Event-Based Laser Speckle Velocity Regression for Planar Motion

Philipp Walderich[1*], Philipp Werny[1], Thomas Kaster[1], Christian Hinke[1] and Constantin Häfner[1,2]

[1] *Chair for Laser Technology - RWTH Aachen University, Steinbachstr. 15 52074 Aachen, Germany*
[2] *Fraunhofer Institute for Laser Technology ILT, Steinbachstr. 15 52074 Aachen, Germany*
[*] *Corresponding author's e-mail: Philipp.walderich@llt.rwth-aachen.de*

Precise and high-frequency velocity measurement is essential for robotic laser material processing. Traditional vision-based methods, such as template matching, are limited by fixed frame rates and motion blur, reducing measurement accuracy. Event cameras offer a novel solution by capturing brightness changes asynchronously at the pixel level, focusing on salient features like edges. This makes them well-suited for capturing laser speckle patterns with improved temporal resolution and reduced data redundancy. This paper proposes a method for two-dimensional velocity estimation of laser speckle patterns using Spiking Neural Networks (SNNs), which naturally align with the asynchronous output of event cameras. The approach frames velocity estimation as a regression task, where the SNN is trained and evaluated in a simulated environment replicating 2D motion patterns. A dedicated event dataset based on hexapod motion with sub-millimeter precision is used, and results are validated against laser tracker-based reference measurements. Demonstrating a max. mean squared velocity error of 0.0019 mm/s at 1 mm/s command velocity across diverse motion profiles including linear, circular and abruptly changing trajectories. The findings highlight the potential of SNNs and event-based sensing for accurate, high-speed motion estimation in laser-based robotic applications and point to current limitations and future research tasks.

DOI: 10.2961/jlmn.2026.02.2005
Keywords: laser speckle, event camera, motion estimation, spiking neural networks

1. Introduction

The increasing complexity and diversity of production tasks, especially in high-wage economies, drives demand for advanced, flexible and low-cost automation solutions [1]. Laser Material Processing (LMP) holds promise for flexible, automated production using lower-cost robotic systems [2–4]. Yet, its precision is limited by relatively high motion tolerances of conventional robot guidance systems [2,5]. External metrology systems (e.g. laser trackers, iGPS, and vision-based setups) can monitor robot end-effector motion accurately but often provide limited update rates and are constrained by line-of-sight requirements. In contrast, laser speckle sensors can be integrated into the robot processing head and directly measure motion between the robot and the surface of the workpiece. Further, speckle patterns are highly sensitive to small displacements or motion. As such, Laser Speckle Imaging (LSI) has been successfully applied as non-contact laser speckle velocimetry solution to measure planar robotic tool speeds using image cross-correlation [6,7].

But commonly used camera chips (CCD or CMOS) capture full-frame intensity images at fixed frame rates. This limits temporal resolution and can reduce measurement accuracy at higher velocities due to motion blur and decorrelation of the speckle pattern in subsequent images. These drawbacks lead to a narrowing measurement range or forces a trade-off between accuracy and spatial resolution [7–9].
Unlike conventional frame-based sensors, event cameras operate asynchronously and pixel-wise. Each pixel independently reports an event once the perceived brightness change exceeds a predefined threshold. An event is described by a tuple (x, y, t, p), with the location of the pixel x and y where an event was triggered at time t. Further, the binary variable polarity p indicates positive or negative brightness changes. This new sensing paradigm results in data that emphasizes edges and their motion while ignoring static background information. Thereby, low-latency, high-temporal-resolution data can be achieved, depending on the observed image and motion. Accordingly, event cameras excel at capturing high-speed motion with drastically decreased motion blur while offering large dynamic ranges. These properties make event cameras particularly attractive for speckle motion estimation, where temporal resolution and contrast are important for signal quality [10].

Ge et al. [11,12] applied event cameras to speckle-based distance measurement during motion by reconstructing intensity images from event data and applying conventional correlation-based methods. While their approach achieved promising accuracy at low velocities (e.g., < 2 % error at 2.5 mm/s), it relied on temporal integration of event polarities to construct pseudo-frames, sacrificing the native temporal fidelity of the sensor. To fully benefit from the asynchronous output, data must be processed using specialized, non-frame-based processing algorithms [10].

Spiking Neural Networks (SNN) represent a biologically inspired class of neural models that are particularly well suited for processing the asynchronous, event-driven data produced by such cameras. In SNNs information is transmitted via discrete temporal events — spikes — rather than continuous numerical activations [13,14]. In neuron models such as the Leaky Integrate-and-Fire (LIF) neuron, membrane potential increases with incoming spikes and decays over time; a spike is emitted once the threshold is crossed, followed by a reset of the membrane potential [15]. This architecture aligns well with event-based data, preserving temporal dynamics and enabling full asynchronous, low-power

inference when deployed on neuromorphic hardware [14]. However, training SNNs remains difficult due to the non-differentiable nature of spike generation inside a neuron model. A common solution involves the use of surrogate gradients—smooth approximations of the spike function—to allow gradient-based optimization techniques [16].

Recent work [17–20] demonstrated that SNNs can successfully process event streams for continuous regression tasks using surrogate gradient-based learning. Various decoding strategies, such as rate or membrane potential decoding have been explored to achieve accurate motions estimation and efficient computation. However, increasing network depth and temporal resolution often introduces higher training complexity, requiring careful trade-offs depending on the application [17–20].

In this work, we present a novel approach combining speckle projection, event-based acquisition and a spiking regression network trained via surrogate gradients for planar velocity regression. A dataset was generated under controlled motion using a hexapod, with high-resolution ground truth provided by a laser tracker. The goal is to estimate continuous, axis-wise velocity components directly from asynchronous event streams of laser speckle patterns. Specifically, the suitability of laser speckle signals for event-based motion estimation is investigated. It examined how SNNs can handle speckle-derived data in a regression task and assessed which network architecture achieves highest accuracy in estimating 2d velocities. Further, variation of prediction errors over different motion regimes, including linear, abrupt changing and circular trajectories is analyzed.

