# Event-based Feature Tracking with Probabilistic Data Association

Alex Zihao Zhu, Nikolay Atanasov, and Kostas Daniilidis

***Abstract*— Asynchronous event-based sensors present new challenges in basic robot vision problems like feature tracking. The few existing approaches rely on grouping events into models and computing optical flow after assigning future events to those models. Such a hard commitment in data association attenuates the optical flow quality and causes shorter flow tracks. In this paper, we introduce a novel soft data association modeled with probabilities. The association probabilities are computed in an intertwined EM scheme with the optical flow computation that maximizes the expectation (marginalization) over all associations. In addition, to enable longer tracks we compute the affine deformation with respect to the initial point and use the resulting residual as a measure of persistence. The computed optical flow enables a varying temporal integration different for every feature and sized inversely proportional to the length of the flow. We show results in egomotion and very fast vehicle sequences and we show the superiority over standard frame-based cameras.**

## I. Introduction

Robot vision continues to primarily rely on the main paradigm of frame-based cameras that acquire whole frames with fixed time exposure and frame rate. An alternative camera paradigm has emerged recently, that captures at almost unlimited frame rate changes in intensity and records events at specific time-points and image locations. Such sensors like the DVS [1] or DAVIS [2] cameras enable tasks entailing very fast motions and high dynamic range. However, to facilitate these tasks we have to redefine basic vision problems like optical flow or feature tracking because of the lack of representations that could make use of fundamental assumptions like the brightness change constraint [3].

The fundamental challenge underlying event-based tracking is the lack of any data association between event and established features. Grouping the events and searching for the most similar event group is impossible because events are arriving asynchronously and grouping would require a time window specification. Without knowing the optical flow we are unable to assign a new event to the closest feature unless the flow is under one pixel. In this paper, we introduce a novel approach for feature definition by addressing the data association for each event to a feature as a hidden soft random variable. We regard the associations as probabilities because we do not need to make a hard commitment of an event to a feature. We apply an expectation-maximization (EM) scheme, where given optical flow we compute probabilities (weights) for data association and then we take the expectation over these probabilities in order to compute the optical flow. Inspired by the Good Features to Track approach [4] we model the alignment between events and features as an affine transformation and end the duration of the feature based on the quality of the alignment as well as

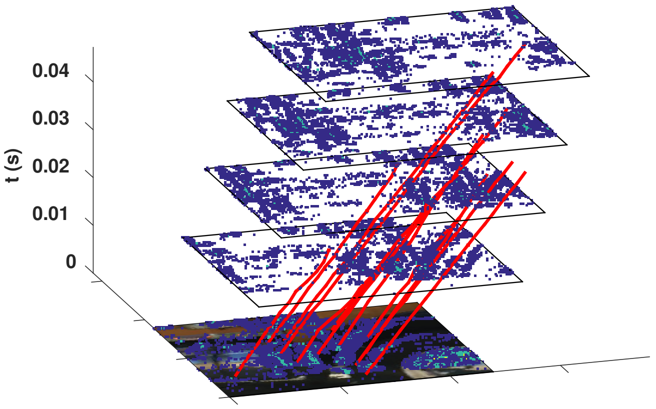

Fig. 1: Selected features tracked on a semi truck driving at 60 miles/hr, 3 meters from the camera. Intermediate images are generated by integrating events for a period equal to three times their lifetimes.

the convergence of the EM iteration. Grouping of the events into a feature is not by a fixed spatiotemporal window but rather by a lifetime defined by a fixed length of the optical flow computed in the previous steps.

We show in egomotion as well as in very fast motion scenarios that we can track robustly features over long trajectories. With this paper we make the following novel contributions to the state of the art of event-based tracking:

- Events are grouped into features based on lifetime defined by the length of the optical flow.
- Assignment of events to existing features is soft and computed as probability based on a predicted flow.
- Flow is computed as a maximization of the expectation over all data associations.
- Deformation of the feature is modeled as affine and the residual of the affine fit serves as a termination criterion

## II. Related Work

Litzenberger et al. [5], inspired by mean-shift tracking [6], create clusters of events by assigning events to the closest centroid. Each cluster is weighted by the mean frequency of the events and inactive clusters are eliminated. Kim et al. [7] estimate a 3D-rotation for the purpose of mosaicking by updating a particle filter with the likelihood of each new event given the current pose estimate. [8] propose an approach where an event is assigned to the spatially closest feature model and its euclidean transformation and scaling with respect to the model is computed. Initialization is achieved by fitting spatiotemporal planes to the event space, as in [9]. Lagorce et al. [10] define features using

