UNIVERSITAT POLITÈCNICA DE CATALUNYA

Programa de Doctorat:

AUTOMÀTICA, ROBÒTICA I VISIÓ

Pla de recerca:

Bio-inspired event-driven intelligence for ego-motion estimation

Yi Tian

Directors:

Juan Andrade Cetto

Setembre 2021

Contents

1 Introduction and motivation **2**

2 Objectives and expected contributions **4**

3 State of the art **4**

3.1 Event-based Optical flow estimation methods 4

3.1.1 Model-based method 4

3.1.2 Learning-based method 5

3.2 Spiking Neural Network for event-based vision 6

3.2.1 SNN for optical flow estimation 6

3.2.2 SNN for ego-motion estimation 6

3.3 Ego-motion estimation methods from optical flow 6

4 Work Plan **7**

4.1 Task 1: Literature Review 7

4.2 Task 2: Trained SNN model 7

4.3 Task 3: SNN for motion estimation 7

4.4 Task 4: Thesis writing 8

5 Resources **9**

6 Preliminary work **9**

6.1 Overview 9

6.1.1 Spiking Neural Network Architecture 9

6.1.2 Spiking Neuron Model 10

6.2 Optical flow computation 10

6.2.1 Normal optical flow 10

6.2.2 Computation of full flow 11

6.3 Ego-motion estimation using optical flow 12

6.3.1 Motion field model of rigid static scene 12

6.3.2 Pure rotational motion 14

6.3.3 Pure translational motion 14

6.3.4 RANSAC for robust estimation 14

6.4 Results 15

6.4.1 Datasets and implementation details 15

6.4.2 Pooling flow evaluation 16

6.4.3 Ransac ego-motion estimation evaluation 17

6.4.4 Pure translational direction estimation 18

6.5 Discussion 19

1 Introduction and motivation

Biological organisms monitor their movement by perceiving ego-motion using visual cues. The ego-motion in computer vision, referring to the 3-D camera motion relative to a rigid scene, forms the basis for visual odometry (VO) and simultaneous localization and mapping (SLAM) [1]. They play pivotal roles for mobile robotic and augmented reality (AR) applications. Especially, for self-driving vehicles or autonomous aerial robots, the ability to navigate in complex environments is crucial yet challenging. Failures occur under uncontrolled environments, such as low-textured scene or illumination changes, due to the limitations of current visual motion perception techniques. Both the conventional digital cameras and vision processing algorithms differs from the biological vision systems. In terms of sensors, conventional frame-based cameras measure light at each pixel and output an entire image synchronously, while biological retina cells operate independently and asynchronously. At the algorithm level, current development in artificial intelligence has made remarkable achievements in computer vision tasks. Notably, inspired by the hierarchical architecture of the neural network in our brain, deep analog neural network (ANN) has yielded state-of-the-art performance for image classification benchmarks [2]. However, the success comes at the cost of sacrificing huge human labor by annotating data and tremendous computational resources. In contrast, our brain process data streams in a spike-based event-driven way. Neurons are connected by synapses and communicate through discrete action potentials, which allow sparse and efficient data transfer [3]. Such a gap between their computing principles makes the current vision perception systems less robust and effective than the biological systems for ego-motion estimation.

For animals and human beings, the visual cue is processed in the dorsal visual pathway from the primary visual cortex V1 to the parietal lobe to gain spatial awareness and guidance of actions (shown in Figure 1) [4]. In concrete, when the retina receives the light, the rod and cone cells first change their membrane potentials. These changes further propagate to the ganglion cells with temporal information. The ganglion cells fire according to the light intensity changes and generate either transient burst action potentials or sustained discharge through the optic nerve to the visual cortex. Visual motion is then processed in a hierarchical structure with interconnected regions [5]. Each region is sensitive to more complex motion patterns. Neurons in the primary visual cortex (V1) are direction-selective as they only respond to oriented edges, which is commonly refer to aperture problem, while neurons in higher area named the middle temporal (MT) fire in proportion to the true direction. They receive the spikes from the V1 area and have a larger receptive field. Neurons in medial superior temporal (MST) area receive major input from MT area and respond to the entire optical flow pattern, which is defined as the pattern of apparent motion of 3-D world objects on the retina caused by the relative motion between an observer and the visual scene [6]. Researches have shown selectivity to different flow patterns such as expansion, contraction, rotation or spirals in the MST area [7, 8]. Animals can adjust its posture and stabilize the heading direction by perceiving the ego-motion from the retinal optic flow.

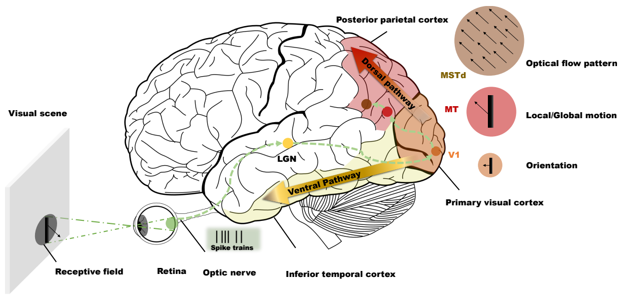


Figure 1: The visual motion processing pathway in human brain

Mimicking the bio visual pathway up to the retinal ganglion cells, neuromorphic vision system such as dynamic vision sensor (DVS) [9] or DAVIS [10], shows the advantage to overcome common issues of frame-based cameras. They are commonly called ¨silicon retina¨ or event-based camera. In the present dissertation, we use event-based camera to refer to such technology. Event-based cameras detect independently the change of luminance in each pixel and produce an asynchronous feed of pixel coordinates where changes occur. Since no frame or image is produced, the event-based processing can detect fast motions with low latency down to 1µs time resolution. Since each pixel is processed independently, they have high dynamic range up to 140dB, which makes them robust to changes in illumination conditions. Further, their sparse output without redundant information requires low power consumption to be adopted in mobile embedded systems. All the advantages make them an ideal candidate for challenging mobile robotic tasks. However, solving the visual perception problems for the case of event cameras requires novel algorithms to cope with the unfamiliar event-driven data.

One promising solution to the challenges involves using brain-inspired spiking neural network (SNN), which is considered to be the third generation of neural networks [11]. While ANNs make use of the amplitude of the signals, SNNs process discrete spikes or binary events by using timing information. Inspired by the working principles of the brain, neurons in SNNs fire when their membrane potential reaches a threshold and further propagate the non-binary signal to other neurons through synapses. They are event-driven as they are only active when spikes arrive. The sparse and event-driven characters of SNN allows computational and energy efficiency, making it an natural match to process the spatio-temporal data from event-based cameras. Further, they enable unsupervised learning by spike-timing-dependent plasticity (STDP) rules. The biologically plausible STDP rule is a local rule based on timing information and suits hardware implementation [12]. However, as exploiting the state-of-the-art training techniques in conventional ANNs requires continuous and differentiable data, training an SNN is tricky. One approach to realize learning in SNN is using ANN-to-SNN conversion manner, which refers to training an ANN and converting the trained model to SNN. They have yield competitive performance on image classification benchmarks but need larger inference time. To apply supervised learning, approaches using gradient descent based global backpropagation on SNNs usually require a differentiable approximate function [13, 14]. For local learning rules, it is difficult to train on multiple layers as the spike propagation vanishes in deeper layers. Thus, a common approach is training them by means of a local-global fashion: train a SNN layer by layer using local learning rules and add global backpropagation layer for classification. Lacking appropriate training approaches has been the major limitation for the application of SNNs. Nonetheless, the combination of event-based camera and SNN has arouses increasing interests in different communities, from neurosience to robotics. Recently, several works have been done using SNN on neuromorphic chips combined with event camera for high-speed control tasks [15].

We believe that combining the spatio-temporal data from an event-based camera and the temporal-processing capability from SNN can yield a bio-inspired computational-efficient solution for visual motion perception tasks. However, these bio-inspired SNNs have only been used to tackle low-level tasks in the literature such as feature extraction [16], optical flow [17, 18], object/gesture recognition [19, 20], trajectory prediction [21]. Limited work has been done using pure SNN framework for higher-level or large-scale tasks such as camera motion estimation and tracking. Our interests remain: How to fully exploit the temporal-processing scheme of SNNs and the event camera? Which is the appropriate learning approaches to unlock the potential of bio-inspired SNNs to perform continuous complex tasks such as 3-D ego-motion estimation?

2 Objectives and expected contributions

The objective of this dissertation is to design and implement a biologically-inspired system for ego-motion estimation from event camera data. We will explore SNN approaches that are naturally adapted to event-camera. The software implementation is expected to yield reliable results to estimate 3-D motion of the camera. Both simulated and real-scene event camera data will be tested.

The first step lies in developing and implementing the mathematical model for ego-motion estimation from optical flow. Initial study and implementation of event-based optical flow and ego-motion estimation will be done using the state-of-the-art SNN frameworks. The next phase of the PhD will involve developing our own learning methods for optical flow estimation. Different learning algorithms will be explored such as unsupervised learning using Spike-Timing Dependent Plasticity (STDP) rules, self-supervised learning and supervised learning using spike-based back propagation. The results will be compared with other state-of-the-art approaches on event-based optical flow benchmarks. In the third step, we will explore the ego-motion estimation method to develop robust ego-motion estimation methods. Both optimization and learning-based methods will be explored. The expected approaches will be able to estimate 3-D camera motion up to a scale without prior knowledge of the rigid scene structure. In the end, we will aim at fully exploiting the high-temporal resolution and asynchronous data from the event camera to perform high speed ego-motion estimation in real time.