2. Methods

Neuron model - Spiking neurons are used to model the neuron dynamics. The accumulation of incoming spikes does constitute the membrane potential of a neuron, which leads to the creation of a spike when a threshold is reached. Depending on the synaptic weight the impact of an incoming spike on a neuron's membrane potential is scaled while temporal decrease of the membrane potential is given by the decay factor. Various mathematical neuron models are described in the literature [13,14]. The simplest is the Leaky Integrate-and-Fire (LIF) model. To be able to retain memory over time and improve the network ability to model sequential dependencies a recurrent LIF (RLIF) neuron model was used in this approach. The RLIF model introduces a feedback mechanism where the membrane potential at any given time depends not only on the external input but also on the neuron's own past activity. The membrane potential $u_{\mathrm{t}}^{(\mathrm{l})}$ of a neuron at timestep t and layer l is updated as:

$$u_{\mathrm{t}}^{(\mathrm{l})} = \lambda u_{\mathrm{t-1}}^{(\mathrm{l})} + I_{\mathrm{t}}^{(\mathrm{l})} - S_{\mathrm{t-1}}^{(\mathrm{l})} + \vartheta S_{\mathrm{t-1}}^{(\mathrm{l})}. \quad (1)$$

Where $S_{t-1}^{(l)}$ is the binary spike output from the previous time step, λ is the decay factor and ϑ is a recurrent weight applied to previous output spikes, which modulates the neuron's response. $I_{\mathrm{t}}^{(\mathrm{l})}$ is the input current at time t [21].
The recurrent feedback can be interpreted as a form of self-regulation, where past spikes contribute to future neuronal behavior, allowing for improved stability in learning and improved learning for complex temporal dependencies [22].

Network architecture - The core architecture used in this work is a two-layer convolutional Spiking Neural Network (ConvSNN) similar to [17], which consists of two 2D convolutional layers. Each kernel has a size of 5×5 and a stride of 2 to progressively reduce spatial resolution while increasing feature abstraction. The first convolutional layer applies 16 channels, followed by 32 channels in the second layer. In contrast to [17] neurons are modeled as recurrent Leaky Integrate-and-Fire (RLIF) units with fixed membrane decay constants—0.7878 in the first layer and 0.3939 in the second—to balance temporal integration across the hierarchy. Spike thresholds were initialized with a base scaling factor and were learned independently for each layer during training, resulting in final thresholds of 0.9590 and 0.7132, respectively. In Table 1 the network structure is summarized.

Table 1 Hyperparameter Spiking Neural Network Architecture.

Layer	Conv 1	Conv 2	Fully connected Output
Kernel Size	5 x 5	5 x 5	-
Channel	16	32	-
Stride	2	2	-
λ	0.7878	0.3939	-
V_{th}	0.9590	0.7132	

The output of the second convolutional layer is aggregated via Global Average Spike Pooling (GASP) [15,17], which averages spike activity across spatial and channel dimensions. The pooled output $g_{\mathrm{i}}(t)$ for channel i at time t is defined as:

$$g_{\mathrm{i}}(t) = \sum_{\mathrm{x}\in\{0,\dots,\mathrm{W}-1\}} \sum_{\mathrm{y}\in\{0,\dots,\mathrm{H}-1\}} S_{\mathrm{i}}(t,x,y). \quad (2)$$

where $S_{\mathrm{i}}(t,x,y)$ denotes the binary spike output at position (x,y), time t, and channel i, and W,H are the spatial dimensions of the convolutional output. This pooled representation is then passed to a fully connected layer that outputs two continuous values corresponding to the planar velocity components of $v(t) = [v_{\mathrm{x}}, v_{\mathrm{y}}]$.

$$\boldsymbol{v}(t) = \frac{1}{N}(W[g_1 \dots g_{\mathrm{C}}]^{\mathrm{T}})(t). \quad (3)$$

Here, C denotes the number of channels in the final convolutional layer, $g_{\mathrm{i}}(t)$ the pooled spike rate per channel, and W the learned weight matrix of the decode. Successive synaptic weights are scaled by $N = 1/(W \cdot H)$ to ensure scale-invariance across input resolutions. [15,17]

Loss Function - The Mean Euclidean Error (MEE) was selected as the loss function to measure the discrepancy between predicted and reference velocities [17]. MEE is computed as follows:

$$\mathrm{MEE} = \frac{1}{T}\sum_{\mathrm{t=\tau}}^{\mathrm{T}} \|\hat{v}_{\mathrm{t}} - v_{\mathrm{t}}\|_2. \quad (4)$$

where $\hat{v}_{\mathrm{t}} \in \mathbb{R}^2$ denotes the predicted velocity at time t, and v_{t} the corresponding ground truth. The norm is taken in mm/s, making the error physically interpretable. To reduce the effect of transient signal behavior at the beginning of each training input, loss computation starts after a fixed time of $\tau = 20$ ms.

Training - One widely adopted solution to train SNNs is the use of Backpropagation Through Time (BPTT) with surrogate gradients [15,17,23]. The key idea behind surrogate gradient descent is to replace the non-differentiable Heaviside step function, which defines spike generation, with a differentiable approximation during the backward pass. Used surrogate function is the Sigmoid function:

$$\frac{\partial \tilde{S}}{\partial U} = \frac{1}{1+e^{\theta-U}}. \quad (5)$$

Allowing gradient-based optimization technique Adam to be used with a learning rate of 0.0078 and a batch size of 32. Early stopping was employed based on two independent criteria:

1. Training plateau detection: triggered if validation loss failed to improve over 25 epochs.
2. Output variance collapse: triggered if the standard deviation of predictions across a validation batch is below a predefined threshold ($\sigma < 0.01$), indicating that the network converged to a trivial solution.

Training input consisted of temporally segmented event sequences of 50 ms duration, with spatial dimensions of 80×80 pixels and two input channels for event polarities. Sequences are used for training at random with a seed of 14. Continuous-time SNNs need to be discretized to be simulated on GPUs. Each training sequence consists of 50 segments of accumulated event data aligned with the ground truth timestamp at a resolution of 1 ms.

Bayesian hyperparameter optimization was used to explore architectural and training parameter configurations, including learning rate, membrane decay constants, spike thresholds and network width.

