



# Retina-Based Pipe-Like Object Tracking Implemented Through Spiking Neural Network on a Snake Robot

Zhuangyi Jiang<sup>1</sup>, Zhenshan Bing<sup>1\*</sup>, Kai Huang<sup>2</sup> and Alois Knoll<sup>1</sup>

<sup>1</sup> Chair of Robotics, Artificial Intelligence and Real-time Systems, Department of Informatics, Technical University of Munich, Munich, Germany, <sup>2</sup> Department of Data and Computer Science, Sun Yat-Sen University, Guangzhou, China

## OPEN ACCESS

### Edited by:

Guanghua Xu,  
Xi'an Jiaotong University, China

### Reviewed by:

Guoyuan Li,  
NTNU Ålesund, Norway  
Zonghua Gu,  
Zhejiang University, China

### \*Correspondence:

Zhenshan Bing  
bing@in.tum.de

**Received:** 01 January 2019

**Accepted:** 07 May 2019

**Published:** 29 May 2019

### Citation:

Jiang Z, Bing Z, Huang K and Knoll A (2019) Retina-Based Pipe-Like Object Tracking Implemented Through Spiking Neural Network on a Snake Robot. *Front. Neurorobot.* 13:29.  
doi: 10.3389/fnbot.2019.00029

Vision based-target tracking ability is crucial to bio-inspired snake robots for exploring unknown environments. However, it is difficult for the traditional vision modules of snake robots to overcome the image blur resulting from periodic swings. A promising approach is to use a neuromorphic vision sensor (NVS), which mimics the biological retina to detect a target at a higher temporal frequency and in a wider dynamic range. In this study, an NVS and a spiking neural network (SNN) were performed on a snake robot for the first time to achieve pipe-like object tracking. An SNN based on Hough Transform was designed to detect a target with an asynchronous event stream fed by the NVS. Combining the state of snake motion analyzed by the joint position sensors, a tracking framework was proposed. The experimental results obtained from the simulator demonstrated the validity of our framework and the autonomous locomotion ability of our snake robot. Comparing the performances of the SNN model on CPUs and on GPUs, respectively, the SNN model showed the best performance on a GPU under a simplified and synchronous update rule while it possessed higher precision on a CPU in an asynchronous way.

**Keywords:** neuromorphic vision, spiking neural network, snake robot, Hough transform, target tracking

## 1. INTRODUCTION

Target tracking performed on mobile robots, such as bio-inspired snake robots, remains a challenging research topic. Specifically, when using visual approaches based on the conventional vision sensor which has a rigid connection to a mobile robot, there are mainly two challenges: (1) The primary issue is how to overcome the image blur resulting from the fast motion and the unpredictable tremble of the robot. Meanwhile, if there was no change in a scene, the traditional camera with a fixed frame rate would bring a large quantity of redundant data, which constraints the design and application of real-time tracking approaches. (2) Another issue is that the relative position of the target cannot be obtained fast and precisely from the sensors assembled on the robot, including IMU sensors, vision sensors, and time-of-flight sensors. In addition, owing to the limitation of space and weight of the real snake robot, it is usually difficult to utilize more sensors with higher precision and larger volume or stereo vision sensors for gaining depth information.

There have been extensive articles aiming to solve the aforementioned problems in visual target tracking on robots. A natural solution for tracking on blurred image sequence is to first perform deblurring and then apply tracking algorithm on the deblurred sequence. An improved method is directly tracking the target without deblurring (Jin et al., 2005). By generating blur templates of the target from blur-free frames, the target is represented by a sparse matrix and tracked by a particle filter (Wu et al., 2011; Ma et al., 2016). Although these frameworks are blur-tolerant, they are still time-consuming. An alternative approach performed on mobile robots is tracking objects in special color. Hu et al. (2009) designed a vision-based autonomous robotic fish and implemented red-ball tracking. However, this method cannot be used in a complex environment or for objects with low color contrast. Recently, researchers have attempted various new types of vision sensors in target tracking, such as structured light sensors (Ponte et al., 2014) and neuromorphic vision sensors (NVS) (Schraml et al., 2010; Glover and Bartolozzi, 2016; Liu et al., 2016; Moeyns et al., 2016; Seifozzakerini et al., 2016).