This PhD dissertation seeks to contribute to developing a bio-inspired event-camera system for ego-motion estimation, which can be later employed in visual localization and mapping for mobile robotic systems. Event cameras are naturally suitable for motion-relevant tasks such as optical flow estimation or ego-motion estimation, which forms the basis of tracking, SLAM in computer vision and robotics community. Spiking Neural Network (SNN) is the ideal candidate for event-based camera. This dissertation aims to exploit the advantage of event-camera and bio-inspired visual motion processing methods. All the algorithms will be implemented using open source frameworks either in python or C++. The expected results may provide insights for efficient computational vision motion perception systems.

3 State of the art

3.1 Event-based Optical flow estimation methods

Optical flow refers to the pattern of apparent motion of 3-D world objects on the image plane by the relative motion between the camera and the visual scene. For conventional cameras, the optical flow is usually computed using the spatial and temporal derivatives based on the brightness constancy assumption and smoothness assumptions [22]. Conventional optical flow techniques include differential methods, frequency-based methods, correlation-based methods. For more details readers can refer to [23]. We will focus on event-based optical flow estimation methods in this section. For event cameras, given the asynchronous streams with timing information and naturally detected edges, the computation of optical flow has gained growing interest. It is challenging as well because there is no brightness value from the output data and it is not spatially continuous due to its sparsity. Event-based optical flow techniques include model-based and learning-based methods.

3.1.1 Model-based method

Based on gradient-based methods, early works have been bone to present an adaptive version of the Lucas-Kanade algorithm for event-based vision [24, 25], but their performance is limited as it is difficult to estimate derivatives due to the sparse encoding of address-event representations (AER). Recently, Almatrafi et al. [26] present the distance surface methods, which is an assigned intensity value based on the proximity to the spatially closest detected event pixel. It is then serves as input to the intensity-based optical flow methods.

Spatiotemporal plane fitting shows to be an alternative robust approach by considering the local distribution of events in x-y-t space [27, 28, 29, 30]. Optical flow can be computed taking the inverse gradients of a plane fitted to the surface based on the constant velocity assumption within this surface. Benosman et al. [27] applied linear least-squares method to a spatio-temporal neighborhood on each newly coming event. Mueggler et al. [31] improved the method using active events surface, which only stores the last generated events. Based on plane-fitting methods, Pijnacker Hordijk et al. [28] improved method by reducing the number of parameters and used a time-window with adaptive length. Since the plane-fitting method is sensitive to noises, RANSAC-based technique is often used for outlier rejection [31, 32]. Implementation on FPGA is also done in [29]. Recent work [33] uses a Principle Component Analysis (PCA) approach and shows reduction in computation time. Though plane-fitting method is able to work with sparse representations, it can only output normal flow and the performance depends on the good fitting of the spatio-temporal window to the scene texture.

Energy-based methods use spatio-temporal filters, or Gabor filters for motion speed and direction selectivity. Biologically-inspired methods [34, 25, 35, 36, 37], which try to mimic the motion estimation mechanism in dorsal stream of the biological visual system [38], fall into this category. Orchard et al. [35] uses SNN to detect optical flow using self-designed spatio-temporal filters with different speed and directions. Paredes-Valles et al. [36] learns these filter by a novel unsupervised STDP method. The idea is to let the neural network recognize the spatio-temporal pattern through coincidence detection, which can be traced back to the Hassentein-Reichardt Element Motion Detector (EMD) model [39]. The limitation of this method lies in that it requires a large number of spatio-temporal filters and has higher computational and memory requirements since the spatial and temporal frequency content of the scene is not known in advance. Especially, since the implementation can only output a confidence value, it has accuracy problems when it comes to the magnitude .

3.1.2 Learning-based method

Deep learning techniques using analog neural network (ANN) have been exploited for event-based optical flow estimation in [40, 41, 42]. Zhu et al. [40] trained a CNN based on a self-supervised learning method using photometric loss of the gray scale images. They later improved their work to perform unsupervised learning of optical flow, ego-motion and depth based on motion-compensation loss [41]. Ye et al. [42] presented the first ANN framework to estimate dense flow, ego-motion and depth from event only using unsupervised learning. Recently, Stoffregen et al. [43] trains synthetic event data with supervised learning based on the work from [40, 41] and yield higher performance on benchmarks. In [44], the authors propose a light weight neural network that is able to estimate optical flow in a self-supervised manner using the contrast maximization proxy loss. In [45], a lightweight pyramid network is presented to learn event-based optical flow in a self-supervised way. The proposed framework is simplified based on Laplacian pyramidal decomposition and channel attention mechanism, which shows lower computation cost. Other learning-based methods using SNNs or hybrid structures will be discussed in Section 3.2.1.

3.2 Spiking Neural Network for event-based vision

3.2.1 SNN for optical flow estimation

Methods using SNN for optical flow estimation without learning [35, 37] usually fall in the bio-inspired Gabor-filter method category that was described in 3.1.1.

SNN-based method for learning event-based optical flow has been exploited in [36, 46, 47, 48]. For unsupervised learning, Paredes-Valles et al. [36] presented a convolutional SNN framework for local and global event-based optical flow estimation based on a novel unsupervised STDP method. Barbier and Triesch [47] presents a SNN that learns orientation, motion and disparity similar to the simple and complex cells in primary visual cortex. The network was trained by STDP from stereo event-based inputs. However, all the methods mentioned above suffer the common problem of spatio-temporal filter methods (refer to Section 3.1.1) and can only be deployed for those dataset similar to that used for training. Lee et al. [46] proposed a hybrid architecture combining SNN in the encoder and ANN in the decoder. The network is trained by self-supervised learning method from EV-FlowNet [49]. They later proposed in [50] another framework which contains both SNN and ANN-based encoders to extract features from the event streams and synchronized grayscale images respectively. The fused output is then passed to the rest of the network. Recently, Paredes-Vallés et al. [48] proposes another pure SNN approach for event-based optical flow estimation using self-supervised learning based on the loss function from [41].

3.2.2 SNN for ego-motion estimation

While previous works do exist using ANN architecture to learn ego-motion from event camera inputs [42, 41], for SNN, the only work related is found in [51]. The authors presented a SNN framework to continuously perform 3-DoF (Degree of Freedom) angular velocity regression based on supervised learning approach from [52]. However, using SNN to recover 6-DoF camera motion has not been addressed in the literature so far.

3.3 Ego-motion estimation methods from optical flow

Given optical flow information, it is sufficient for observers to perceive translational direction of self motion [53]. Neurophysiological and the psychophysical studies try to discover the optic flow processing properties in the primate visual pathway. The response of MST neurons to different flow field patterns caused by expansion, rotation or translation can be learned by unsupervised learning [54, 55, 56] or back-propagation network [57]. Heading estimation (or translation direction) has been recovered by tuning neurons to individual optic flow patterns by template matching models [58]. Lappe and Rauschecker [59] proposed a population-based neural network that is able to encode motion speed and direction and shows selectivity for translational direction based on the method of [60].

In the computer vision community, recovery of the camera motion often involves optimization methods by minimizing the difference between the measured flow and the model flow. Visual motion model can be constructed either in a discrete or a differential viewpoint. High speed cameras like event cameras are not limited by frames, and can be valid to use the differential model. Longuet-Higgins and Prazdny's model [61] is the most used one to describe the relationship between visual motion fields and ego-motion in a rigid environment through a differential viewpoint. Given the visual motion model, robustly ego-motion estimation methods are usually composed of three components: optimization constraint (including squared distance constraint, bilinear constrains, motion parallax constraints), optimization technique and robust estimator [62].

While early works use combined methods to estimate translation and rotation in one step with known depth value [63], most algorithms treat rotation and translation separately [64, 65, 60].

Rieger and Lawton [65] tried to solve translation first based on motion parallax. It works when there are large depth differences but is difficult to measure near occlusion boundaries. Heeger and Jepson [60] derived the subspace constraint from the bilinear constraint to release the dependency on rotational motion and proposed subspace method.

For optimization techniques, least squares (LSQ) method is used [60, 65] as linear optimization. Nonlinear problems using numerical solvers include fix-point iteration (FP) [64, 66] and Gauss-Newton iteration (GN) [67, 68]. Instantaneous inaccurate estimations often occur either due to errors in the optic flow estimation or additional non-static objects contributing to the visual motion fields in real world scenarios. Robust estimators techniques such as random sample consensus (RANSAC) [69] are often combined with optimization techniques .

4 Work Plan

4.1 Task 1: Literature Review

The literature review phase consists in a deep study and critical analysis of all the books, publications and experimental data related to specific topic. At the beginning of the research, literature review should be done related to event camera optical flow estimation techniques, spiking neural network and ego-motion estimation methods. Periodic literature review is fundamental throughout the research cycle.

4.2 Task 2: Trained SNN model

The initial task will focus on studying and implementation of existing SNN architectures for event-based optical flow and camera motion estimation. The goal is to explore the feasibility and performance of state-of-art SNN frameworks for optical flow and ego-motion estimation tasks. This task will involve:

Task 2.1: Inference test for optical flow estimation

In this task, we will focus on replicating event-based optical flow estimation by existing SNN framework using unsupervised learning method. We will generate different simulated datasets to evaluate the optical flow estimation method. Depending on the results, we will develop techniques to refine the optical flow estimation performance.

Task 2.2: Ego-motion estimation methods

We will study the mathematical formulation of visual motion field and ego-motion estimation methods based on optical flow.

Task 2.3: Integration and tests

The ego-motion estimation method will be implemented and integrated into the SNN framework. Depending on the performance of the systems, only constrained motion (pure rotation, pure translation or planar scene) and simulated artificial scene will be used for tests.

4.3 Task 3: SNN for motion estimation

In this task we will explore and develop a SNN framework for optical flow estimation and ego-motion.

Task 3.1: SNN architecture and learning method

In this task we will develop our neural network architecture and learning method. Different learning methods will be explored including self-supervised learning, supervised learning and unsupervised learning. Hybrid architecture that combines SNN and ANN will also considered.

Task 3.2: Optical flow learning

We will train our neural network and evaluate the results on event-based optical flow benchmarks such as MVSEC dataset [40].