In addition to the final two-layer ConvSNN, several alternative architectures were initially explored, including deeper convolutional variants with three and five layers, a fully connected spiking network (Forward SNN). To capture temporal dynamics better a recurrent LIF neuron model (RLIF) was applied to the Forward SNN; however, these configurations were not further optimized due to inferior performance or reduced training stability compared to the 2-layer ConvSNN.

3. Experiments

Experimental set-up - The experimental set-up used to create the training and evaluation data set is schematically shown in Fig. 1.

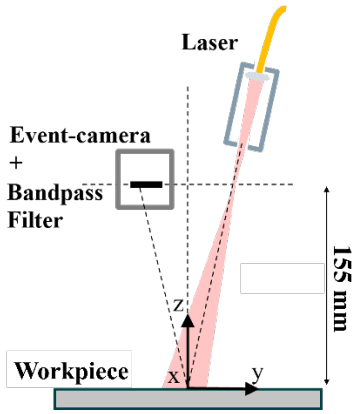

Fig. 1 Schematic depiction of experimental set-up.

Camera chip bandpass filter and laser are mounted together as one sensor head and positioned above the workpiece in a fixed in position. The focus point of the laser and the camera chip are positioned 155 mm above the surface leading to a laser spot diameter of 6 mm on the workpiece.

A Thorlabs CPS635R laser diode module, with a wavelength of 635 nm, a bandwidth of 15 nm and a max. output power of 1.2 mW is used. The bandpass filter FLH635-10 by Thorlabs Inc. with a central wavelength of 635 nm and a full width at half maximum of 10 nm is used to minimize the influence of surrounding light sources. The event camera used is DVXplorer by iniVation AG with a VGA resolution chip of 640x480 pixel and a pixel pitch of 9 µm. The readout ROI of interest is set to 80x80 pixel for all experiments. It offers a dynamic range of up to 110 dB, a temporal resolution of 200 µs and a maximum event throughput of 165 million events per second (MEPS). The workpiece metal plate is moved relative to the fixed laser and camera with a hexapod M-824.3DG by Physik Instrumente (PI) SE & Co. KG. The hexapod allows positioning in the x and y axes with a travel range of ±22.5 mm, while the z-axis travel range is ±12.5 mm. The maximum achievable velocity is 1 mm/s. A laser tracker (LT) Leica Absolute Tracker type AT930 from Hexagon Metrology GmbH is used to measure 3d position of the hexapod during motion. Position measurements are available at 1000 Hz measurement frequency and a maximum error of 21 µm. The coordinate system of the LT is initialized with accordance to the hexapod coordinate system. Event camera and laser tracker data capturing are synchronized with hexapod motion execution via an external square wave trigger signal from a function generator, which is controlled by a python script. Event camera and tracker data include the received trigger signal. That way time-synchronization of experimental data can be ensured by data preprocessing.

Experimental plan - To generate the data set two types of trajectories are introduced. The first trajectory is a collection of linear motion segments of random direction and constant velocity in the x-y-plane. Each linear motion segment is randomly sampled within the velocity and x-y-range of the hexapod. The goal is to create a data set that exposes the network to a wide variety of directions and velocities during the training and validation process. Data from linear motion is split in a training data set (90 %) and a validation data set (10 %).

The second trajectory is a laser processing-ready adaptation [5] of the ISO 9283 robotic accuracy trajectory. The trajectory is given in Fig. 2.

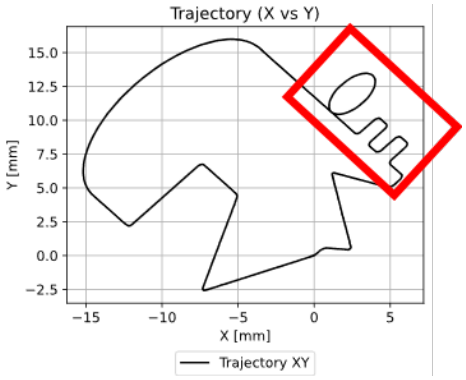

Fig. 2 Laser Tracker position data of ISO trajectory after 45° rotation.

Main features analyzed later in the paper are marked by the red bounding box. The trajectory is solely used to evaluate network estimation performance with regards to typical motion features as sharp edges, circular motion and axis-dependent accuracy. 100 % of data is used as a validation data set. Further, generalization capability of the network can be tested using a previously unknown trajectory.

Data pre-processing - To enable training and output evaluation of the network different pre-processing steps are applied to the recorded data set. The first step is time-synchronization of the recorded laser tracker and event data by extracting the data between the first and last recorded trigger for each trajectory. Then timestamps of LT and event data are normalized by subtracting the first from all subsequent timestamps. Further, position data is normalized by subtracting the initial position value from subsequent data to account for a potential relative offset between measured position and the coordinate system origin. Following, LT data was rotated by +45 ° around the z-axis of the LT coordinate system to align it with the x- and y-axis of the event camera chip.

Velocity calculation and filtering - Between each timestep of $\Delta t = 1$ ms between LT measurements constant velocity is assumed. Hence, reference velocity v_t at timestep t can be calculated by differentiation:

$$v_\mathrm{t} = \frac{x(t+\Delta t)-x(t)}{\Delta t}. \tag{6}$$

Where $x(t)$ denotes the position tuple in all three axis directions recorded by the LT at time t. Due to the positional uncertainty of the laser tracker $U_\mathrm{p} = 21$ µm high uncertainty in the calculated reference velocity U_v relative to the maximum velocity of the hexapod can be expected. Uncertainty can be calculated by error propagation under the assumption of equal uncertainty at each timestep:

$$U_\mathrm{v} = \frac{2U_\mathrm{p}}{\Delta t} = 42\ \mathrm{mm/s}. \tag{7}$$

Due to the constant velocity assumption, high frequent fluctuations of the velocity during motion are considered noise. To suppress high-frequency noise introduced by numerical differentiation of the laser tracker position data, a low-pass Butterworth filter was applied to the resulting velocity signal. This filter was chosen for its smooth frequency response and its ability to attenuate high-frequency components without distorting the underlying motion profile, making it well-suited for preserving the slowly varying dynamics of the reference trajectories.