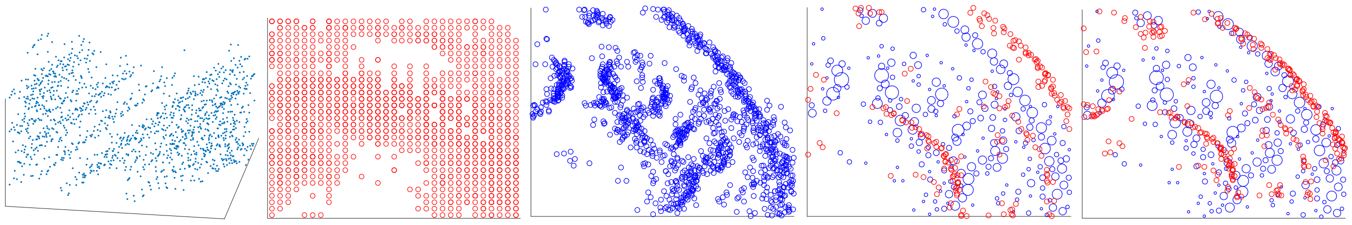
Fig. 2: Graphical outline of the algorithm. From left to right: 1. Event stream within the spatiotemporal window. Note the diagonal lines formed by the linear optical flow. 2. Events integrated directly onto the image with no flow correction. 3. Propagated events with estimated flow. Note the removal of the motion blur. 4. Later set of propagated events before affine warping. The size of the blue circles are the weights of each point after decimation. 5. Later set of propagated events after affine warping.

the Hough transform and then assign events using the ICP principle. Tschechne et al. [11] introduced the notion of a motion streak using biological vision principles where event tracks are detected by tuning spatiotemporal orientation over a longer temporal support. [12] and [13] were the first to combine a frame-based camera and event-based sensor on the same pixel-array for tracking. Using a corner and an edge detector this approach initializes a feature patch which is enhanced by new events that are registered using a 2D euclidean transformation. The commitment of an event to a feature is hard and hence the registration is prone to false event associations.

A common characteristic of the above approaches is the hard commitment, usually via ICP, to the assignment of an event to a model/feature with a subsequent estimate of a transformation given that commitment. Our approach integrates both data association and transformation estimation into a sound probabilistic scheme that makes it less prone to wrong correspondences. It does not make use of grayscale feature initializations. It is tested in very fast sequences where we show superiority over standard frame-based techniques. Barranco et al. [14] have created an evaluation dataset which offers ground truth optical flow but not longer feature trajectories. It provides self-motion estimates and we plan to use the dataset on the application of our feature tracking in visual odometry.

## III. Problem Formulation

### A. Sensor Model and Event Propagation

Let $F \in \mathbb{R}^3$ and $f(t) \in \mathbb{R}^2$ be the projection of $F$ onto the image plane at time $t$:

$$\begin{pmatrix} f(t) \\ 1 \end{pmatrix} \sim K \begin{bmatrix} R(t) & T(t) \end{bmatrix} \begin{pmatrix} F \\ 1 \end{pmatrix} \tag{1}$$

where $K$ is the camera calibration matrix and $\begin{bmatrix} R(t) & T(t) \end{bmatrix}$ is the camera pose. In the remainder, we refer to the projections $f(t)$ as *features* and consider a set of features $\mathcal{F}(t)$. Given a feature $f \in \mathcal{F}(0)$, define a spatial window $B(s) := \{x \in \mathbb{R}^2 \mid \|x - f\| < s\}$. Let $\{P_j \in \mathbb{R}^3\}_{j=1}^m$ be a set of 3-D points, whose projections $\{p_j(0)\}_{j=1}^m$ onto the image plane at time 0 are contained within the window $B(s)$. Let $\mathcal{P}^f(t)$ denote the set of point projections associated with feature $f \in \mathcal{F}(0)$ at time $t$. At discrete times $t_1, \dots, t_n$, the sensor generates a set of events $\{e_i := (x_i, t_i)\}_{i=1}^n$, where

$$x_i := p_{\pi(i)}(t_i) + \eta(t_i), \quad \eta(t_i) \sim \mathcal{N}(0, \Sigma), \quad \forall i$$

and $\pi : \{1, \dots, n\} \to \{1, \dots, m\}$ is an unknown many-to-one function representing the *data association* between the events $\{e_i\}$ and projections $\{p_j\}$ that generate them.