Task 3.3: Motion estimation integration

We will integrate our ego-motion estimation optimization method to the SNN framework. We aim to recover 3-D camera motion for both artificial and natural scenes.

Task 3.4: Ego-motion learning

Finally, we aim to develop a neural network that can learn 3-D motion directly. Tests on simulated dataset and experiments using event-camera on real scenes will be done to evaluate the performance of our method.

4.4 Task 4: Thesis writing

All the results will be integrated in the final doctoral thesis. At least two journal papers are planned. Yearly conference papers will also be proposed.

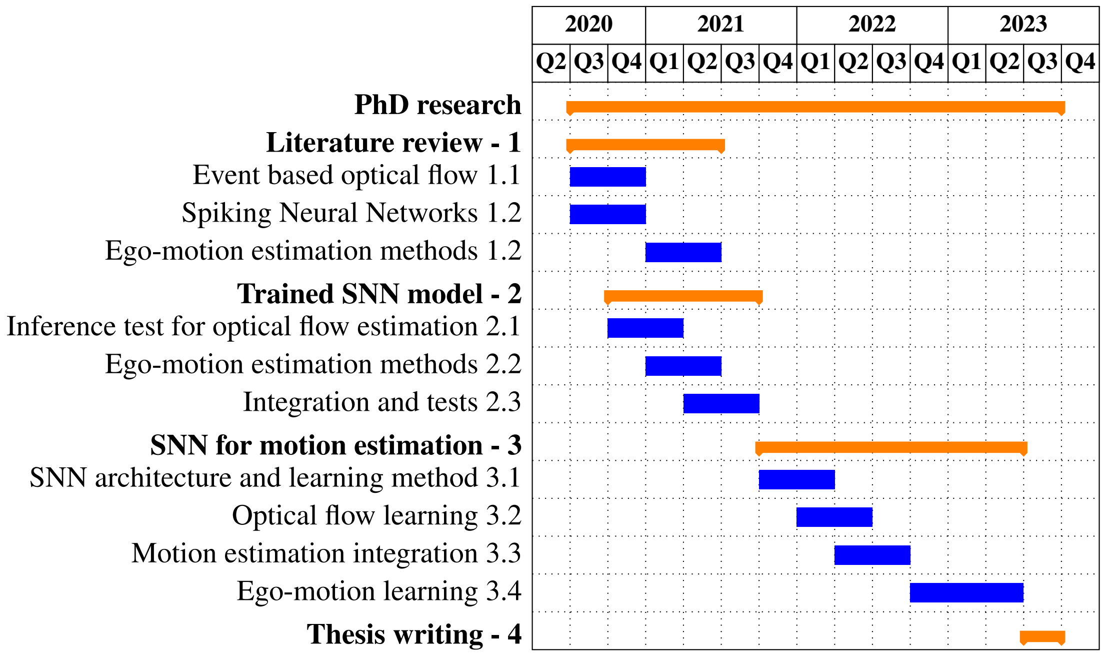


Figure 2: Intended work plan of the proposed thesis

5 Resources

This research is partially supported by the Spanish State Research Agency through projects EB-SLAM (DPI2017-89564-P). The work is being developed at the Institut de Robotica i Informatica Industrial (IRI), UPC-CSIC, in Barcelona, within the Mobile Robotics research group. Our laboratory for this project is equipped with a standard desktop computer with Ubuntu 18.4 operative system, ROS, Matlab among other applications.

Several online event camera datasets and simulators of natural and synthetic scenes that contains events and the ground truth for optical flow and ego-motion can be exploited for testing, such as ESIM simulator [70], DVSFFLOW16 [32] and MVSEC dataset [40]. For spiking neural network, open-source simulators available include Nengo [71], Brian2 [72] or NEST. They all provide python library and visualization functions. Other available resources include simulation frameworks with PyTorch functionality such as BindsNet [73] or SpykeTorch [74].

In case of real-time experiments, we possess two event cameras DAVIS240C in conjunction with ROS environment. Our Mobile Robotics laboratory is also equipped with other sensors and an Optitrack system.

6 Preliminary work

The report of the current state of our research is presented as follows:

6.1 Overview

We propose to use spiking neural network (SNN) for ego-motion estimation from optical flow. As an initial approach we start with a trained SNN model [36]. We integrate our pooling method in the SNN framework and implement our ego-motion estimation algorithms combined with RANSAC for robust results. Tests on the pure rotation and pure translation motion have been done.

6.1.1 Spiking Neural Network Architecture

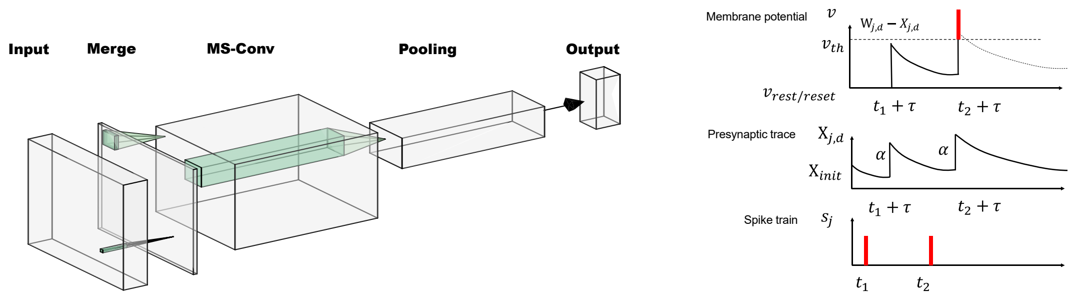
We adopt the SNN framework from [36], which presents a hierarchical SNN architecture using an adaptive neuron model and unsupervised spike-timing-dependent plasticity (STDP) learning. The neural network performs feature extraction, shows selectivity to the local motion from these features and integrates the local motion into global motion. We modified the neural network structure according to Figure 3a. The descriptions for each layer are as follows:

- **Input layer:** The input layer is two dimension corresponding to the event polarities. For each step, each neuron in the input layer is assigned with events in that temporal window with no overlap. The input size is scaled by 4 to improve computational efficiency.

- **Merge:** Since polarity information is not useful in our current motion estimation method, the merge layer decreases the complexity of the network by combining the two dimension layers into one single layer using a single 1×1 kernel.

- **MS-Conv layer:** MS layer is a convolution layer for local optical flow estimation. Neural connections are multisynaptic with different constant transmission delays.

- **Pooling layer:** Pooling layer is a convolutional layer to solve the aperture problem. We implemented a combined pooling method with sliding temporal window to integrate noisy and sparse normal flows from the previous layer. The output full flows serve for motion estimation.



(a) Spiking Neural Network Architecture

(b) Neural dynamics model

Figure 3: Spiking Neural Network model

6.1.2 Spiking Neuron Model

The spiking neuron model used is Leaky Integrate and Fire (LIF) model. It is one of the most commonly adopted model in literature due to its simplicity for implementation and low computational cost. The membrane potential of the LIF neuron $V(t)$ changes according to the presynaptic spikes received. When $V(t)$ reach the threshold V_{th}, the neuron fires, generating a spike or action potential and resets to V_{reset}. If there are no presynaptic inputs, $V(t)$ decays exponentially to V_{rest} controlled by a time constant τ_m. The input for a current post-synaptic neuron is the sum of the spike trains $s(t)$ received from its multisynaptic connections $m \times n$, where m and n indicate the number of synaptic delays and number of neurons in the previous layers. W is the strength or weight for each synaptic connection. The neural dynamics is defined as:

$$\tau_m \frac{dV(t)}{dt} = -(V(t) - V_{rest}) + \sum_{j=1}^{n} \sum_{d=1}^{m} (W_{j,d} s_j(t - \tau_d)) \tag{1}$$

The neural model in the network from [36] is a adaptive version of LIF, which uses presynaptic trace as an inhibitory item for the input as homeostasis mechanism. The contribution of the spike train from the previous layer to the presynaptic trace is scaled by a parameter α. The presynaptic trace decays exponentially according to a time constant τ_{trace}. The dynamic of the presynaptic trace $X_{j,d}$ for each synapse is defined as:

$$\tau_{trace} \frac{dX_{j,d}(t)}{dt} = -X_{j,d}(t) + \alpha s_j(t - \tau_d) \tag{2}$$

For simplification, we set resting membrane potential V_{rest} to zero and τ_{trace} same as τ_m. The dynamic equation of a neuron can be modified as Equation 3:

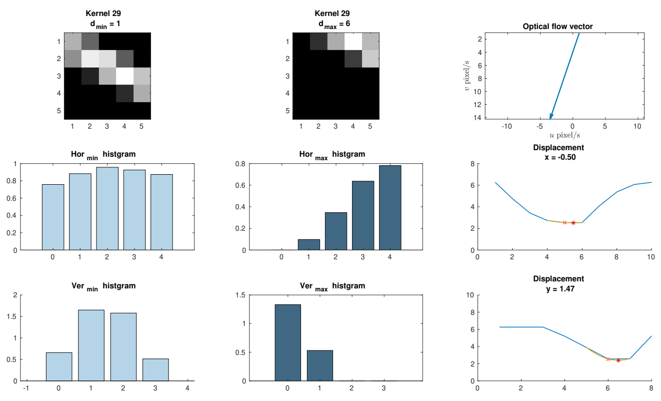
$$V(t + \Delta t) = V(t) e^{-\frac{\Delta t}{\tau_m}} + \sum_{j=1}^{n} \sum_{d=1}^{m} (W_{j,d} s_j(t - \tau_d) - X_{j,d}(t) e^{-\frac{\Delta t}{\tau_m}} - \alpha s_j(t - \Delta t - \tau_d)) \tag{3}$$

where Δt indicates the simulation step time. Figure 3b shows the membrane potential change of a post-synaptic neuron due to the input received from one synaptic connection.