Finally, reference velocities and event data are segmented by time into small batches to enable memory efficient simulation and training. Each segment covered a 100 ms time window corresponding to 100 timesteps with 1 ms resolution. The event data within each segment are collected into voxels by binning event data into 1 ms slices. Leading to tensors of shape (100, 2, H, W), where two channels represent the polarity, H x W spatial resolution of the region of interest. This conversion preserved the temporal structure while enabling input compatibility with spiking convolutional networks. Finally, each voxel event segment was paired with its corresponding sequence of reference velocity vectors derived from the laser tracker data. This alignment resulted in temporally synchronized input–target pairs suitable for supervised learning in SNNs.

4. Results

In Fig. 3 the accumulation of events over a 20 ms period resulting from the tracking of a laser speckle pattern during linear motion by means of an event camera is depicted.

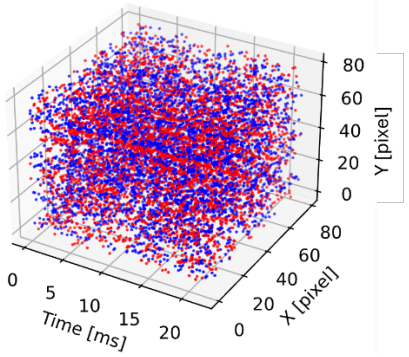

Fig. 3 Accumulated event data of 20 ms during linear motion experiment.

Positive (red) and negative events (blue) are observable in a dense distribution over the whole ROI of 80 x 80 pixel. Accumulations of events form traces in spatial-temporal space. Based on this event data, the SNN proposed in this paper is used to estimate axis-wise components of in-plane velocity.

In Fig. 4 the density of velocities of the reference data is depicted for each axis direction before and after filtering of the data.

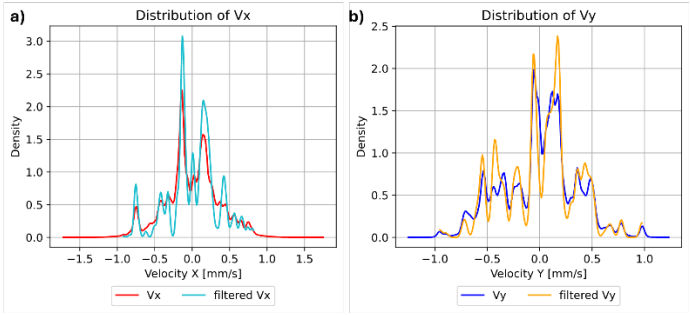

Fig. 4 Axis-wise density distribution of reference velocities before and after signal filtering, (a) velocity distribution in x-axis direction, (b) velocity distribution of y-axis direction.

A narrower density distribution is observable besides higher and more distinct peaks in the plot for the filtered signal. Further a higher density around central velocities compared to the edges of the distribution is visible for both axes.

In Fig. 5 the recorded number of events per second for different reference velocity bins is depicted.

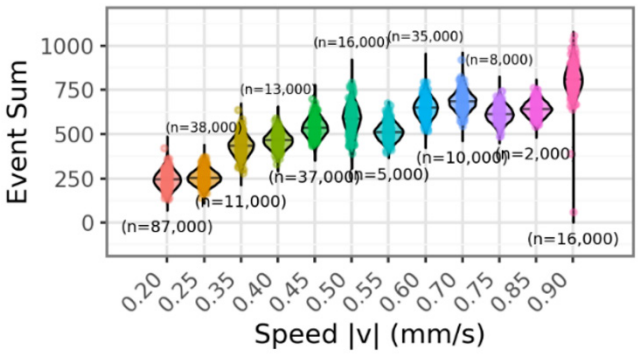

Fig. 5 Number of events per second over different reference velocities.

A correlation between the events/s recorded and an increasing reference velocity can be observed. But, for each distinct velocity bin depicted the events/s recorded are part of a distribution causing overlapping, ambiguous areas with higher and lower velocities in the plot. Further, at 0.9 mm/s the data shows large outliers, with event rates ranging from 0 to 1000 events/s. This behavior was not present in the original data and is expected to be introduced by the segmentation step during pre-processing.

Only about 1000 data points out of 2.4 million are among those visible outliers hence the influence on the model is expected to be neglectable. During a standstill measurement without any motion of the surface a mean noise of 116.8 events/s is present. In contrast at least 270000 events/s were recorded during motion at the lowest velocity. Meaning that the relative contribution of noise events to the signal is less than 0,05 %.

Linear training and evaluation data set - In Fig. 6 the training loss curve of the best model is given for 14 epochs.

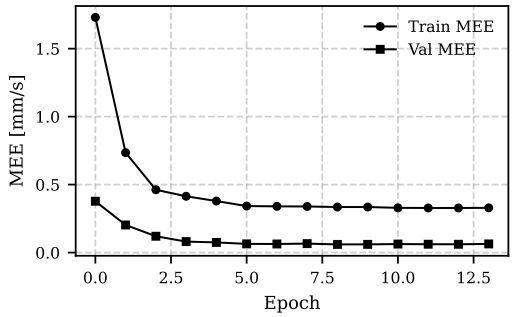

Fig. 6 Mean Euclidean error of training and validation data for each epoch used.

The training ended prematurely after 14 epochs as it reached a loss value with low change over multiple epochs, which is an early stopping criterion. A minimal training loss of 0.328 mm/s was reached. The validation loss remains lower than the training loss, but follows qualitatively the same trajectory with the fifth epoch marking the entrance to the lower limit mee.

In Fig. 7 the estimation results of the SNN on the validation data set over a period of 100 s with the according estimation error is depicted.