**Problem** (Event-based Feature Tracking)**.** *Given a set of events $\mathcal{E}$ generated by the point projections $\bigcup_{t=0}^T \bigcup_{f \in \mathcal{F}(0)} \mathcal{P}^f(t)$, estimate the feature projections $\mathcal{F}(t)$ in the image plane over time.*

## IV. Method

In Section IV-A, we introduce an optical flow based constraint within a spatiotemporal window. Section IV-B then shows that we can optimize this constraint over the optical flow using the Expectation Maximization algorithm. The resulting flow can then be used to reconstruct the set of point projections within the spatial window, which we then use in Section IV-C to refine the feature position using the EM-ICP algorithm [15]. Our tracking method then iterates between Section IV-B and Section IV-C to track a given set of features over time in the event stream. Section IV-D outlines our technique to select the size of the temporal windows in an asynchronous fashion, Section IV-E details our method for initializing the feature positions, and Section IV-F summarizes our method for estimating the image features within each window. The entire algorithm is also summarized in Algorithm 1 and Figure 2.

### A. Spatiotemporal Optical Flow Constraint

The motion of a feature $f(t) \in \mathcal{F}(t)$ in the image plane can be described using its *optical flow* $\dot{f}(t)$ as follows:

$$f(t) = f(0) + \int_0^t \dot{f}(s) ds = f(0) + t v(t), \tag{2}$$

where $v(t) := \frac{1}{t} \int_0^t \dot{f}(s) ds$ is the average flow of $f(0)$ over time. If $t$ is sufficiently small, we can **assume** that the average flow $v$ is constant and equal to the average flows of all point projections $\mathcal{P}(0)$ associated with $f(0)$. We can define a spatiotemporal window around $f(0)$ as the collection of events up to time $t$ that propagate backwards onto $B(s)$:

$$W(s,t) := \{e_i \mid t_i < t, x_i - t_i v \in B(s)\} \tag{3}$$

Thus, provided that $t$ is small, events corresponding to the same point in $\mathcal{P}(0)$ should propagate backwards onto the same image location. In other words, the following equality should hold for any pair $i, k \in [n]$ of events:

$$\|(x_i - t_i v) - (x_k - t_k v)\|^2 \mathbb{1}_{\{\pi(i)=\pi(k)=j\}} = 0, \quad \forall\, i, k \in [n] \tag{4}$$

However, since the data association $\pi$ between events and 3D points is unknown, we can hope to satisfy the above requirement only in expectation:

$$\begin{aligned} &\mathbb{E}_{\pi(i),\pi(k)} \|(x_i - t_i v) - (x_k - t_k v)\|^2 \mathbb{1}_{\{\pi(i)=\pi(k)=j\}} \\ &\quad = \left[\sum_{j=1}^{m} r_{ij} r_{kj}\right] \|(x_i - t_i v) - (x_k - t_k v)\|^2 = 0 \end{aligned} \tag{5}$$

where $r_{ij} := \mathbb{P}(\pi(i) = j)$ and we assume that $\pi(i)$ is independent of $\pi(k)$.

Given an affine transformation $(A, b)$ and the flow $v$ of feature $f(0)$, we model the noise in the event generation process by defining the probability that event $e_i$ was generated from point $p_j$ as proportional to the pdf $\phi(A(x_i - t_i v) + b; p_j, \Sigma)$ of a Normal distribution with mean $p_j$ and covariance $\Sigma$, i.e.,

$$r_{ij}(\{p_j\}) := \frac{\phi(A(x_i - t_i v) + b; p_j, \Sigma)}{\sum_{l=1}^{m} \phi(A(x_i - t_i v) + b; p_l, \Sigma)} \tag{6}$$

Where the argument $\{p_j\}$ is the set of points over which the means are defined. From here on, we will assume that $r_{ij}$ with no argument implies that the set is the point projections $\{p_j\}$. Note also that $\Sigma$ is a parameter to be experimentally tuned.

We propose an iterative approach to estimate the data assocation probabilities $r_{ij}$ between the events $\{e_i^f\}$ and points $\{p_j^f\}$, the affine transformation $A, b$, and the optical flow $v$ of feature $f$.