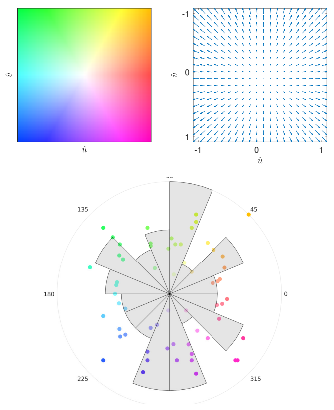
6.2 Optical flow computation

6.2.1 Normal optical flow

We used the 64 kernels trained from a rotating disk dataset for inference test. These kernels are three-dimensional filters x, y and τ. Each identifies motion at different directions and speeds. Since the kernels were trained in an unsupervised manner, proper readout mechanisms are required to get the local velocities. We recalculate the local velocity vectors for each kernel based on sum of absolute differences (SAD) block matching method [75]. Two synapses with the most information are selected for computation. First, the histograms of the horizontal and vertical directions are computed. Secondly, we find the SAD for both horizontal and vertical pairs, which refers to the displacement of the edge. To get a more accurate displacement value, we further apply a least squares fitting to get the optimal solution. Figure 4a shows the computation results for one of the 64 kernels. We assign a color coding scheme for optical flow field according to Figure 4b. Direction is encoded in hue value and speed is represented in saturation. Brightness is set to zero. Figure 4b shows the distribution of the 64 kernels with its corresponding color. Kernels have been divided into 16 bins according to their direction. Note that we have more kernels in vertical direction than in horizontal direction.



(a) SAD fitting for optical flow computation

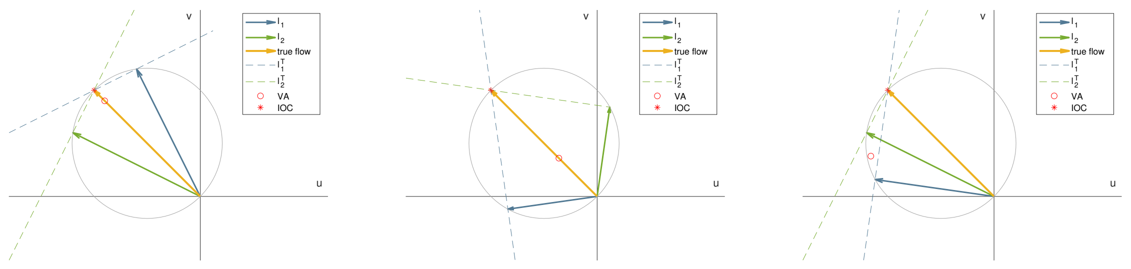


(b) Optical flow color code of 64 kernels

Figure 4: Optical flow computation for trained kernels

6.2.2 Computation of full flow

In the MS layer, the detection of the local motion is feature-dependent and has aperture problem. The component of the optical flow parallel to the edge is unknown. In biological systems, it is believed that the aperture problem is solved in the MT layer by pooling the optic flow across a larger receptive field [76]. Different pooling mechanisms may exist. Mathematically, veridical solutions for aperture problems include geometric method [77] and Intersection of the Constraints (IOC) method [78]. Geometric method searches for the best solution for a fitting circle among the normal flow data. Intersection of Constraints solution estimates the true velocity from at least two normal flow vectors. Since the true velocity always lies upon the constraint line, the unique arithmetic solution can be calculated looking for the intersection of at least two constraint lines. However, IOC may not yield correct solution for rotation flows [79]. In addition, for noisy and high-temporal input from the event camera, the IOC approach solution may deviate significantly. On the other hand, neuroscientific researches show that human visual systems may pool that 1D and 2D motion differently using vector-average (VA) solutions [80] and intersection-of-constraints (IOC) respectively [81]. Stimulus is classified into two types. Type I refers those flow vectors whose directions lie to the same side of the true velocity. In this case, the VA solutions will generate veridical estimates similar to IOC solutions, while for Type II stimulus, the IOC solution does not lie between the two vectors (See Figure 5). Here we propose a spatial-temporal pooling method combining VA and IOC methods to solve the the aperture problem for noisy output from the SNN. Note that pooling is based on the assumption that the motion is consistent within the receptive field. The receptive field for pooling layer should be small enough to meet the assumption while at the same time be large enough to contain different edges. The details are described as follows:



(a) Type I (b) Type I (c) Type II

Figure 5: Type I and Type II stimulus

Sliding temporal window integration: First, we implemented a sliding window which integrates the nearest steps of normal flow output from the MS-layer. Since at least two different edges are needed for pooling, this implementation allows integration of more spikes in the pooling process without further enlarging the receptive field or increasing the simulation step. The latter will reduce the accuracy of the output in the MS layer.

Bin-based vector-average integration: The normal flow is determined both by the full flow and the texture pattern. IOC method gives the veridical value of the true flow from any two separate normal flows. However, given the poor speed selectivity and erroneous firings for the kernels of similar directions, normal flow vectors of similar direction and different magnitude, named the Type II stimulus in Figure5c, appear frequently in the same receptive fields. The IOC computation of this case can deviate significantly and lead to errors. To alleviate this problem, we assign each kernel to a direction-based bin such as Figure 4b. Pooling flow is calculated using the VA method in the case that two kernels lie in the same bin.

Least square method for intersection-of-constraints solution: When neurons in the same receptive field fire corresponding to at least two different direction-based bins, the full flow is estimated by calculating the least square solution for IOC of the VA values for each bin. The problem can be formulated as: given n normal flow vectors (u_i, v_i), find $\vec{v_{pool}}$ as the intersection of all constraint lines l_i, which is the orthogonal line to the flow vector. Each flow vector lies in a line l_i passed through the original point and (u_i, v_i):

$$l_i : v_i x - u_i y = 0 \tag{4}$$

The constraint line l_i^T should meet:

$$l_i^T : -k u_i x - k v_i y + k(u_i^2 + v_i^2) = 0 \tag{5}$$

where k is a constant factor. Given n constraint lines, the intersection of lines (x, y) can be represented in a matrix form:

$$\begin{bmatrix} -u_1 & -v_1 & u_1^2 + v_1^2 \\ -u_2 & -v_2 & u_2^2 + v_2^2 \\ \vdots & \vdots & \vdots \\ -u_n & -v_n & u_n^2 + v_n^2 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = 0 \quad \text{where} \quad A = \begin{bmatrix} -u_1 & -v_1 & u_1^2 + v_1^2 \\ -u_2 & -v_2 & u_2^2 + v_2^2 \\ \vdots & \vdots & \vdots \\ -u_n & -v_n & u_n^2 + v_n^2 \end{bmatrix} \quad X = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{6}$$

Given the linear homogeneous equation $AX = 0$, we can get the IOC solution by finding the X that minimizes $\|AX\|^2$ subject to $\|X\| = 1$, which should be the eigen vector of $A^T A$ associated with the smallest eigenvalue.

6.3 Ego-motion estimation using optical flow

6.3.1 Motion field model of rigid static scene

We would like to present the basic concepts and the formation of retina motion (motion field) in response to camera instantaneous velocities here since it forms the basis for our ego-motion estimation method. The motion field model of rigid scene was derived previously by [61]. Figure

6 shows the projection of 3D world point P on the image plane p from the Longuet-Higgins Prazdny model. In the camera reference frame, with the origin as the projection center and Z axis as the optical axis, the image plane is perpendicular to the optical axis, which is the surface of the projection of 3D world points. The intersection of the optical axis Z and the image plane is the principal points $O(0, 0, f)^T$ in the camera frame. f is the focal length. We denote the image point $p(x, y)$ as the projection of scene point $P(X, Y, Z)^T$ in the 3D world. For an ideal pinhole camera model, their relationship can be expressed as:

$$\vec{p} = f\frac{\vec{P}}{Z} \tag{7}$$

Take the time derivative of both sides of Equation 7, we obtain the relation between the velocity $\vec{V}$ of world point P and the velocity of the image point, which is the motion field $\vec{v} = (u, v)$:

$$\vec{v} = f\frac{Z\vec{V} - V_z\vec{P}}{Z^2} \tag{8}$$

Suppose the camera is moving with a translation velocity $\vec{T} = (T_x, T_y, T_z)$ and angular velocity $\vec{\omega} = (\omega_x, \omega_y, \omega_z)$. The relative motion $\vec{V}$ between world point P and camera can be described as:

$$\vec{V} = -(\vec{T} + \omega \overset{\rightharpoonup}{\times} \vec{P}) \tag{9}$$

The motion field model can be derived combining Equations 8 and 9. We write it in a matrix representation, for each optical flow vector (u_i, v_i):

$$\begin{bmatrix} u_i \\ v_i \end{bmatrix} = \begin{bmatrix} -\frac{f}{Z_i} & 0 & \frac{x_i}{Z_i} \\ 0 & -\frac{f}{Z_i} & \frac{y_i}{Z_i} \end{bmatrix} \begin{bmatrix} T_x \\ T_y \\ T_z \end{bmatrix} + \begin{bmatrix} \frac{x_i y_i}{f} & -\frac{f^2+x_i^2}{f} & y_i \\ \frac{f^2+y_i^2}{f} & -\frac{x_i y_i}{f} & -x_i \end{bmatrix} \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} = \begin{bmatrix} A_{Ti} & A_{\omega i} \end{bmatrix} \begin{bmatrix} \vec{T} \\ \vec{\omega} \end{bmatrix} \tag{10}$$

where $A_{Ti} = \begin{bmatrix} -\frac{f}{Z_i} & 0 & \frac{x_i}{Z_i} \\ 0 & -\frac{f}{Z_i} & \frac{y_i}{Z_i} \end{bmatrix}$, $A_{\omega i} = \begin{bmatrix} \frac{x_i y_i}{f} & -\frac{f^2+x_i^2}{f} & y_i \\ \frac{f^2+y_i^2}{f} & -\frac{x_i y_i}{f} & -x_i \end{bmatrix}$.