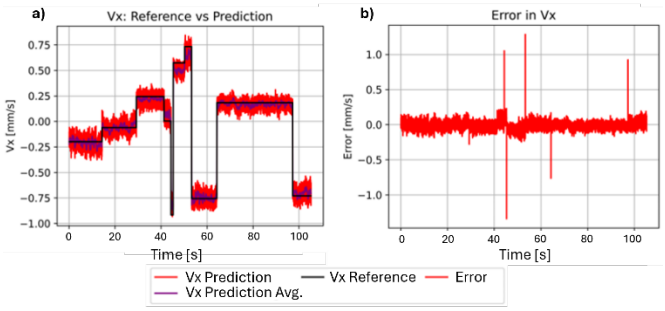

Fig. 7 SNN velocity estimation comparison against ground truth with (a) showing the predicted velocity against the reference and (b) depicting the resulting error over time.

In this period different patterns are observable which are similar for network output in y-axis direction. First, a relatively high noise compared to the available velocity range of 1 mm/s is observable with an error standard deviation of 0.365 mm/s (see Table 2), which is close to the minimal MEE training loss. Further, high transients between velocities occur jointly with high error peaks. While mostly, the estimation seems to oscillate around a mean close to the ground truth, a once reached offset after the transient is either constant or leads to further drift. But no correction of the error by the network is achieved over time. In Table 2 the mean absolute velocity error (MAE) and mean squared velocity error (MSE) with the error standard deviation are given.

Table 2 Axis wise MAE, MSE and Standard Deviation of SNN velocity output.

	MAE [mm/s]	MSE [mm/s]	Std. Dev [mm/s]
x-axis	0.0454	0.0035	0.365
y-axis	0.0353	0.0022	

Mean errors are relatively low compared to standard deviation of the error further indicating a strong oscillation around a lower constant deviation from the reference. The standard deviation of the filtered reference velocity signal with 0.3750 mm/s is even slightly higher. Error shows axis dependency with higher errors for velocity components in x-direction. Lower MSE than MAE further shows a low influence of outliers on the error statistics.

In total, a pearson coefficient of 0.9833 shows a high commonality between model estimation and reference values.

In Fig. 8 the network output at abrupt changes of velocity for the x-axis component is depicted in detail and in comparison, to the reference velocity and a low-pass filter smoothed output of the network.

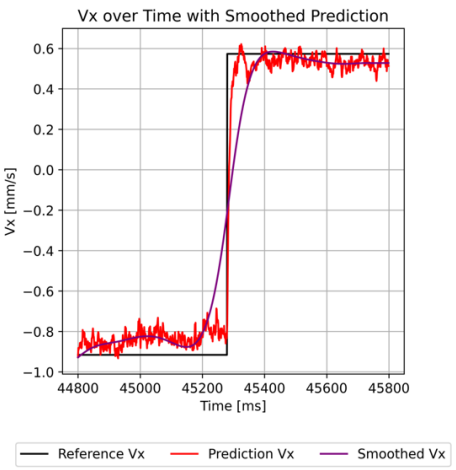

Fig. 8 SNN velocity estimation against ground truth zoomed transient.

Chopped and shuffled training or evaluation data of linear motion leads to unrealistic transients of abrupt jumps in the ground truth. The network output follows quickly but during transition initial error is high and decreases over time until estimation has settled to a value close to the ground truth. This explains high error peaks for high transients. Further, the influence of error outliers on the statistics can be considered low as most velocities are low to medium velocities (see Fig. 4) and hence not that many high transients occur relative to the overall number.
Additionally, during the adjustment of the model output, higher errors are accumulated, which deteriorates MSE and MAE compared to realistic transients.

In Fig 9 the box plot of the prediction error for different velocity bins in both axis directions is depicted.

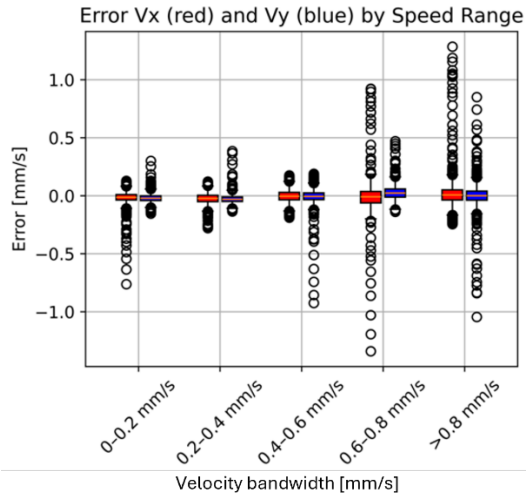

Fig. 9 SNN velocity estimation error box plot.

Besides the axis dependency of the error a velocity dependency can be observed with increasing errors with the velocity. Further the number and spread of outliers increases with higher velocities.

Generalization data set - The x-axis velocity component of SNN output for the ISO trajectory with the according estimation error is depicted in Fig. 10.

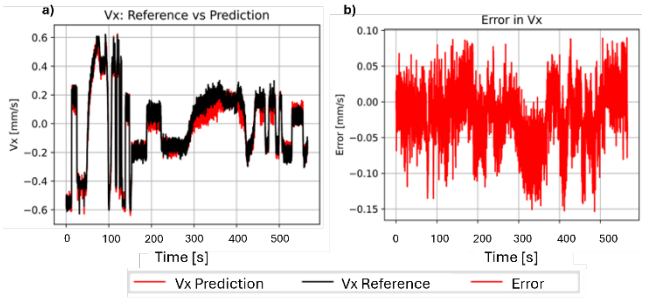

Fig. 10 SNN velocity estimation in x-direction and according error with (a) showing the predicted velocity against the reference and (b) depicting the resulting error over time.

The output of the SNN can follow the course of reference velocity with oscillations of max. 0.177 mm/s around the mean. But the error comes with a visible increase of the mean error at $t_1 = 300$ s motion time and seems stay with constant offset until $t_2 = 400$ s. In contrast the mean error jumps to mean error values close to zero at $t_2 = 500\ s$.
Point t_1 marks the beginning of the circular motion feature while point t_3 is at the start of the abrupt change of motion feature visible in the red marked area of Fig. 2. The x-axis velocity component of the SNN output for the ISO trajectory at the abrupt change feature is depicted in Fig. 11.