## B. EM Optical Flow Estimation

In this section, we propose an Expectation Maximization algorithm for solving (5) over a spatiotemporal window $W(s,t)$ with a set of events $\{e_i, i \in [1, n]\}$. Within this window, our optical flow constraint becomes

$$\min_v \sum_{i=1}^{n} \sum_{k=1}^{n} \left[\sum_{j=1}^{m} r_{ij} r_{kj}\right] \|(x_i - t_i v) - (x_k - t_k v)\|^2 \tag{7}$$

In the E step, we update the $r_{ij}$ and $r_{kj}$, given $v$ using (6). Initially, the set of point positions $\{p_j\}$ is unknown, and so we first approximate the $\{p_j\}$ by the set of propagated events $\{x_i - t_i v\}$. In general, $x_i - t_i v \to p_{\pi(i)}$ as $v \to v'$, where $v'$ is the true optical flow. In addition, as $A$ and $b$ are unknown, we initialize them as $A = I$ and $b = 0$. The full update, then, is $r_{ij}(\{e_i\})$.

The M step now involves solving for $v$ given the $r_{ij}$. As we assumed that the average optical flow $v$ is constant, (7) is a linear least squares problem in $v$, which corresponds to the general overdetermined system:

**Algorithm 1** Event-based Feature Tracking with Probabilistic Data Association

```
Initialization
  Initialize $\tau$ as $t'/k$ and integrate events for a short
    period of time over the image.
  Detect corner points using Harris corners on the
    integrated image, and initialize features $f_j$ at each
    corner.
Tracking
  Collect events for $k\tau$ seconds
  for each feature do
    $A \leftarrow I_2$, $b \leftarrow 0$, $v \leftarrow 0$, cost$\leftarrow \infty$, $\{p_j\} \leftarrow \{\}$
    while cost $> \epsilon$ do
      Find events within $W(s, k\tau)$ (3)
      Update $r_{ij}(\{p_j\})$ (6)
      Update $A$, $b$ (8)
      Calculate cost (7)
    end while
    Propagate events within the window to $t_0$ using $v$
    if $\{p_j\} = \{\}$ then
      $\{p_j\} \leftarrow$ propagated events
      continue
    end if
    cost$\leftarrow \infty$
    while cost $> \epsilon$ do
      Find events within $W(s, t)$ (3)
      Update $r_{ij}(\{p_i\})$ (6)
      Estimate $A$, $b$ and $\dot{x}$ using (11)
      Calculate cost (10)
    end while
    $\{p_j\} \leftarrow \{p_j\} - b + v \times k\tau$
  end for
  $\tau \leftarrow 1/\text{median}(\{\|v\|\})$
```

$$\begin{aligned} YD =& X \\ \text{where } Y :=& v^T \\ D :=& [\sqrt{w_{12}}(t_1 - t_2), \dots, \sqrt{w_{1n}}(t_1 - t_n), \dots, \\ & \sqrt{w_{n(n-1)}}(t_n - t_{n-1})] \\ X :=& [\sqrt{w_{12}}(x_1 - x_2), \dots, \sqrt{w_{1n}}(x_1 - x_n), \dots, \\ & \sqrt{w_{n(n-1)}}(x_n - x_{n-1})] \\ w_{ik} :=& \sum_{j=1}^{n} r_{ij} r_{kj} \end{aligned} \tag{8}$$

To get the normal equations, we multiply both sides on the right by $D^T$:

$$Y = (XD^T)(DD^T)^{-1} = \frac{\sum_{i=1}^{n} \sum_{k=1}^{n} w_{ik}(x_i - x_k)(t_i - t_k)}{\sum_{i=1}^{n} \sum_{k=1}^{n} w_{ik}(t_i - t_k)^2} \tag{9}$$

We iterate equations (6) and (8) until convergence of the error (4). As in [15], we reject outlier matches by thresholding the likelihood $w_{ik}$ when computing (8) by setting all the $w_{ik}$ higher than some threshold $\epsilon$ to 0.