The motion field in the image plane is the result of ego-motion of the observer moving in a rigid static scene. The motion field vector is the sum to the translation-dependent component and the rotational component. Notice that the translational component is depth-dependent and the rotational part does not carry information of depth. Motion fields depend on the six parameters of the 3D motion and also the surface of the 3-D points in the world. If depth Z_i is known, 3-D velocities can be solved by choosing at least three non-collinear points while using more data points is useful for noisy input. Without the depth information, from the motion fields, the 3-D ego-motion can be only determined up to a scaling factor.

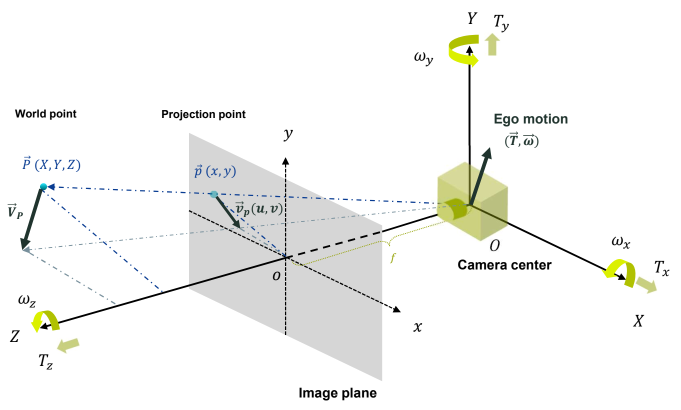


Figure 6: Longuet-Higgins Prazdny model

6.3.2 Pure rotational motion

For the pure rotational motion above the Z axis, given at least two image points and their optical flow, the rotational components can be optimized in one step by solving the least squares optimization problem as follows:

$$\vec{v_p} = \begin{bmatrix} u_1 \\ v_1 \\ u_2 \\ v_2 \\ \vdots \\ u_n \\ v_n \end{bmatrix} = A_\omega \vec{\omega} = \begin{bmatrix} A_{\omega 1} \\ A_{\omega 2} \\ \vdots \\ A_{\omega n} \end{bmatrix} \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} \tag{11}$$

where $\vec{v_p}$ is a $2n$ dimension vector, A_ω is a $2n \times 3$ matrix.

6.3.3 Pure translational motion

For the pure translation case, the motion fields depend not only upon the camera motion but also upon the depth value at each corresponding point. If the depth is known, translational velocity can be estimated using the same method as pure rotation case. However, Without the depth information, from the motion fields, only the translational direction can be determined. If T_z is not zero, the translational component can be written as

$$u = \frac{(x - x_0)T_z}{Z} \qquad v = \frac{(x - y_0)T_z}{Z} \tag{12}$$

The point (x_0, y_0) is called the vanishing point of the translational direction, which is the location in the image plane independent of the depth information. The magnitude of the flow vector is proportional to the distance between the point p and the vanishing point p_0 and inversely proportional to the depth. Specifically, if $T_z < 0$, p_0 is the focus of expansion (FoE) as all the motion field vectors point away from it, while if $T_z > 0$, p_0 is the focus of contraction (FoC). The vanishing point provides information about the direction of the flow, which does not depend upon the depth. By coplanarity constraint we know that image point $\vec{p}(x, y, f)$, flow vector $\vec{v_p}(u, v)$, and linear velocity lie on the same plane (see Figure 6):

$$(\vec{p} \times \vec{v_p})^T \vec{T} = 0 \tag{13}$$

With n image points and their corresponding flow vectors we can get a homogeneous system:

$$\mathbf{A}\tilde{\mathbf{T}} = \begin{bmatrix} (\vec{p_1} \times \vec{v_{p1}})^T \\ (\vec{p_2} \times \vec{v_{p2}})^T \\ \vdots \\ (\vec{p_n} \times \vec{v_{pn}})^T \end{bmatrix} \tilde{\mathbf{T}} = \mathbf{0} \tag{14}$$

In practice, we constrain the translational vector $\vec{T}$ to have unit magnitude, the translational direction can be determined by choosing the eigenvector corresponding to the smallest eigenvalue obtained by singular value decomposition (SVD) [64].

6.3.4 RANSAC for robust estimation

As the motion field output from the neural network is noisy, instead of using least squares method [64], we implement a RANdom SAmple Consensus (RANSAC) algorithm for robust motion estimation. We consider both average endpoint error (AEE) and average angle error (AAE) between the input flow and the model flow. The outline of the algorithm is presented as follows:

Algorithm 1: RANSAC for motion estimation

```
Data: $num\_pts = n$
Input: Pooling layer points $\vec{p} \in \mathbb{R}^{2\times n}$, pooling flow $\vec{v_p} \in \mathbb{R}^{2\times n}$
Output: Ego motion vector $\vec{M} = (\vec{T}, \vec{\omega})$
1  if $(n > min\_pts)$ then
2      while $(cnt\_iterations < k\_th)$ do
3          sample_points $\vec{p_{sp}} \in \mathbb{R}^{2\times m}$   collinearity check( $\vec{p_{sp}}$) = false
4          sample_model $\vec{M_{sp}}$ = model parameters fitted to $\vec{p_{sp}}$
5          for $\vec{p_i} = 1 : n$ do
6              $AEE_i = \|\vec{v_{ex}} - \vec{v_{pi}}\|$   $AAE_i = \arccos \frac{\vec{v_{ex}} \cdot \vec{v_{pi}}}{\|\vec{v_{ex}}\|\|\vec{v_{pi}}\|}$
7              if $(AEE_i < AEE_{th})\&\&(AAE_i < AAE_{th})$ then
8                  $\vec{p_i} \in \vec{p_{inliers}}$
9                  $cnt\_inliners + +$
10         if $cnt\_inliners > inliner\_th$ then
11             $cnt\_inliners = d$
12             Inliner_model $\vec{M_{inliers}}$ = model parameters fitted to $\vec{p_{inliers}}$
13             $\overline{err} = \frac{1}{d} \sum_{i=1}^{d} (\frac{AEE_i}{AEE_{th}} + \frac{AAE_i}{AAE_{th}})$
14             if $\overline{err} < err\_best$ then
15                 Fit_model $\vec{M}$ = $\vec{M_{inliers}}$
16                 $err\_best = err$
17         $cnt\_iterations + +$
18     return $\vec{M}, err\_best$
```

6.4 Results

6.4.1 Datasets and implementation details

The datasets for tests are generated by ESIM simulator [70]. We created several synthetic dataset using planar scene render with two patterns: checkerboard and windmills (shown in Figure 7a). The planar scene is placed at the depth of 1 meter in camera reference frame. The camera intrinsic parameter is $(200, 200, 120, 90)$ with the focal length of 200 $pixels$. We defined several trajectories to simulate different camera motion. To facilitate the evaluation of our algorithms, we first tested our system with pure constant rotational motion and pure constant translational motion (depth value is set to be 1 meter). For pure constant rotational motion, we generated the dataset with the rotation speed of 0.57 rad/s above the z axis in the camera reference frame, which is close to the original dataset used for training. Similarly, for pure translational motion tests, we generated a dataset corresponding to the diagonal translational motion $(0.18, -0.18\ m/s)$ in the camera reference frame. To estimate the translational direction with unknown depth, we use 3D scene which involving depth variations (refer to Figure 11a).
The implementation is carried out with C++ using the cuSNN library. All the tests are run with simulation step of 1 ms with a 500 ms data sequence. The parameters tuned for the SNN architecture and Ransac algorithms are shown in Table 1. Since α is an inhibition term in the neural network and is related to the firing times. We set a smaller value for α to get enough spikes for the estimation method. However, to compromise the decrease of accuracy, we increase the firing threshold V_{th} and decay time τ_m. The optimal pooling receptive field depends on the texture of the pattern and it should meet the assumption of consistent motion. Increasing the receptive field size

can include different edges and give more accurate full flow results after pooling, while smaller value allows more flow vectors in the pooling layer for ego-motion estimation. Depending on the testing motion statistics and the test pattern used, we set the sliding integration window size to 10 ms and bin size to 8.

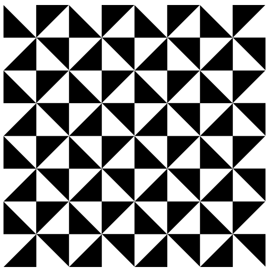
Layer	V_{th}	τ_m	α	rf	bin	W
	mv	ms	–	ms	–	ms
MS	0.8	40	0.1	5	–	–
Pooling	0	5	1	7	8	10

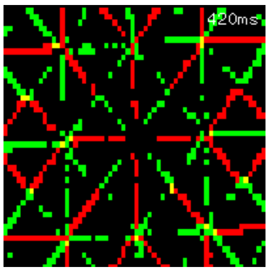
(a) Parameters for SNN

Motion	k	$inlier_{th}$	AEE_{th}	AAE_{th}
	–	–	$pixels$	$degree$
Trans	20	7	22	0.18
Rot	20	8	50	0.5

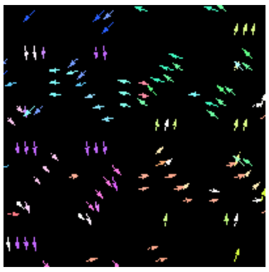
(b) Parameters for Ransac

Table 1: Parameters used for windmills pattern


(a) Windmills pattern



(b) Input


(c) Normal flow

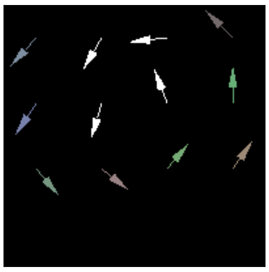
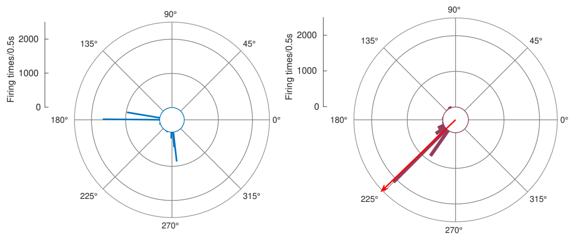

(d) Full flow

Figure 7: SNN simulation results: 7a shows the windmills pattern we use for event datasets generation. 7b shows the input events, ON and Off polarity is encoded in green and red color respectively. 7c and 7d present the normal flow output from the MS-layer and full flow from the pooling layer. All the flow vectors have been normalized for plotting. Direction and speed are encoded with different colors according to Figure 4b. Outliers rejected by Ransac algorithms are shown in write color.