Tracking by using the neuromorphic vision sensors has become a promising solution. The NVS, typically the Dynamic Vision Sensor (DVS) (Lichtsteiner et al., 2008), mimics the biological retina to generate spikes in the order of microseconds in response to the pixel-level changes of brightness caused by motion. An output event (also named as a spike) of the DVS carries three kinds of information, including the timestamp  $t$  when the event occurred, the pixel coordinate  $(x, y)$ , and the polarity  $p$  that represents the trend of the brightness change. The polarity 1 represents increasing brightness, while the polarity -1 means the brightness is decreasing. NVSs offer significant advantages over standard frame-based cameras, namely a very high dynamic range, no motion blur, and a latency in the order of microseconds (Gehrig et al., 2018). Hence, the NVS is suitable for working under bad light conditions and on high-speed mobile platforms. There has been substantial research showing the advantages of using a DVS camera in various vision tasks, such as high-speed target tracking (Drazen et al., 2011; Mueggler et al., 2014; Lagorce et al., 2015), object recognition (Kheradpisheh et al., 2018), and visual odometry (Kueng et al., 2016; Rebecq et al., 2017). Moreover, due to the fact that a pixel of an NVS is a silicon retinal neuron and an event is a unit impulse with polarity, the asynchronous event train generated by an NVS can be directly fed into Spiking Neural Networks (SNNs) as input spikes for implementing target detecting and tracking in a faster and more brain-like way.

The wheel-less snake robot (Wright et al., 2007) is a kind of typical bio-inspired mobile robot, which is composed of many modules alternately connected in vertical and horizontal planes. Its abundant degrees of freedom help it achieve various three-dimensional gaits, such as rolling, side-winding and slithering. The slithering gait is a forward locomotion gait where the biological snakes use undulations to push their bodies forward (Hu and Shelley, 2012). Under this gait, the snake head can still remain stable to locate the moving direction of the quarry or the natural enemies. Similarly, the wheel-less snake robot is able to move and look forward under a slithering gait and achieve target tracking (Bing et al., 2017).

In this work, we presented a pipe-like object detecting and autonomous tracking framework, which was performed on our wheel-less snake robot with a monocular DVS camera by applying a spiking neural network which is inspired by the Hough transform (Wiesmann et al., 2012; Seifozzakerini et al., 2016). First, we achieved line detection for a standing pipe and circle detection for a lying pipe on the snake robot in the Neurorobotics Platform (NRP). The fixed connections between the input neurons corresponding to pixels of DVS and neurons representing the points in parameter space were created according to the principle of the Hough transform. Secondly, a depth estimation method based on a monocular DVS was proposed to estimate the pose of the snake robot and the relative position of the target pipe by the change of object size. Thirdly, an adaptive tracking strategy which generates a series of control signals of turning left or turning right was adopted to implement real-time tracking. Finally, target tracking experiments were conducted on the wheel-less snake robot modeled in V-REP and in NRP, respectively; and our SNN model was evaluated on CPUs and on GPUs, respectively.

This paper is based on our previous work (Jiang et al., 2017), which we extend in several ways:

- Besides V-REP, we validated our tracking framework in another simulator - NRP.
- We extend the range of shapes to detect and track so that the snake robot can track a target pipe in various views. We not only detected and tracked the standing pipe (shape in lines) but also the lying pipe (shape in circle).
- We revised conditions applied in detecting the standing pipe, which is more biologically plausible.
- GPUs were used as accelerators to speed up object detecting on the SNN.

The rest of the paper is organized as follows. In section 2, we describe the proposed tracking framework, including the overview, the model of spiking neural networks for detecting and the relative position estimation algorithm. In section 3, we show and discuss the results of experiments conducted on a wheel-less snake robot. The conclusions are drawn in section 4.