## C. Feature Alignment

The flow estimate from Section IV-B can then be used to propagate the events within the window to a common time $t_0$. Given the correct flow, this set of propagated events is then the approximation to the projection of the points $P_j$ at time $t_0$, up to an affine transformation. As a result, given an estimate of the set of point projections at time $t_0$, $\{p_j := p_j(t_0) \in \mathbb{R}^2\}_{j=1}^m$, we can align the events with their corresponding points using a similar EM formulation as in Section IV-B. These estimated point projections are initialized on the first iteration, and the method is outlined in Section IV-F. The cost function for this alignment is that of the EM-ICP algorithm [15]:

$$\min_{A,b,r} \sum_{i=1}^{n} \sum_{j=1}^{m} r_{ij} \|A(x_i - t_i v) + b - p_j\|^2 \tag{10}$$

We can minimize this cost function using exactly the steps from Section IV-B. In the E step, we can use (6) to update $r_{ij}$.

The M step is also similar:

$$M : Y = (XD^T)(DD^T)^{-1} \tag{11}$$

where:

$$\begin{aligned} Y &:= \begin{bmatrix} A & b \end{bmatrix} \\ X &:= \begin{bmatrix} \sqrt{r_{11}} p_1, \dots, \sqrt{r_{1m}} p_m, \dots, \sqrt{r_{nm}} p_m \end{bmatrix} \\ D &:= \Bigg[ \sqrt{r_{11}} \begin{pmatrix} x_1 - t_1 v \\ 1 \end{pmatrix}, \dots, \sqrt{r_{1m}} \begin{pmatrix} x_1 - t_1 v \\ 1 \end{pmatrix}, \dots, \\ &\qquad \sqrt{r_{nm}} \begin{pmatrix} x_n - t_n v \\ 1 \end{pmatrix} \Bigg] \end{aligned}$$

As in Section IV-B, we iterate (6) and (11) until the error function (10) converges. We then use the new estimate for $b$ to refine the prior estimate for the image feature positions, and propagate them to the next window as:

$$f_j(t_n) = f_j(t_0) - b + v(t_n - t_0) \tag{12}$$

Similarly, the point projections are propagated to the next time:

$$p_j(t_n) = p_j(t_0) + v(t_n - t_0) \tag{13}$$

If the EM fails to converge, or if the value of (10) is too high after convergence, we consider the tracker lost and abandon the feature.

### D. Temporal Window Selection

Many event based techniques work with temporal windows of fixed size. However, these techniques have many similar drawbacks to traditional cameras, as the amount of information within the windows is highly variable depending on the optical flow within the image. Due to the quantization of the spatial domain, no information about the optical flow can be gained from the event stream until the projected points have moved at least one pixel within the image. On the other hand, too large of a window may violate the constant optical flow assumption made in Section IV-B. To ensure that the temporal windows are of an appropriate size, we dynamically adjust them using the concept of event 'lifetimes' [16]. Given the optical flow of a feature within a prior spatiotemporal window, we can estimate its lifetime $\tau$ as the expected time for the feature to move one pixel in the spatial domain:

$$\tau = \frac{1}{\|v\|} \tag{14}$$

For robustness against erroneous flow estimates, we estimate the lifetimes of several windows, and set the next temporal window size as $k$ times the median lifetime. In our experiments, we observed that $k = 3$ was a reasonable value to avoid large optical flow deviations while still capturing a sufficient number of events. This technique can be extended to have separate temporal window sizes for each tracked feature for fully asynchronous tracking between features. However, we found that, in our testing, the variation in optical flow of our features was not large enough to require this.

### E. Feature Selection

As this method relies on the assumption that the projected points are sparse, it will fail on spatial windows with dense points throughout. In addition, the matching scheme in Section IV-C suffers from the same aperture problem as traditional feature matching techniques. To avoid selecting such windows for our tracker, we propagate all events within each temporal window onto the image plane with zero flow to generate an integrated image. As events are typically generated over edges in the image, this integrated image is similar to an edge map. We then use the Harris corner detector [17] to select spatial windows with edge orientations in multiple directions.

### F. Point Set Generation

In order to perform the affine feature alignment step in Section IV-C, we must have an initial estimate of the set of point projections $\{p_j\}$. As the true point projections are unknown, we approximate them with the events in the first spatiotemporal window, propagated to the last time in the window, $T$, using the flow calculated in Section IV-B. This gives us a noisy set of points that approximate the true point projections, without motion blur. For each subsequent iteration, these points are propagated to the current time using (13), and aligned with the propagated events for that iteration. To reduce the computation time for matching against this potentially large feature set, we perform the sphere decimation algorithm in [15] to reduce the cardinality of this set.