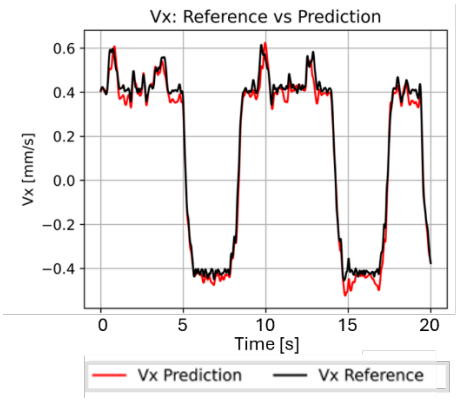

Fig. 11 SNN velocity estimation at abrupt change feature.

The five transients between different velocities can be well represented by the SNN output. Deviations occur during parts of constant velocity with small fluctuations. In Fig 12 the error boxplot of the abrupt change feature is depicted.

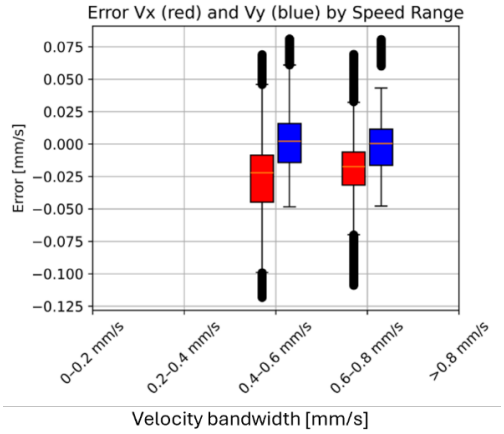

Fig. 12 Velocity estimation error box plot at abrupt change feature.

Axis dependency is visible for the abrupt change feature, but velocity dependency is not that prominent in both axis with y-axis errors being close to zero for both velocity bins.

The x-axis velocity component of the SNN output for the ISO trajectory at the circular motion change feature is depicted in Fig 13.

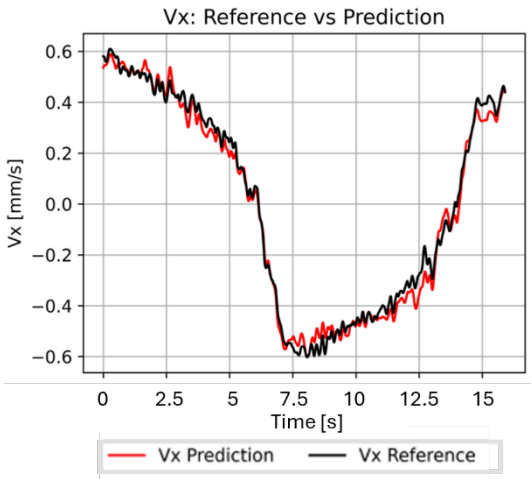

Fig. 13: SNN velocity estimation at circular change feature.

The higher the slope of the curve the closer the SNN output aligns with the ground truth. More subtle changes show higher deviations. In Fig 14 the error boxplot of the circular motion feature is depicted.

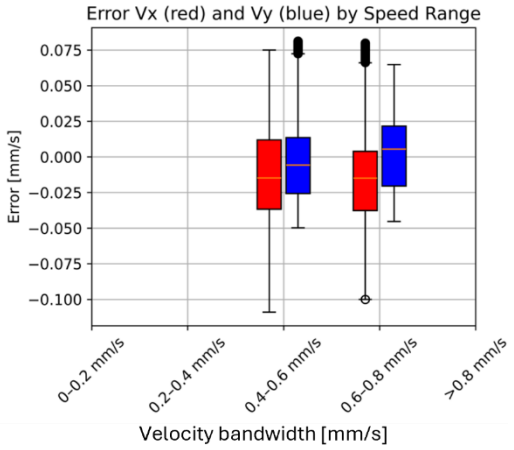

Fig. 14 Velocity estimation error box plot at subtle change feature.

Error axis dependency is observable with higher errors for x-axis motion components, but velocity dependency is not a pronounced pattern in the circular feature. Variance is larger than for the abrupt change feature for all axis and velocities. In table 3 an overview of the error statistics of the ISO trajectory velocity estimation is given.

Table 3 Axis wise MAE, MSE and Standard Deviation of SNN velocity output of ISO trajectory.

	MAE [mm/s]	MSE [mm/s]	Std. Dev [mm/s]
x-axis	0.0341	0.0019	0.2460
y-axis	0.025	0.001	

Overall axis-dependency with higher errors for the x-axis estimation is visible for the ISO trajectory too, but each error metric is smaller than for the linear motion.

In Fig 15 the box plot of the prediction error at the ISO trajectory for different velocity bins in both axis directions is depicted.

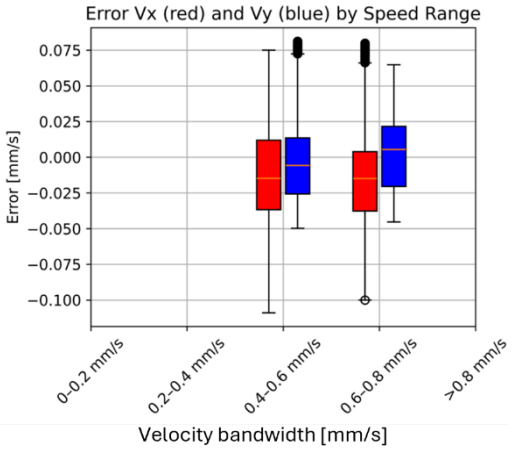

Fig. 15 SNN velocity estimation error box plot.

Median error and variance are improved for higher velocities, which is an inversed velocity dependency compared to the linear motion results. Further, Axis dependency of the error is visible, with higher errors for x-axis motion estimates.

5. Discussion

Event cameras typically produce sparse data, which can be challenging for event data processing solutions such as SNNs [10]. However, when laser speckle patterns are properly configured, they can generate dense event representations that are well-suited for SNN-based motion estimation as seen in Fig. 3. At the same time dense speckle patterns can lead to excessive event generation and contradict the fundamental advantage of event cameras, namely their ability to minimize redundant information by ignoring static background. This may lead to camera bandwidth saturation, introducing latency or information loss. Therefore, the speckle pattern size and structure must be tailored for the intended application. Possible solutions include distributing the speckle pattern across a larger pixel area by adjusting optical parameters such as the illuminated surface or the use of a different aperture size. Particularly when scaling to higher velocities balancing spatial pattern stability and speckle grain size becomes highly relevant as the signal projection on the sensor may exceed the event bandwidth.