## 2. METHODOLOGY

### 2.1. Tracking Framework

Target tracking is a typical instance of the autonomous locomotion control. Therefore, tracking framework consists of three components: sensing, planning and acting, which are also the essential components of an autonomous system (Ponte et al., 2014). More concretely, the proposed framework for pipe-like object tracking on a wheel-less snake robot is composed of 4 constituents, as shown in **Figure 1**. (1) Sensor. The DVS camera observes the environment and generates asynchronous events as the input of the SNN. For an event  $e(t, x, y, p)$ ,  $(x, y)$  indicates which neuron receives this input spike,  $t$  is the time when the spiking neuron receives this input spike, and  $p$  ( $\pm 1$ ) defines the voltage of this input spike. Meanwhile, the joint encoder records the position of each joint of the snake robot in a short time period. (2) Spiking Neural Network. A two-layer SNN was designed for

object detecting. A neuron in the input layer fixedly connected some neurons in the output layer according to the principle of Hough Transform. The asynchronous events were fed into the input layer of the SNN and impacted the neurons in the output layer by propagating the spikes on the synapses. Once any output neuron excited, a output spike was generated which means a successful detection of the target. (3) Decision maker. It is a non-spiking part. The joint position information obtained from the joint encoders as well as the target position obtained from the SNN were fused to estimate the relative position of target and generate control signals. The function of control signals we used is essentially the same as a Bang-Bang controller. (4) CPG controller. This part is a built-in controller of the wheel-less snake robot that converts the control signals into the parameters of the Central Pattern Generator (CPG) to maintain or adjust the specific locomotion gait.

## 2.2. Model of Spiking Neural Network

Compared to traditional artificial neural networks, spiking neural networks are more similar to the biological brains due to incorporating the concepts of spike-driven synaptic dynamics and temporal dynamics. Temporal dynamics mean the membrane potential of spiking neurons changes spontaneously over time and spike-driven synaptic dynamics describe the information propagation on synapses.

For the neuron model, the most popular one is the Leaky Integrate-and-Fire (LIF) model (Burkitt, 2006), which can be described by the following linear differential equation:

$$\tau_m \frac{dv}{dt} = -v(t) + RI(t). \quad (1)$$

where  $v(t)$  represents the membrane potential at time  $t$ ,  $\tau_m$  is the time constant and  $R$  is the membrane resistance. A LIF neuron

is a simple resistor-capacitor circuit where the leakage term  $\frac{-v(t)}{\tau_m}$  is due to the resistor and the integration term  $\frac{RI(t)}{\tau_m}$  is due to the capacitor that is parallel to the resistor.

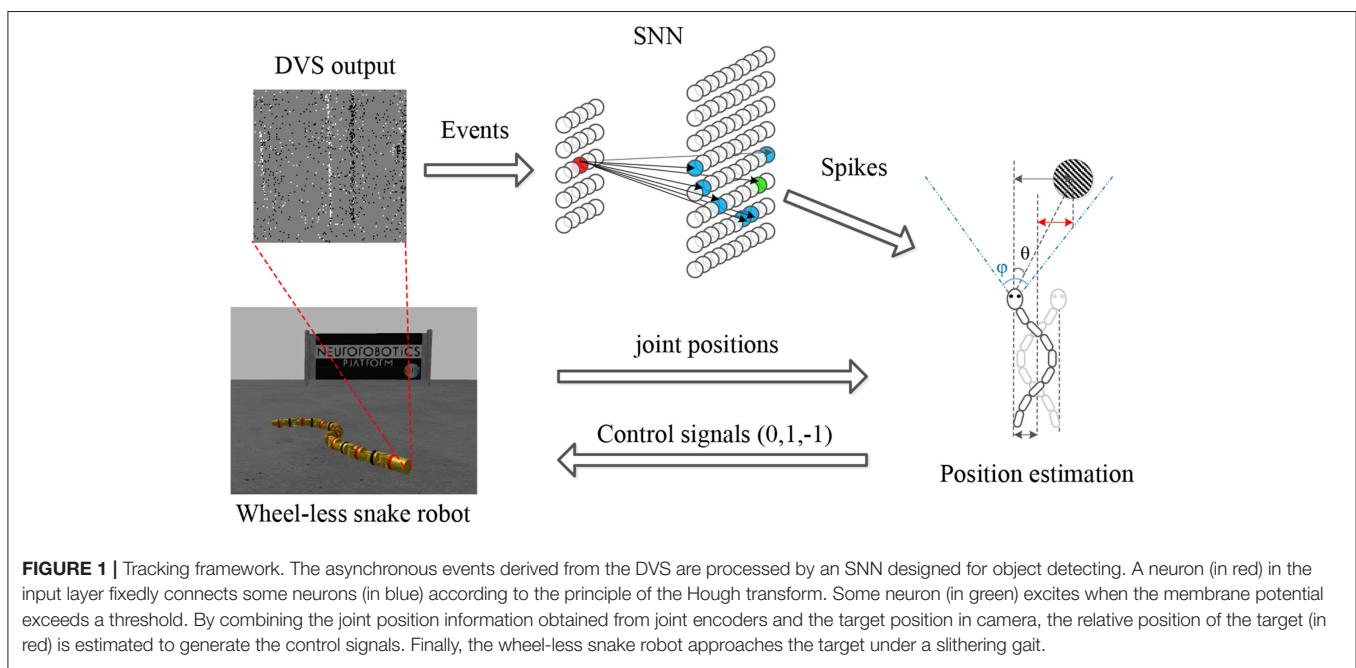
The behavior of a LIF neuron can be depicted as follows. (1) A spiking input causes an increase of the Membrane Potential (MP) of the neuron. (2) In the meantime, the MP always spontaneously decays at a fixed rate. (3) When the MP exceeds the threshold, a spike is generated as an output. Then, the MP of the fired neuron is reset to zero so that the neuron enters a refractory period, during which the MP remains zero and all input spikes are ignored. Because of the similarity between the dynamics of the LIF neuron and the voting process of shape detection based on the Hough transform, the SNN composed of LIF neurons is particularly well adapted for detecting line and circle.