6.4.2 Pooling flow evaluation

Figure 8 shows the responses of the MS-layer neurons and pooling layer neurons in orientation space for the pure translational motion test. The ground truth flow direction for the simulation dataset is 225 °. The checkerboard pattern contains only horizontal and vertical edges while the windmills pattern contains edges of eight different orientations and presents more noisy normal flow results. In both cases, the normal flows in the vertical direction are deviated from its ground-truth value (270 °). That is the result of insufficient kernels that match the exact orientation and magnitude. It also explains the greater deviation for T_y compared to T_x in the translational motion test result (refer to Table 3). Despite the limitation of the kernels available, the pooling layer gives acceptable full flow results from the integration of the normal flow in both cases.
Quantitative evaluation of the AEE and AAE between the pooling flows and the ground truth flows are shown in Table 2. Note that in pure translational motion case, the flow vectors are same for each pixel, while the rotational flow vectors include more variety and shows more errors in both optical flow values and the motion estimation results.


(a) Pure translation motion for checkerboard pattern

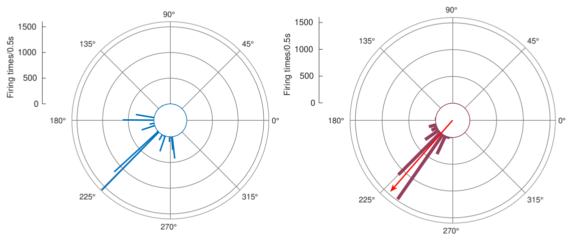
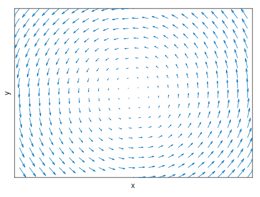
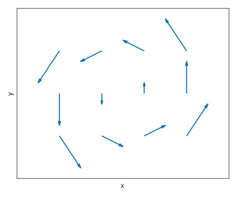

(b) Pure translation motion for windmills pattern

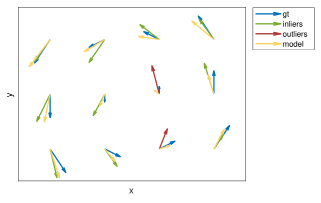
Figure 8: Histograms of neurons firing counts in orientation space: the left graph in blue of each figure represents the histogram of firing counts for MS (normal flow) layer, while the right graph in red represents the firing of the neurons in pooling layer. The red arrow shows the average flow direction.

6.4.3 Ransac ego-motion estimation evaluation

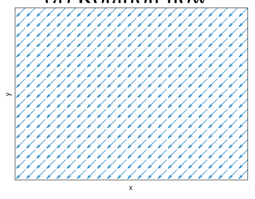
The simulation result for pure constant rotation case with windmills pattern is shown in Figure 7. Figure 9 shows the qualitative results of the full flow in the pooling layer for both rotational and translational motion cases. The Ransac algorithm is able to reject the outliers, which are the flows that cannot match the model flow due to errors either in magnitude or direction.

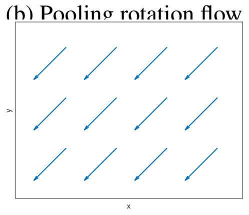

(a) Rotation flow

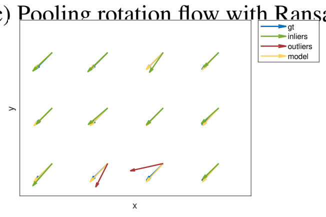

(b) Pooling rotation flow



(c) Pooling rotation flow with Ransac


(d) Translation flow


(e) Pooling translation flow



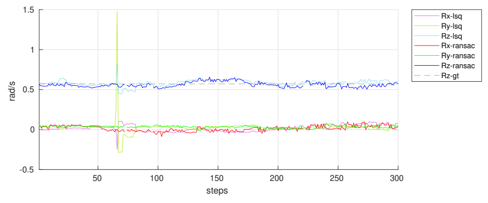
(f) Pooling translation flow with Ransac

Figure 9: Qualitative results for pure rotational and pure translational flow: Figure 9a and Figure 9b shows the ground truth for rotational flow and its corresponding pooling flow. Figure 9c shows one random step of the pooling flow test result. Figure 9d and Figure 9e show the ground truth for translational flow and its corresponding pooling flow. 9f shows one random step of the pooling flow test result. The ground truth flows are plotted in blue color and model flows from Ransac are in yellow. Outputs of the SNN pooling layer are indicated in green and red, where green indicates inliers and red means outliers.

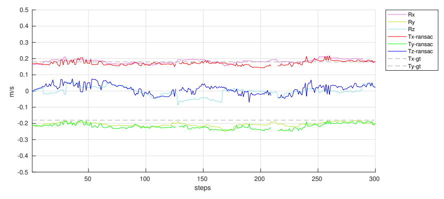
Optical flow rotation	AEE rad/s	AAE $\circ$	**Optical flow** translation	AEE $pixel$	AAE $\circ$
without ransac	30.585	19.086	**without ransac**	12.304	9.729
with ransac	21.240	14.892	**with ransac**	10.146	8.106

Table 2: Qualitative results for optical flow estimation

The quantitative results for ego motion of all the 300 ms steps with available motion update are shown in Table 3 and plotted in Figure 10. Our method using Ransac shows robustness compared to least squares solution.



(a) Pure rotational motion



(b) Pure translation motion

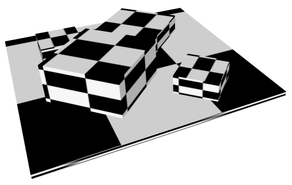
Figure 10: Motion estimation tests results: Figure 10a and figure 10b show the results for pure constant rotational motion tests and pure constant translation motion tests, respectfully. The results obtained from our methods are presented in red, green and blue, corresponding to the motion above x, y and z axis, while the results using least square methods are also plotted in lighter colors for comparison. Ground truth values are plotted with dot lines.

	Rotation motion			**Translation motion**		
	R_x rad/s	R_y rad/s	R_z rad/s	T_x m/s	T_y m/s	T_z m/s
GT	0	0	0.5712	0.18	- 0.18	0
Mean	0.01	0.0345	0.5625	0.172	- 0.218	0.0128
RMSD	0.0334	0.0161	0.0337	0.0131	0.0162	0.0294
RMSE	0.0357	0.0377	0.0342	0.0156	0.0413	0.0321

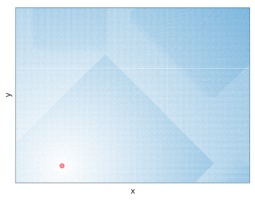
Table 3: Quantitative ego-motion test results

6.4.4 Pure translational direction estimation

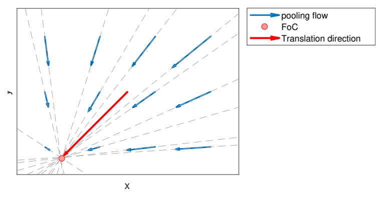
We created a dataset with pure linear velocity of $(0.18, -0.18, -0.5)m/s$ in the camera reference frame for both planar scene with the windmills pattern, and 3D scene with simulated checkerboard pattern boxes to introduce unknown depth. Without any knowledge of the depth information, our method can estimate the translational velocity up to a scale. Figure 11c shows the simulated optical flow for the 3D scene. The intersection of all the flow vectors where the flow velocity is zero indicates the translational direction. We estimate the translation result without applying Ransac. The outputs are normalized to a unit vector and compared to the ground-truth values in Figure 12 and Table 4.



(a) 3D scene with various depth of checkerboard pattern boxes

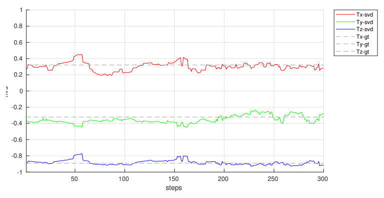


(b) Pure translation flow

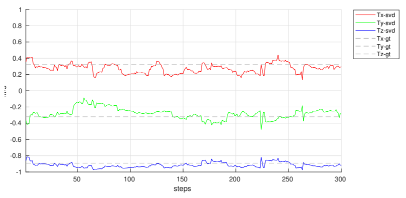


(c) Pooling translation flow for 3D scene

Figure 11: Pure translational flow for 3D scene: Figure 11a shows the 3D scene used for simulation. Figure 11b and 11c plot the simulated flow and pooling flow with the blue vectors. The red point represents the FoC, which indicates the translational direction (shown in the red vector).



(a) Planar scene with windmills pattern



(b) 3D scene

Figure 12: Translational direction estimation tests results: Figure 12a and Figure 12b show the results for pure constant translational motion tests for planar scene and 3D scene, respectively. The results obtained from our methods are presented in red, green and blue, corresponding to the motion above x, y and z axis. Ground truth values are plotted with dot lines.