The observed variation in event rate despite constant motion (see Fig. 5) aligns with known structural properties of laser speckle fields [24–26]. Speckle fields consist of grains with spatially varying size and intensity due to stochastic interference of coherent light scattered by rough surfaces [6]. As event cameras are sensitive to brightness changes, the number of events triggered per pixel is directly influenced by local speckle characteristics. Smaller speckles induce faster intensity changes, resulting in more frequent events. This distribution of speckle sizes explains the continuous distribution of event rates observed in Fig. 5. The variation in speckle size is constant for our workpiece and laser combination and therefore the local event rate increases

proportionally with velocity. The event rate variation is always present. This variability leads to overlapping event rate bands, creating input ambiguities that the network cannot easily resolve. Consequently, the network struggles to correct estimation offsets or drifts, as multiple velocity states may yield similar input signals. This error source is to some degree inherent to the idea presented in this paper. The current dataset, however, is limited to low velocities up to 1 mm/s, which confines evaluation to fine motion. Extending the dataset to higher-speed scenarios will be an important next step to examine the scalability of these effects.

But, increasing the contrast of the speckle pattern, for instance by narrowing the spectral bandwidth of the laser source, leads to sharper speckle edges due to higher temporal coherence [27]. Sharper intensity transitions result in steeper brightness gradients. When motion occurs, such a pattern would produce more localized and temporally precise events than a blurry, low contrast speckle pattern. As a result, the input signal becomes more structured and consistent and is beneficial for velocity estimation using SNNs. Particularly, it would help in suppressing ambiguity introduced by gradual transitions in low-contrast regions.

Static event noise contributes minimally to the total number of events recorded during motion, suggesting a negligible impact on estimation performance. As velocity increases, the relative influence of this static noise further diminishes. However, the accuracy of the reference data obtained from the laser tracker also limits performance. Measurement noise in the reference signal introduces uncertainty in supervision, affecting training stability and potentially contributing to fluctuations in prediction error. Improving reference data quality could therefore yield more consistent learning outcomes. In addition to variations in the event signal, noise in the reference data has a substantial impact on estimation accuracy. The standard deviation of the prediction error is comparable to that of the reference signal, indicating that the network has learned a similarly high sensitivity to noise through the supervision data. This reactivity can be beneficial in dynamic situations, as demonstrated by the model's accurate response to abrupt transitions in the generalization dataset. However, it also amplifies small fluctuations in the event input, leading to increased errors during subtle velocity changes (see Fig. 12). This behavior supports the need for preprocess filtering that was applied. The filtering step improved input-signal quality by narrowing the velocity spectrum to realistic values in range of the hexapod ($< 1\ \mathrm{mm/s}$) and producing more distinct features, as evidenced in Fig. 4. Further improvements could also be achieved through optimized training configurations, as the computational cost of simulating SNNs currently constrains temporal resolution and network depth. Higher temporal precision, while beneficial for accuracy, significantly increases hardware demands, underlining the need for efficient implementation strategies.

The centralized velocity distribution along both axes (see Fig. 4) confirms that low to medium speeds dominate the dataset. Despite this, estimation performance did not deteriorate at higher velocities in the ISO trajectory (see Fig. 15). In fact, performance improved, likely due to more consistent input signals under smoother, more natural transitions. In contrast, performance for high velocities in linear motion was lower (see Fig. 8), but this correlates with unrealistic velocity jumps introduced during preprocessing (see Fig. 7). These sharp transitions from low to high speed are not represented in the training data and cannot be accurately followed by the network. Once higher velocities are reached, the network's estimation stabilizes. A broader evaluation across different motion regimes and environmental conditions will be needed to confirm generalization beyond the current setup and to benchmark performance against classical event-based processing methods.

An axis-dependency error was observed across all experiments, with higher errors for x-axis velocities. One potential cause includes undetected misalignments between the camera, hexapod and laser tracker coordinate system, which could lead to an uneven projection of the true motion onto the camera axes. Such a misalignment would effectively distribute velocity components between the axes in a manner inconsistent with the intended ground truth, leading to systematic errors that differ between axes. Even small angular offsets of $\theta \leq 1\,^\circ$ can introduce measurable cross-axis projection in low-velocity experiments.

For a motion in pure x-axis direction of $V_\mathrm{x} = 0.5\ \frac{\mathrm{mm}}{\mathrm{s}}$ and an angular misalignment of $\theta = 1\,^\circ$ a project velocity in y-direction of:

$$V_\mathrm{y} = V_\mathrm{x} \sin\theta = 0.00875\ \mathrm{mm/s}. \tag{8}$$

would be measurable. Compared to error standard deviation this can account for an error of 3 %.

The incident angle of the beam could further distort the spatial contrast distribution of the speckle field, which influences the event generation and could hence affect motion sensitivity differently across axes. Additionally, high velocity y-axis components seem to be a bit better represented in the data set than x-axis components (see Fig. 4) which could have influenced SNN training. Anisotropic behavior of the laser tracker error or hexapod motion accuracy may influence the accuracy of the reference data and contribute to this discrepancy. Future work should also consider deployment on neuromorphic hardware to validate real-time feasibility and assess performance under practical conditions.

The validation loss remains lower than the training loss presumably because the training data are randomly segmented and shuffled, whereas the validation data preserves their original temporal order. The temporal continuity in the validation data makes the prediction task slightly easier as changes are less abrupt as the once visible in Fig. 8. Hence validation error is lower, but both curves decrease together and plateau, which indicates stable learning without overfitting. Further, generalization across the full range of velocities and motion types was successfully demonstrated in the ISO trajectory. Realistic velocity transitions led to overall improved estimation accuracy. The model performed especially well during abrupt changes, indicating stable and responsive predictions in dynamic phases. While smooth direction changes were also handled consistently, increased variance and error suggest reduced model performance in these segments. This could be attributed to the model's sensitivity to local changes in event rate, which are exaggerated in slow or gradually varying motions leading to higher noise in the prediction. Overall, the dominant source of error across all tests remains the fluctuation around the reference

velocity. With similar error standard deviations for both velocity axes (see Fig. 6 and Fig. 9), this continues to limit performance. Since the standard deviation of the error closely matches that of the reference data, reducing the influence of laser tracker noise could lead to substantial improvements particularly at lower velocities. At higher velocities the relative influence of the LT error is expected to diminish as its absolute error remains constant while signal magnitude increases, indicating potential benefits for future higher speed experiments.