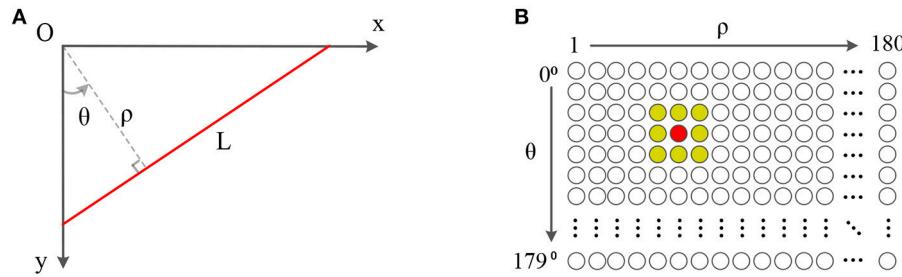
In this work, we designed two SNNs composed of the LIF neurons for line detection and circle detection, respectively. All the SNNs contain a two-layered topological structure, including an input layer and an output layer, as shown in **Figure 1**. The input neurons obtained the events from DVS and duplicated it immediately. The output neurons integrated the spikes and excited when they received enough spikes, which are extended LIF neurons with both a positive threshold and a negative threshold. Each input neuron permanently connected some output neurons according to the equation of specific shape. The membrane potential dynamics of extended LIF neurons was described as Algorithm 1.

### 2.2.1. Line Equation

According to the Hough transform, we assume  $\vec{n} = (\sin \theta, \cos \theta)$  as the normal vector perpendicular to the line  $L$  and  $\rho$  as the normal distance from the line to the origin. Hence, a point  $\vec{p} = (x, y)$  on the line  $L$  can be formulated as the equation:

$$\rho = \vec{n} \cdot \vec{p} = x \sin \theta + y \cos \theta, \quad (2)$$





**FIGURE 2 |** Panel **(A)** is a line in the Cartesian coordinate. Panel **(B)** is a  $180 \times 180$  SNN which  $\theta$  is limited in  $[0^\circ, 180^\circ]$  while  $\rho$  is limited in  $(0, 180]$ . The central neuron in red corresponds to the line  $L$  in **(A)**, which is connected to all neurons in its neighbor for local inhibition.

**Algorithm 1:** The membrane potential updating of an extended LIF neuron ( $\lambda$  is the fixed decay rate,  $v$  is the MP)

```

1: for input spike  $s_i$  with polarity  $p_i$  at  $t_i$  do
2:    $v_i \leftarrow \text{sign}(v_{i-1}) \cdot \max(|v_{i-1}| - \lambda \cdot (t_i - t_{i-1}), 0)$ 
3:    $v_i \leftarrow v_i + p_i$ 
4:   if  $|v_i| \geq v_{th}$  then
5:     Generate output spike  $\delta = \text{sign}(v_i)$  at  $t_i$ 
6:     Reset all connected neurons
7:      $v_i \leftarrow 0$ 
8:   end if
9: end for
```

which maps each point  $(x, y)$  from Cartesian coordinate into parameter space of  $(\theta, \rho)$  as a sinusoidal curves.

