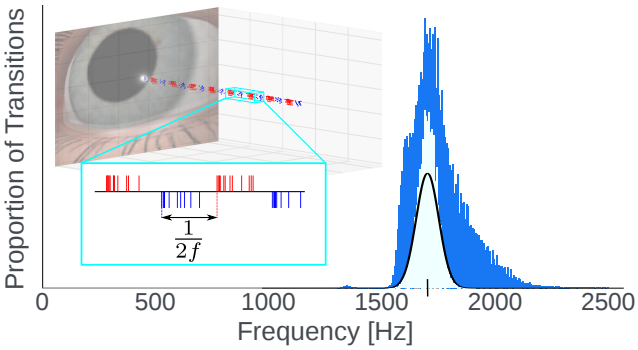
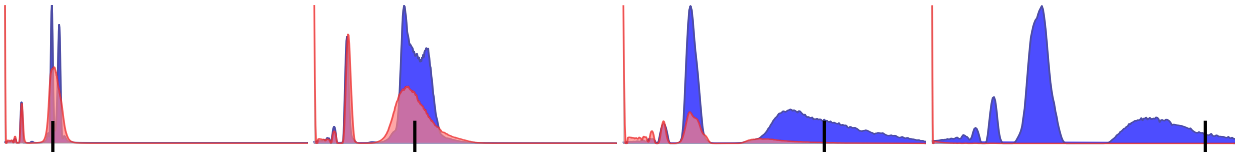


Figure 14: Dark blue bars show the histogram of the frequencies implied by the transitions of the event pulses of a light flashing at 1700 Hz for 10 s (mean $\mu_d = 1741$ Hz, stdev $\sigma_d = 129$ Hz). The sampling function (1) models this as a normal distribution $\mathcal{N}(\mu_s = 1700, \sigma_s = 80)$ (black curve). Multiplying the data with the sampling function reduces the bandwidth consumed (light blue), as $\sigma_s < \sigma_d$.

A.4. Event Camera Bandwidth

As identified in Section 4.5, the bandwidth required to robustly detect a beacon flashing at a fixed frequency is in the low hundreds of Hz for modern event sensors. This limits the number of beacons that can be robustly supported in a one-frequency-per-beacon encoding scheme to just ≈ 5. Figure 15 shows that event to achieve this, the target event sensors needs to be tuned for the task. The distribution of frequencies implied by the transition periods recorded observing a beacon flashing at a fixed frequency, shows that beyond 1 kHz the standard parameterisation fails entirely. Notice that even in our tuned camera, the peak of the distribution stalls at around 1800 Hz, implying that this is the limit of our camera's ability to accurately detect high frequency pulses. Also noteworthy is the smaller peak at the harmonic frequencies of the base frequency; since one 'missed' transition implies half the base frequency, and two 'missed' transitions imply on third the base frequency *etc*., there are peaks at these locations.

Effect of sampling function Since the frequency of the observed stimulus is estimated by inspection of the transition period between negative and positive events, variation in this period causes the recorded transitions to fall within a distribution D. This distribution is quite spread at higher frequencies, with σ in the low hundreds of Hz. Sampling with (1) is equal to a multiplication with D, giving a new distribution of weighted transitions: $W = \mathcal{N}(\frac{1}{2f}, \sigma_s) \times D$ (see Figure 14). Since σ_s is chosen to be less than the standard deviation of D, σ_d, the sampling distribution is the limiting factor that sets the bandwidth consumed by each



(a) 500 Hz (b) 1 kHz (c) 2 kHz (d) 3 kHz

Figure 15: Histograms of detected frequency (from 0 Hz to 3000 Hz) for light source flashing at 500 Hz, 1 kHz, 2 kHz, 3 kHz, with optimised biases (blue) and default biases (red). Black bars on the x axis denote the light frequency. Note the significant spike in detections at half of the target frequency - this occurs because a missed transition implies half of the frequency.

flashing stimulus, *i.e*. setting a small value for σ_s increases the available bandwidth. However, setting σ_s too small risks removing too much of D, which is the signal being measured. Therefore, there exists a tradeoff between available bandwidth and measurement accuracy.

The sampling distributions should not overlap much, since this introduces ambiguity, where the same transition can trigger a similar response in multiple frequency filters. This ultimately restricts the number of frequencies that can be supported on a given bandwidth.

References

[22] Cedric Scheerlinck, Nick Barnes, and Robert Mahony. Continuous-time intensity estimation using event cameras. In *Asian Conference on Computer Vision (ACCV)*, pages 308–324, Dec. 2018.