	Translation motion-planar			**Translation motion-3D**		
	T_x	T_y	T_z	T_x	T_y	T_z
	normalized direction			normalized direction		
GT	0.3208	-0.3208	-0.8912	0.3208	-0.3208	-0.8912
Mean	0.3027	-0.3530	-0.8821	0.2749	-0.2801	-0.9151
RMSD	0.0511	0.0473	0.0305	0.0571	0.0667	0.0304
RMSE	0.0541	0.0571	0.0318	0.0732	0.0780	0.0386

Table 4: Qualitative test results for translational direction estimation

6.5 Discussion

We explored the feasibility to use spiking neural network for ego-motion estimation. In this initial step, we exploited a trained SNN framework to perform ego motion estimation. Tests on different dataset have been carried out to verify and evaluate our method. The performance of our implementation is limited by the statistics of the trained kernels, which is the main impediment in realizing our tests with more complex motions since they rely on flow vectors with more variety out of the range of the trained kernels available.

As the neural network is trained with a fully unsupervised fashion from one simple synthetic dataset sequence, noises occur from the read-out mechanism for the trained model and during erroneous firing of the neurons in the inference phase. Especially, the poor speed selectivity of the MS-layer neurons challenges the IOC method to integrate the full flow as it is sensitive to the magnitude of the normal optical flow with similar directions. In addition, due to the sparse properties of the event camera, the flow vectors are texture and motion dependent. We tuned the neural network considering trade-off among simulation speed, accuracy and sparsity. Given the noisy and sparse normal flows input, we have implemented an optimization-based pooling technique with robust motion estimation method. Our motion estimation method is based on rigid static motion, and does not rely on any assumption of the input data. Despite the inherent limitation of the filter-based optical flow estimation method, our method was able to output reliable results on pure rotational and translational ego-motion tests.

In the future work, we would like to extend the work for 3-D ego motion estimation. We intend to refine the normal optical flow estimation by exploring other SNN architectures and learning methods. More complex motion dataset on real world scenes will be employed for validation.

References

[1] D. Nistér, O. Naroditsky, and J. Bergen, "Visual odometry," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 1, 2004.

[2] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "ImageNet Large Scale Visual Recognition Challenge," *International Journal of Computer Vision*, vol. 115, no. 3, 2015.

[3] K. Roy, A. Jaiswal, and P. Panda, "Towards spike-based machine intelligence with neuromorphic computing," *Nature*, vol. 575, no. 7784, pp. 607–617, 2019. [Online]. Available: http://dx.doi.org/10.1038/s41586-019-1677-2

[4] M. A. Goodale and A. D. Milner, "Separate visual pathways for perception and action," 1992.

[5] D. H. Hubel and T. N. Wiesel, "Receptive fields, binocular interaction and functional architecture in the cat's visual cortex," *The Journal of Physiology*, vol. 160, no. 1, 1962.

[6] G. James Jerome, "The Perception of the Visual World." *Boston: Houghton Mifflin.*, 1950.

[7] C. J. Duffy and R. H. Wurtz, "Sensitivity of MST neurons to optic flow stimuli. I. A continuum of response selectivity to large-field stimuli," *Journal of Neurophysiology*, vol. 65, no. 6, 1991.

[8] M. S. Graziano, R. A. Andersen, and R. J. Snowden, "Tuning of MST neurons to spiral motions," *Journal of Neuroscience*, vol. 14, no. 1, 1994.

[9] P. Lichtsteiner, C. Posch, and T. Delbruck, "A 128 × 128 120 dB 15 μs latency asynchronous temporal contrast vision sensor," *IEEE Journal of Solid-State Circuits*, vol. 43, no. 2, pp. 566–576, feb 2008.

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

[11] W. Maass, "Networks of spiking neurons: The third generation of neural network models," *Neural Networks*, vol. 10, no. 9, pp. 1659–1671, dec 1997.

[12] P. U. Diehl and M. Cook, "Unsupervised learning of digit recognition using spike-timing-dependent plasticity," *Frontiers in Computational Neuroscience*, vol. 0, no. AUGUST, p. 99, aug 2015.

[13] S. M. Bohte, J. N. Kok, and H. La Poutré, "Error-backpropagation in temporally encoded networks of spiking neurons," *Neurocomputing*, vol. 48, no. 1-4, pp. 17–37, oct 2002.

[14] C. Lee, S. S. Sarwar, P. Panda, G. Srinivasan, and K. Roy, "Enabling Spike-Based Backpropagation for Training Deep Neural Network Architectures," *Frontiers in Neuroscience*, vol. 0, p. 119, feb 2020.

[15] A. Vitale, A. Renner, C. Nauer, D. Scaramuzza, and Y. Sandamirskaya, "Event-driven Vision and Control for UAVs on a Neuromorphic Chip," in *IEEE International Conference on Robotics and Automation (ICRA) 2021*, 2021.

[16] G. Haessig, F. Galluppi, X. Lagorce, and R. Benosman, "Neuromorphic networks on the SpiNNaker platform," *Proceedings 2019 IEEE International Conference on Artificial Intelligence Circuits and Systems, AICAS 2019*, pp. 86–91, 2019.

[17] G. Orchard and R. Etienne-Cummings, "Bioinspired visual motion estimation," *Proceedings of the IEEE*, vol. 102, no. 10, pp. 1520–1536, 2014.

[18] G. Haessig, A. Cassidy, R. Alvarez, R. Benosman, and G. Orchard, "Spiking Optical Flow for Event-Based Sensors Using IBM's TrueNorth Neurosynaptic System," *IEEE Transactions on Biomedical Circuits and Systems*, vol. 12, no. 4, pp. 860–870, 2018.

[19] H. Akolkar, S. Panzeri, and C. Bartolozzi, "Spike time based unsupervised learning of receptive fields for event-driven vision," in *Proceedings - IEEE International Conference on Robotics and Automation*, vol. 2015-June, no. June, 2015.

[20] L. Deng, K. Huang, Y. Xing, G. D. Caterina, and J. Soraghan, "A New Spiking Convolutional Recurrent Neural Network (SCRNN) With Applications to Event-Based Hand Gesture Recognition," *Frontiers in Neuroscience — www.frontiersin.org*, vol. 14, p. 590164, 2020.

[21] G. Debat, T. Chauhan, B. R. Cottereau, T. Masquelier, M. Paindavoine, and R. Baures, "Event-Based Trajectory Prediction Using Spiking Neural Networks," *Frontiers in Computational Neuroscience*, vol. 15, may 2021.

[22] B. D. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision," in *Proceedings of the International Joint Conference on Neural Networks*, vol. 2, 1981, pp. 674–679.

[23] S. S. Beauchemin and J. L. Barron, "The Computation of Optical Flow," *ACM Computing Surveys (CSUR)*, vol. 27, no. 3, pp. 433–466, 1995.

[24] R. Benosman, S. H. Ieng, C. Clercq, C. Bartolozzi, and M. Srinivasan, "Asynchronous frameless event-based optical flow," *Neural Networks*, vol. 27, pp. 32–37, 2012.

[25] T. Brosch, S. Tschechne, and H. Neumann, "On event-based optical flow detection," *Frontiers in Neuroscience*, vol. 9, no. APR, 2015.

[26] M. Almatrafi, R. Baldwin, K. Aizawa, and K. Hirakawa, "Distance Surface for Event-Based Optical Flow," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 42, no. 7, pp. 1547–1556, 2020.

[27] 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.

[28] B. J. Pijnacker Hordijk, K. Y. Scheper, and G. C. de Croon, "Vertical landing for micro air vehicles using event-based optical flow," *Journal of Field Robotics*, vol. 35, no. 1, pp. 69–90, 2018.

[29] M. T. Aung, R. Teo, and G. Orchard, "Event-based Plane-fitting Optical Flow for Dynamic Vision Sensors in FPGA," in *Proceedings - IEEE International Symposium on Circuits and Systems*, vol. 2018-May, 2018.

[30] H. Akolkar, S. H. Ieng, and R. Benosman, "Real-time high speed motion prediction using fast aperture-robust event-driven visual flow," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1–1, 2020.

[31] E. Mueggler, C. Forster, N. Baumli, G. Gallego, and D. Scaramuzza, "Lifetime estimation of events from Dynamic Vision Sensors," *Proceedings - IEEE International Conference on Robotics and Automation*, vol. 2015-June, no. June, pp. 4874–4881, jun 2015.

[32] B. Rueckauer and T. Delbruck, “Evaluation of event-based algorithms for optical flow with ground-truth from inertial measurement sensor,” *Frontiers in Neuroscience*, vol. 10, no. APR, apr 2016.

[33] M. Z. Khairallah, F. Bonardi, D. Roussel, and S. Bouchafa, “PCA Event-Based Optical Flow for Visual Odometry,” may 2021. [Online]. Available: http://arxiv.org/abs/2105.03760

[34] L. I. Abdul-Kreem and H. Neumann, “Estimating Visual Motion Using an Event-Based Artificial Retina,” in *Computer Vision, Imaging and Computer Graphics Theory and Applications. VISIGRAPP 2015.* Springer, Cham, 2016, pp. 396–415.

[35] G. Orchard, R. Benosman, R. Etienne-Cummings, and N. V. Thakor, “A spiking neural network architecture for visual motion estimation,” *2013 IEEE Biomedical Circuits and Systems Conference, BioCAS 2013*, no. 1, pp. 298–301, 2013.

[36] F. Paredes-Valles, K. Y. W. Scheper, and G. C. H. E. De Croon, “Unsupervised Learning of a Hierarchical Spiking Neural Network for Optical Flow Estimation: From Events to Global Motion Perception,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 8828, no. c, pp. 1–1, 2019.

[37] F. Peveri, S. Testa, and S. P. Sabatini, “A Cortically-inspired Architecture for Event-based Visual Motion Processing: From Design Principles to Real-world Applications,” in *CVPR 2021 Workshops*, 2021.

[38] R. L. De Valois, N. P. Cottaris, L. E. Mahon, S. D. Elfar, and J. A. Wilson, “Spatial and temporal receptive fields of geniculate and cortical cells and directional selectivity,” *Vision Research*, vol. 40, no. 27, pp. 3685–3702, dec 2000.