As shown in **Figure 2**, the SNN corresponding to the parameter space of  $(\theta, \rho)$  is built up, which consists of  $180 \times 180$  spiking neurons. The first dimension of the SNN represents the angle  $\theta$  and the second dimension is the distance  $\rho$ . In this example, the range of  $\theta$  is  $[0^\circ, 180^\circ]$  with  $1^\circ$  resolution and the range of  $\rho$  is  $(0, 180]$  pixels (180 approximately equals to the diagonal distance of the view field of a DVS128) with 1-pixel resolution. Each neuron of the SNN represents a line with  $(\theta, \rho)$  in the parameter space.

### 2.2.2. Circle Equation

As we all know, a point  $p = (x, y)$  on a circle C can be described by the following standard equation:

$$(x - a)^2 + (y - b)^2 = r^2, \quad (3)$$

where  $(a, b)$  is the coordinate of the center of the detected circle.

An SNN for circle detection is a three-dimensional parameter space of  $(a, b, r)$ , which consists of  $128 \times 128 \times 64$  spiking neurons. The first two dimensions of the SNN represent the position of a circle center in the horizontal direction and the vertical direction, respectively, while the third dimension is the radius of a circle. In this case, the resolutions in all of the three dimensions are 1 pixel. Each neuron of the SNN represents a circle with  $(a, b, r)$  in the parameter space.

### 2.2.3. Lateral Inhibition

The local lateral inhibition, which is a nature of biological neurons, was applied to suppress the noise in this work. Every

spiking neuron was connected to its adjacent neurons. Once a shape was detected, a spiking neuron would excite and all the spiking neurons connected would be inhibited. In order to make a trade-off between the result of noise suppress and the computation cost, we select a  $3 \times 3$  window as the range of local lateral inhibition. This means that neurons in the 8-adjacent of the fired neuron are reset. When a larger inhibition range is used, the target would be detected less often in approximately the same region, but more reset operations need to be done—otherwise, the reverse.

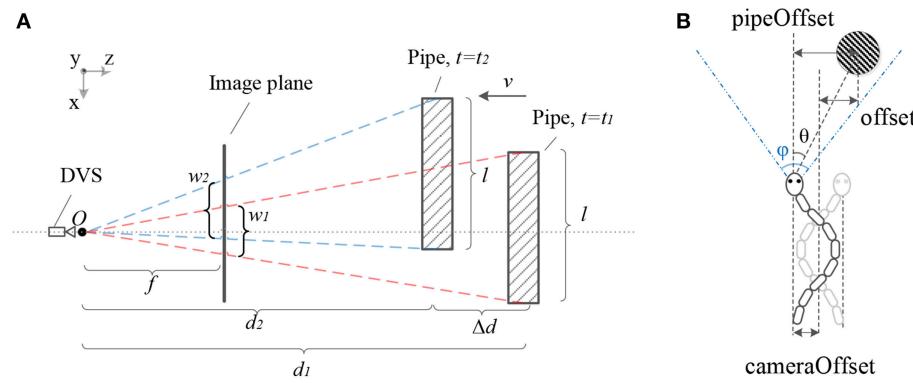
## 2.3. Pipe Detection

For pipe detection, the different poses of a vision sensor will result in different shapes of a pipe in the image plane, a pair of parallel lines or a circle. The circle can be detected directly by an SNN, while a strategy needs to be proposed for recognizing two parallel lines.

The edges on both sides of the pipe body can be detected as two parallel lines. In an indoor environment, the change of brightness is opposite on both sides of the pipe while the DVS camera moves perpendicular to the pipe. The polarity of the events on one edge of the pipe is positive while that on another edge is negative. Once the DVS camera moves to the opposite direction, the polarity of the events on two edges would reverse. The two lines with opposite polarity can be considered as the body of a pipe. Furthermore, this pair of parallel lines should appear at the maximum frequency which is equivalent to the highest fired rate of the spiking neuron. Therefore, three conditions to judge whether a pipe appears are listed below:

- The polarities of the two lines are opposite.
- The two lines are parallel or the difference of the angle is tiny.
- The pair of parallel lines appears at the maximum frequency in a short time period.