## V. Experiments

We present the results of our approach using a DAVIS-240C sensor [2] in two situations. First, we compare the tracking accuracy of our tracking algorithm on a structured, textured area at normal speeds against traditional image based tracking on the frame-based intensity values from the DAVIS. We then demonstrate qualitative results of our algorithm on tracking a vehicle on a highway travelling at roughly 60 miles/hr, which we qualitatively compare to the tracking results on the 240FPS output of an iPhone 6.

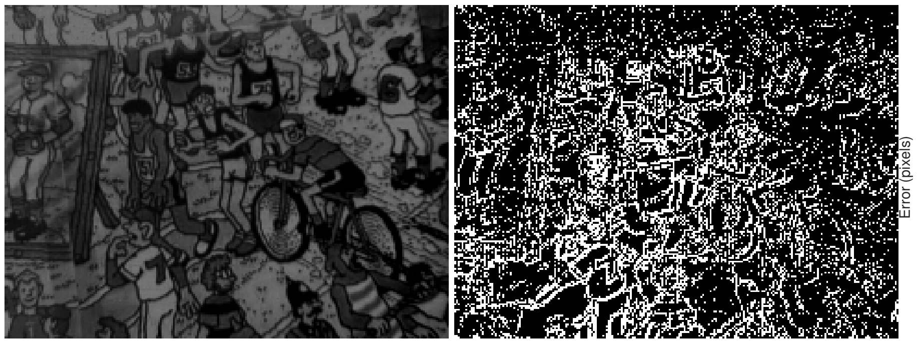
Fig. 3: Comparison between frame-based and integrated-event images.

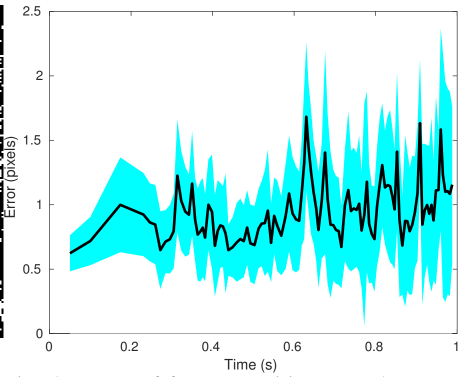

Fig. 4: Norm of feature position error between our method and KLT.

Fig. 5: Images of a truck driving on a highway recorded from the 240 FPS video.

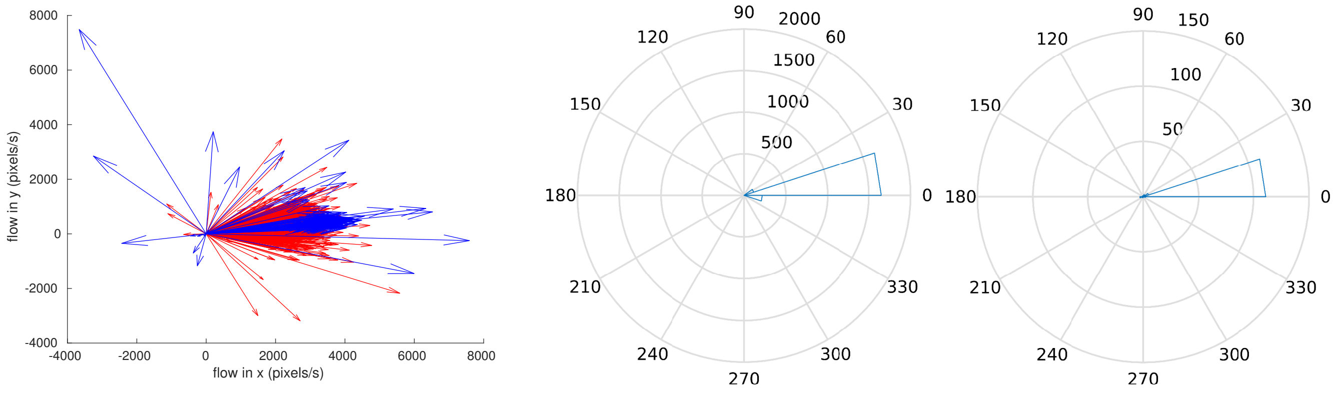

Fig. 6: From left to right: (1) Optical flow estimates from our method (red) and KLT tracking (blue), (2) Polar histogram (20 bins) of optical flow directions estimated by our method, (3) Polar histogram (20 bins) of optical flow directions estimated by KLT.