[39] B. Von Hassenstein and W. Reichardt, “Systemtheoretische Analyse der Zeit-, Reihenfolgen- und Vorzeichenauswertung bei der Bewegungsperzeption des Rüsselkäfers Chlorophanus,” *Zeitschrift fur Naturforschung - Section B Journal of Chemical Sciences*, vol. 11, no. 9-10, 1956.

[40] A. Zhu, L. Yuan, K. Chaney, and K. Daniilidis, “EV-FlowNet: Self-Supervised Optical Flow Estimation for Event-based Cameras,” 2018.

[41] A. Z. Zhu, L. Yuan, K. Chaney, and K. Daniilidis, “Unsupervised event-based optical flow using motion compensation,” in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 11134 LNCS, 2019.

[42] C. Ye, A. Mitrokhin, C. Ferm, J. A. Yorke, and Y. Aloimonos, “Unsupervised Learning of Dense Optical Flow , Depth and Egomotion from Sparse Event Data,” in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018, pp. 5831–5838.

[43] T. Stoffregen, C. Scheerlinck, D. Scaramuzza, T. Drummond, N. Barnes, L. Kleeman, and R. Mahony, “Reducing the Sim-to-Real Gap for Event Cameras,” in *European Conference on Computer Vision (ECCV)*, vol. 12372 LNCS, 2020, pp. 534–549.

[44] F. Paredes-Valles and G. C. H. E. de Croon, “Back to Event Basics: Self-Supervised Learning of Image Reconstruction for Event Cameras via Photometric Constancy,” pp. 3446–3455, 2021.

[45] Z. Li, J. Shen, and R. Liu, "A Lightweight Network to Learn Optical Flow from Event Data," in *2020 25th International Conference on Pattern Recognition (ICPR)*. IEEE, jan 2021, pp. 1–7.

[46] C. Lee, A. K. Kosta, A. Z. Zhu, K. Chaney, K. Daniilidis, and K. Roy, "Spike-FlowNet: Event-Based Optical Flow Estimation with Energy-Efficient Hybrid Neural Networks," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 12374 LNCS, pp. 366–382, 2020.

[47] T. Barbier and J. Triesch, "Spike timing-based unsupervised learning of orientation, disparity, and motion representations in a spiking neural network *," *CVPR 2021 Workshops*, vol. 25, 2021.

[48] F. Paredes-Vallés, J. Hagenaars, and G. de Croon, "Self-Supervised Learning of Event-Based Optical Flow with Spiking Neural Networks," jun 2021. [Online]. Available: https://arxiv.org/abs/2106.01862v1

[49] A. Z. Zhu, D. Thakur, T. Özaslan, B. Pfrommer, V. Kumar, and K. Daniilidis, "The Multi-vehicle Stereo Event Camera Dataset: An Event Camera Dataset for 3D Perception," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 2032–2039, jul 2018.

[50] C. Lee, A. K. Kosta, and K. Roy, "Fusion-FlowNet: Energy-Efficient Optical Flow Estimation using Sensor Fusion and Deep Fused Spiking-Analog Network Architectures," mar 2021. [Online]. Available: https://arxiv.org/abs/2103.10592v1

[51] M. Gehrig, S. B. Shrestha, D. Mouritzen, and D. Scaramuzza, "Event-Based Angular Velocity Regression with Spiking Networks," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, may 2020, pp. 4195–4202.

[52] S. B. Shrestha and G. Orchard, "Slayer: Spike layer error reassignment in time," in *Advances in Neural Information Processing Systems*, vol. 2018-Decem, no. NeurIPS, 2018, pp. 1412–1421.

[53] W. H. Warren and D. J. Hannon, "Direction of self-motion is perceived from optical flow," *Nature*, vol. 336, no. 6195, 1988.

[54] K. Zhang, M. I. Sereno, and M. E. Sereno, "Emergence of Position-Independent Detectors of Sense of Rotation and Dilation with Hebbian Learning: An Analysis," *Neural Computation*, vol. 5, no. 4, 1993.

[55] R. Wang, "A simple competitive account of some response properties of visual neurons in area MSTd." *Neural computation*, vol. 7, no. 2, 1995.

[56] R. S. Zemel and T. J. Sejnowski, "A model for encoding multiple object motions and self-motion in area MST of primate visual cortex," *Journal of Neuroscience*, vol. 18, no. 1, 1998.

[57] N. G. Hatsopoulos and W. H. Warren, "Visual navigation with a neural network," *Neural Networks*, vol. 4, no. 3, 1991.

[58] J. A. Perrone, "Model for the computation of self-motion in biological systems," *Journal of the Optical Society of America A*, vol. 9, no. 2, 1992.

[59] M. Lappe and J. P. Rauschecker, "A Neural Network for the Processing of Optic Flow from Ego-Motion in Man and Higher Mammals," *Neural Computation*, vol. 5, no. 3, 1993.

[60] D. J. Heeger and A. D. Jepson, “Subspace methods for recovering rigid motion I: Algorithm and implementation,” *International Journal of Computer Vision*, vol. 7, no. 2, pp. 95–117, 1992.

[61] H. C. Longuet-Higgins and K. Prazdny, “The interpretation of a moving retinal image,” *Proceedings of the Royal Society of London - Biological Sciences*, vol. 208, no. 1173, 1980.

[62] F. Raudies and H. Neumann, “A review and evaluation of methods estimating ego-motion,” *Computer Vision and Image Understanding*, vol. 116, no. 5, pp. 606–633, may 2012.

[63] X. Zhuang, R. M. Haralick, and Y. Zhao, “From depth and optical flow to rigid body motion.” in *Proceedings - Conference on Computer Vision and Pattern Recognition (CVPR) 1988*. Publ by IEEE, 1988, pp. 393–397.

[64] A. R. Bruss and B. K. Horn, “Passive navigation.” *Computer Vision, Graphics, and Image Processing*, vol. 21, no. 1, 1983.

[65] J. H. Rieger and D. T. Lawton, “Processing differential image motion,” *Journal of the Optical Society of America A*, vol. 2, no. 2, 1985.

[66] A. Chiuso, R. Brockett, and S. Soatto, “Optimal structure from motion: local ambiguities and global estimates,” *International Journal of Computer Vision*, vol. 39, no. 3, pp. 195–228, sep 2000.

[67] T. Zhang and C. Tomasi, “On the consistency of instantaneous rigid motion estimation,” *International Journal of Computer Vision*, vol. 46, no. 1, pp. 51–79, 2002.

[68] PauwelsKarl and v. H. M., “Optimal instantaneous rigid motion estimation insensitive to local minima,” *Computer Vision and Image Understanding*, vol. 104, no. 1, pp. 77–86, oct 2006.

[69] F. Raudies and H. Neumann, “An efficient linear method for the estimation of ego-motion from optical flow,” in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 5748 LNCS, 2009, pp. 11–20.

[70] H. Rebecq, D. Gehrig, and D. Scaramuzza, “ESIM: an Open Event Camera Simulator,” in *Conf. on Robotics Learning (CoRL)*, 2018.

[71] T. Bekolay, J. Bergstra, E. Hunsberger, T. DeWolf, T. C. Stewart, D. Rasmussen, X. Choo, A. R. Voelker, and C. Eliasmith, “Nengo: A Python tool for building large-scale functional brain models,” *Frontiers in Neuroinformatics*, vol. 7, no. JAN, 2014.

[72] M. Stimberg, R. Brette, and D. F. Goodman, “Brian 2, an intuitive and efficient neural simulator,” *eLife*, vol. 8, 2019.

[73] H. Hazan, D. J. Saunders, H. Khan, D. Patel, D. T. Sanghavi, H. T. Siegelmann, and R. Kozma, “BindsNET: A machine learning-oriented spiking neural networks library in python,” *Frontiers in Neuroinformatics*, vol. 12, 2018.

[74] M. Mozafari, M. Ganjtabesh, A. Nowzari-Dalini, and T. Masquelier, “SpykeTorch: Efficient simulation of convolutional spiking neural networks with at most one spike per neuron,” *Frontiers in Neuroscience*, vol. 13, no. JUL, 2019.

[75] K. McGuire, G. De Croon, C. De Wagter, K. Tuyls, and H. Kappen, “Efficient Optical Flow and Stereo Vision for Velocity Estimation and Obstacle Avoidance on an Autonomous Pocket Drone,” *IEEE Robotics and Automation Letters*, vol. 2, no. 2, pp. 1070–1076, apr 2017.

[76] E. P. Simoncelli and D. J. Heeger, “A model of neuronal responses in visual area MT,” *Vision Research*, vol. 38, no. 5, 1998.

[77] W. Gander, G. H. Golub, and R. Strebel, “Least-squares fitting of circles and ellipses,” *BIT Numerical Mathematics 1994 34:4*, vol. 34, no. 4, pp. 558–578, dec 1994. [Online]. Available: https://link.springer.com/article/10.1007/BF01934268

[78] E. H. Adelson and J. A. Movshon, “Phenomenal coherence of moving visual patterns,” *Nature 1982 300:5892*, vol. 300, no. 5892, pp. 523–525, 1982. [Online]. Available: https://www.nature.com/articles/300523a0

[79] G. P. Caplovitz, P. J. Hsieh, and P. U. Tse, “Mechanisms underlying the perceived angular velocity of a rigidly rotating object,” *Vision Research*, vol. 46, no. 18, pp. 2877–2893, sep 2006.

[80] C. Yo and H. R. Wilson, “Perceived direction of moving two-dimensional patterns depends on duration, contrast and eccentricity,” *Vision Research*, vol. 32, no. 1, 1992.

[81] L. Bowns and D. Alais, “Large shifts in perceived motion direction reveal multiple global motion solutions,” *Vision Research*, vol. 46, no. 8-9, 2006.