In this work, we tested all the detected lines in each time slot and only one pair of lines satisfying the above conditions would be found out. The target can be detected by Algorithm 2 and represented as a 4-tuple  $P(t, \theta, w, pos)$ , where  $t$  is the timestamp,  $\theta$  is the angle of the pipe, the  $w$  is the width of the pipe and the  $pos$  is the offset in pixel which is relative to the left side of the field of view.



**FIGURE 3 |** Panel **(A)** shows the geometric relationship between the DVS camera and the target pipe. When  $t = t_1$  the pipe is on the far position from camera. After the snake robot moves at an approximately constant velocity  $v$ , the pipe is on the near position when  $t = t_2$ . Panel **(B)** shows the geometric relationship among the pipe offset in the image plane, camera offset and actual offset while the snake-like robot moves forwards with a periodic swing.

### Algorithm 2: Event-based pipe detecting in the SNN

```

1: for each time slot do
2:   for event  $e_i = (t_i, x_i, y_i, p_i)$  do
3:     for every angle  $\theta_j$  in SNN do
4:       Calculate distance  $\rho_{\theta_j} = \text{argmin} |\rho - x_i \sin \theta_j - y_i \cos \theta_j|$ 
5:       Update neuron  $N(\theta_j, \rho_{\theta_j})$  at  $t_i$  with polarity  $p_i$ 
6:       if a neuron fires then
7:         Store the output spike
8:       end if
9:     end for
10:   end for
11:   Find out a pair of output spikes satisfying the judgment
    conditions
12:   if a pipe exits then
13:     Output the pipe  $P(t, \theta, w, pos)$ 
14:   end if
15: end for

```

## 2.4. Tracking

### 2.4.1. Motion Analysis of Snake Robot

Due to the fact that the DVS was mounted on the head of the snake robot, the horizontal offset and the orientation of the head should be known for obtaining precise tracking performance. Therefore, we collected the data from the joint encoder installed on the head and recorded the pose of the head, which includes the trajectory and the rotation in the simulation environment. Then, the horizontal offset and the orientation were analyzed by using FFT. After applying the head orientation compensation approach, the head of the snake robot kept always looking forward along the moving direction, which is a benefit for simplifying the model to estimate the relative position of the target. On the other hand, the horizontal offset was estimated based on the head joint position. This idea is derived from the observation that the rotation of the joints is the essential driving force of the wheel-less snake robot.

For tracking tasks, our snake robot moved under a slithering gait, which provides the most stable pose of camera. The horizontal offset of DVS was fitted by FFT. In the meantime,



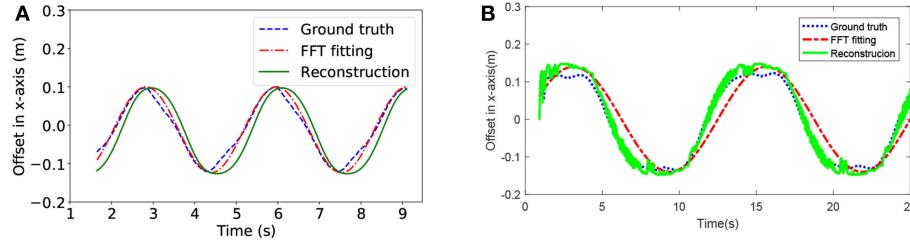
**FIGURE 4 |** The simulation environment in NRP, which includes a hollow pipe and a snake robot with a DVS.

we reconstructed the horizontal trajectory of the head module by integrating the head joint position  $\theta$ . In fact, the head joint position and the horizontal trajectory are both the periodic signal, which can be formulated as Equation (4) and Equation (6), respectively. Wu and Ma (2010) indicated that these two signals have the same form but different amplitude and phase; however, it was unable to give the offset of the head module in real time. Therefore, considering the joint rotation is the main driving force and reason to the motion of wheel-less snake robots, the head joint position was regarded as the argument of the horizontal trajectory. By tuning the phase and the amplitude, we got the offset of the DVS from the head joint position according to the Equation (6). Further, the situation of turning was approximately treated as that of moving straight.