In each experiment, we used 31x31 pixel patches, with $\Sigma_j$ set to $2 \times I_2$. At the beginning of each sequence, a manually picked integration time is selected to start the algorithm to guarantee that the first temporal window contained significant apparent motion in the spatial domain. However, this time is very robust, and we found that any integration time where points moved at least 2 pixels was sufficient. In both experiments, 20 features were generated, with new features initialized if fewer than 12 remained.

### A. Comparison with Frame-Based Tracking

To quantitatively analyze the performance of our algorithm, we compare our results to the KLT Tracker [18] on a sequence where the DAVIS camera was moved in front of a textured surface (Fig. 3). Due to the relatively low frame rate of 25Hz for the frame based images on the DAVIS, this motion was restricted to relatively low speeds. Features were initialized from the integrated event image, and tracked in both the event stream as well as the frame based images until the majority of features were lost in both trackers. During the one second tracking period, the features moved on average 100 pixels.

We show the mean tracking error for features that have not been discarded in Fig. 4, where the black line is the mean tracking error over all the features, and the cyan region is one standard deviation around the mean error. As the event based measurements arrive much faster than the frame based ones, we interpolate the event based feature position estimates to the nearest frame based position estimate in time using the event based optical flow estimate. The overall mean error from this technique is 0.9492 pixels, which is comparable to the state of the art in this topic [12].

### B. Tracking on Scenes with High Apparent Motion

To test the algorithm on scenes with very high apparent motion, the camera was placed on the side of a highway with a speed limit of 60 miles per hour. Cars passed the camera at a distance between 3-4 meters, and passed the field of view in under 0.5s. We present here the results of tracking on a semi truck driving by at the posted speed limit. In this sequence, the average flow magnitude was 4000 pixels/s, and the (roughly) 15m long truck passed the camera's field of view in 800ms. The frame based images from the DAVIS sensor for these vehicles were almost completely blurred out. For comparison, we also recorded the scene with an iPhone 6 at 240 FPS (Fig. 5), on which we also ran a KLT tracker. The 240 FPS video is sufficient to capture the motion in this sequence, but is beginning to show motion blur on the order of one or two pixels. The two cameras' extrinsic parameters were estimated using stereo calibration. Unfortunately, due to the relatively close distance of the vehicles to the camera, we were unable to accurately warp the images onto one another for a quantitative comparison, and so we will instead give qualitative comparisons for our flow estimation based on a warp of the iPhone images assuming that points all have a depth of 3 meters.

We visualize a subset of the feature tracks in Fig. 1. It is interesting to note that, while the first integrated event image (superimposed over the iPhone image) has a significant amount of motion blur, the subsequent images have structures only a few pixels thick, due to the lifetime estimation from the optical flow.

In Fig. 6, we analyze the distribution of the direction of the optical flow vectors estimated by our method and by the KLT tracker. We can see that the majority of flow vectors lie between 0 and 20 degrees. This can also be seen in the left-most plot in Fig. 6, which shows individual flow vectors, with optical flow calculated within tracks shorter than 20ms removed. From these plots, we can see that both the direction and magnitude of the KLT flow vectors are very similar, although they should not perfectly correspond. For a visual comparison, we provide the tracking segment in our video accompanying this paper.

## VI. Conclusion

We have presented a novel approach for feature tracking in asynchronous event-based sensors that relies on probabilistic data association. Estimating optical flow becomes, thus, not sensitive to erroneous associations of new events and is computed from the expectation over all associations. To increase persistence of our tracks we compute the affine transformation for each feature with respect to the starting time. Existing approaches use a hard correspondence commitment and usually compute a similitude transformation. The spatiotemporal support of our features is adaptive and defined by the size of the flow rather than a fixed time or a number of events. We show that it outperforms classic KLT trackers when they are applied to 240 FPS cameras capturing very fast motions of the order of one field of view per half a second. We plan a real-time implementation so that we can apply it in real robot scenarios of visual odometry and moving object tracking.

## VII. Acknowledgement

We are grateful for support through the following grants: NSF-DGE-0966142 (IGERT), NSF-IIP-1439681 (I/UCRC), NSF-IIS-1426840, ARL MAST-CTA W911NF-08-2-0004, ARL RCTA W911NF-10-2-0016, ONR N00014-17-1-2093, and ONR STTR (Robotics Research), and the DARPA FLA program.