6. Conclusion

In this work, we investigated the application of Spiking Neural Networks for continuous-time planar velocity regression using event data. The results demonstrate that SNNs can successfully process the dense event stream generated by a moving laser speckle field. The approach shows promising performance across various motion scenarios with typical features of robotic motion. However, the evaluation has so far been limited to restricted parameter space in terms of velocities and environmental variations. Future work will have to focus on optimizing the speckle pattern projection to able to control event data rate and reduce its variations. Especially with increasing data rates at higher velocities. Additionally, robustness against environmental factors such as interfering with light from industrial environments needs to be addressed to enable real world applications.

Acknowledgments

(1) The authors acknowledge the financial support by the Federal Ministry of Education and Research of Germany in the framework of Research Campus Digital Photonic Production (project number: 13N15423).

(2) Funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany's Excellence Strategy – EXC-2025 Internet of Production – 390621612.

References

[1] T. Bauernhansl, M. ten Hompel, and B. Vogel-Heuser: Industrie 4.0 in Produktion, Automatisierung und Logistik: Anwendung · Technologien · Migration (Springer Vieweg, Wiesbaden, 2014), 5.

[2] J. Bremer, P. Walderich, N. Pirch, J. H. Schleifenbaum, A. Gasser, and T. Schopphoven: J. Laser Appl., **33**, (2021), 12045.

[3] R. Poprawe, C. Hinke, W. Meiners, F. Eibl, O. Zarei, M. Voshage, S. Ziegler, H. Schleifenbaum, A. Gasser, T. Schopphoven, E. Willenborg, J. Flemmer, C. Weingarten, J. Finger, and M. Reininghaus: Proc. SPIE, Vol. 10519, (2018), 1051907.

[4] T. Kaster, J.-H. Rissom, E. Breit, J.-N. Schneider, L. Gorissen, and C. Hinke: Proc. SPIE, Vol. 13356, (2025), 1335602.

[5] P. Walderich, L. M. Gorißen, T. Kaster, and C. R. Hinke: J. Laser Micro/Nanoeng. **19**, (2024), 79.

[6] J. W. Goodman: *Laser Speckle and Related Phenomena* edited by J. C. Dainty (Springer, Berlin, Heidelberg, 1975), 46.

[7] T. O. H. Charrett, R. P. Tatam, and L. Waugh: Proc. SPIE, Vol. 10329, (2017), 103290S.

[8] P. Rehain, J. Ramanathan, Y. M. Sua, S. Zhu, D. Tafone, and Y.-P. Huang: Opt. Lett. **46**, (2021), 4346.

[9] Q. Fang, A. Tomar, and A. K. Dunn: Biomed. Opt. Express **15**, (2024), 1004.

[10] G. Gallego, T. Delbrück, G. Orchard, C. Bartolozzi, B. Taba, A. Censi, S. Leutenegger, A. J. Davison, J. Conradt, K. Daniilidis, and D. Scaramuzza: IEEE Trans. Pattern Anal. Mach. Intell. **44**, (2022), 154.

[11] Z. Ge, N. Meng, L. Song, and E. Y. Lam: Appl. Opt. **60**, (2021), 172.

[12] Zhou Ge, Yizhao Gao, Hayden K.-H. So, and Edmund Y. Lam: Opt. Lett. **46**, (2021), 3885.

[13] M. Pfeiffer and T. Pfeil: Front. Neurosci. **12**, (2018), 774.

[14] K. Roy, A. Jaiswal, and P. Panda: Nature **575**, (2019), 607.

[15] A. Tavanaei, M. Ghodrati, S. R. Kheradpisheh, T. Masquelier, and A. Maida: Neural Netw. **111**, (2019), 47.

[16] S. B. Shrestha and G. Orchard, "SLAYER: Spike Layer Error Reassignment in Time, arXiv:1810.08646, (2018).

[17] M. Gehrig, S. Shrestha, D. Mouritzen, and D. Scaramuzz: Proc. IEEE Int. Conf. Robot. Autom. (ICRA), (2020), 4195.

[18] A. Henkes, J. K. Eshraghian, and H. Wessels: R. Soc. Open Sci. **11**, (2024), 231606.

[19] S. B. Tandale and M. Stoffel: npj Unconv. Comput., 1, (2024), 2.

[20] B. Yang, Y. Liao, and J. Ke: Opt. Lasers Eng. **189**, (2025), 108944.

[21] P.-S. V. Sun, A. Titterton, A. Gopiani, T. Santos, A. Basu, W. D. Lu, and J. K. Eshraghian: arXiv:2211.10725, (2022).

[22] J. K. Eshraghian, M. Ward, E. Neftci, X. Wang, G. Lenz, G. Dwivedi, M. Bennamoun, D. S. Jeong, and W. D. Lu in Proc. IEEE, 111, (2023), 1016.

[24] Q. Zhang, Y. Wu, Y. Pan, X. Wang, Y. Liu, Y. Fang, Y. He, J. Tian, and Z. Wang: Nat. Commun. **14**, (2023), 1234.

[25] X.-B. Hu, M. X. Dong, Z. H. Zhu, W. Gao, and C. Rosales-Guzmán: Sci. Rep., 10, (2020), 199.

[26] Z. Hajjarian and S. K. Nadkarni: Opt. Lett. **40**, (2015), 764.

[27] J. W. Goodman: Speckle Phenomena in Optics: Theory and Applications (Roberts and Company Publishers, 2007), 9.

(Received: June 27, 2025, Accepted: April 5, 2026)