## References

[1] P. Lichtsteiner, C. Posch, and T. Delbruck, "A 128× 128 120 dB 15 $\mu$s latency asynchronous temporal contrast vision sensor," *IEEE journal of solid-state circuits*, vol. 43, no. 2, pp. 566–576, 2008.

[2] C. Brandli, R. Berner, M. Yang, S.-C. Liu, and T. Delbruck, "A 240× 180 130 dB 3 $\mu$s latency global shutter spatiotemporal vision sensor," *IEEE Journal of Solid-State Circuits*, vol. 49, no. 10, pp. 2333–2341, 2014.

[3] B. K. Horn and B. G. Schunck, "Determining optical flow," *Artificial intelligence*, vol. 17, no. 1-3, pp. 185–203, 1981.

[4] J. Shi and C. Tomasi, "Good features to track," in *Computer Vision and Pattern Recognition, 1994. Proceedings CVPR'94., 1994 IEEE Computer Society Conference on*, pp. 593–600, IEEE, 1994.

[5] M. Litzenberger, C. Posch, D. Bauer, A. Belbachir, P. Schon, B. Kohn, and H. Garn, "Embedded vision system for real-time object tracking using an asynchronous transient vision sensor," in *2006 IEEE 12th Digital Signal Processing Workshop & 4th IEEE Signal Processing Education Workshop*, pp. 173–178, IEEE, 2006.

[6] D. Comaniciu, V. Ramesh, and P. Meer, "Real-time tracking of non-rigid objects using mean shift," in *Computer Vision and Pattern Recognition, 2000. Proceedings. IEEE Conference on*, vol. 2, pp. 142–149, IEEE, 2000.

[7] H. Kim, A. Handa, R. Benosman, S.-H. Ieng, and A. J. Davison, "Simultaneous mosaicing and tracking with an event camera," *J. Solid State Circ*, vol. 43, pp. 566–576, 2008.

[8] Z. Ni, S.-H. Ieng, C. Posch, S. Régnier, and R. Benosman, "Visual tracking using neuromorphic asynchronous event-based cameras," *Neural computation*, 2015.

[9] R. Benosman, C. Clercq, X. Lagorce, S.-H. Ieng, and C. Bartolozzi, "Event-based visual flow," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 25, no. 2, pp. 407–417, 2014.

[10] X. Lagorce, C. Meyer, S.-H. Ieng, D. Filliat, and R. Benosman, "Asynchronous event-based multikernel algorithm for high-speed visual features tracking," *IEEE transactions on neural networks and learning systems*, vol. 26, no. 8, pp. 1710–1720, 2015.

[11] S. Tschechne, T. Brosch, R. Sailer, N. von Egloffstein, L. I. Abdul-Kreem, and H. Neumann, "On event-based motion detection and integration," in *Proceedings of the 8th International Conference on Bioinspired Information and Communications Technologies*, BICT '14, 2014.

[12] D. Tedaldi, G. Gallego, E. Mueggler, and D. Scaramuzza, "Feature detection and tracking with the dynamic and active-pixel vision sensor (davis)," in *Int. Conf. on Event-Based Control, Comm. and Signal Proc.(EBCCSP), Krakow, Poland*, 2016.

[13] B. Kueng, E. Mueggler, G. Gallego, and D. Scaramuzza, "Low-latency visual odometry using event-based feature tracks," in *IEEE/RSJ International Conference on Robotics and Intelligent Systems*, IEEE/RSJ, 2016.

[14] F. Barranco, C. Fermuller, Y. Aloimonos, and T. Delbruck, "A dataset for visual navigation with neuromorphic methods," *Frontiers in neuroscience*, vol. 10, 2016.

[15] S. Granger and X. Pennec, "Multi-scale em-icp: A fast and robust approach for surface registration," in *European Conference on Computer Vision*, pp. 418–432, Springer, 2002.

[16] E. Mueggler, C. Forster, N. Baumli, G. Gallego, and D. Scaramuzza, "Lifetime estimation of events from dynamic vision sensors," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4874–4881, IEEE, 2015.

[17] C. Harris and M. Stephens, "A combined corner and edge detector.," Citeseer, 1988.

[18] B. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision," in *Int. Joint Conf. on Artificial Intelligence (IJCAI)*, vol. 81, pp. 674–679, 1981.