$$\theta = a \cdot \sin(\omega \cdot t + \phi) \quad (4)$$

$$\begin{aligned} \theta_{\text{tuned}} &= \alpha \cdot a \cdot \sin[(\omega \cdot t + \phi) + \phi_{\text{pos}}] \\ &= \alpha \cdot (\theta \cdot \cos \phi_{\text{pos}} + \theta' \cdot \sin \phi_{\text{pos}}). \end{aligned} \quad (5)$$

$$\begin{aligned} \text{Position} &= A \cdot \sin(\theta_{\text{tuned}}) = A \cdot \sin[\alpha \cdot (\theta \cdot \cos \phi_{\text{pos}} \\ &+ \theta' \cdot \sin \phi_{\text{pos}})]. \end{aligned} \quad (6)$$



**FIGURE 5 |** The horizontal offset estimation for the head module of the snake robot while it is moving forwards. Panel **(A)** is from V-REP, Panel **(B)** is from NRP.

where  $A$  is the amplitude of the swing which is perpendicular to the direction of motion,  $\alpha$  is the ratio of the 90 degrees to the  $a$ ,  $\theta$  is the value of the joint encoder,  $\theta'$  is the joint velocity which is also the first-order derivative of  $\theta$  and  $\phi_{pos}$  is the phase difference. In our case, the tuple  $(A, \alpha, \phi)$  is  $(-0.124, 1.000, 0.000)$  for V-REP and  $(1.400, 2.571, 0.908)$  for NRP, respectively. These parameters are different in V-REP and NRP because the joint controllers and environment parameters in these two simulators are a little bit different.

#### 2.4.2. Relative Position Estimation

By using a distance sensor, such as an ultrasonic sensor, an IR sensor, etc., we can actively measure the relative distance between the snake robot and target. However, a time-of-flight distance sensor is usually directed, which means more sensors need to be installed in the limited space of the snake robot, especially when the position of the target is unknown. It is plausible to the actual snake robot as well. Besides that, when using a DVS along with distance sensors, we must ensure the consistency of different sensor measurements, that is, to make sure the measured data represent the same object. Therefore, aiming to use fewer sensors, we detected the target and estimated depth simultaneously by using a single DVS sensor.

Once the target is detected by the SNN, we can estimate the offset of the target in the horizontal plane and the forward relative distance between the DVS camera and the target. As shown in **Figure 3A**, in a time period  $\Delta t = t_2 - t_1$ , there is a functional relation between the decrease of the distance  $\Delta d = |d_2 - d_1|$  on the z-axis and the increase of the visible width of the target  $\Delta w = |w_2 - w_1|$ . Besides that, the distance  $d$  is always inversely proportional to the width of the target  $w$ , the scale factor is the focus length  $f$  multiplying the actual width  $l$ . Moreover, we can reasonably assume that the snake robot moves forward at an approximately constant speed since that is very small. Hence, the  $\Delta d$  can be estimated by multiplying the elapsed time and the speed of the snake robot. In summary, we can calculate the distance  $d_2$  depending on the displacement in a time period and the change of the target visible width according to the following equations.

$$\Delta d = |d_2 - d_1| = v \cdot \Delta t, \quad (7)$$

$$d = f \cdot l \cdot w^{-1}, \quad (8)$$

$$d_2 = \frac{f \cdot l}{w_2} = \frac{w_1 \cdot v \cdot \Delta t}{\Delta w}. \quad (9)$$

The snake-like robot moves slowly and the  $\Delta t$  and the  $\Delta w$  between two consecutive output spikes are tiny. Therefore, the error of distance calculated by the Equation (9) is remarkable. To reduce the error, two discrete output spikes are selected for distance estimating and the interval is 10 spikes in this paper.

Assuming that the position of the moving snake robot and the target pipe satisfies the relationship shown in **Figure 3B**, the snake robot swings along the central line while it is moving forwards. Hence, in order to obtain the offset between the central line and the target, we calculated two kinds of offset, the *pipeOffset* and the *cameraOffset*, respectively. Further, we proposed the method to estimate the pipe position for tracking as shown in Algorithm 3. We calculated the *pipeOffset* according to the ration of  $\tan \theta$  to  $\tan \frac{\phi}{2}$  ( $\phi$  is the FOV of DVS). Then we estimated *cameraOffset* according to the Equation (6).

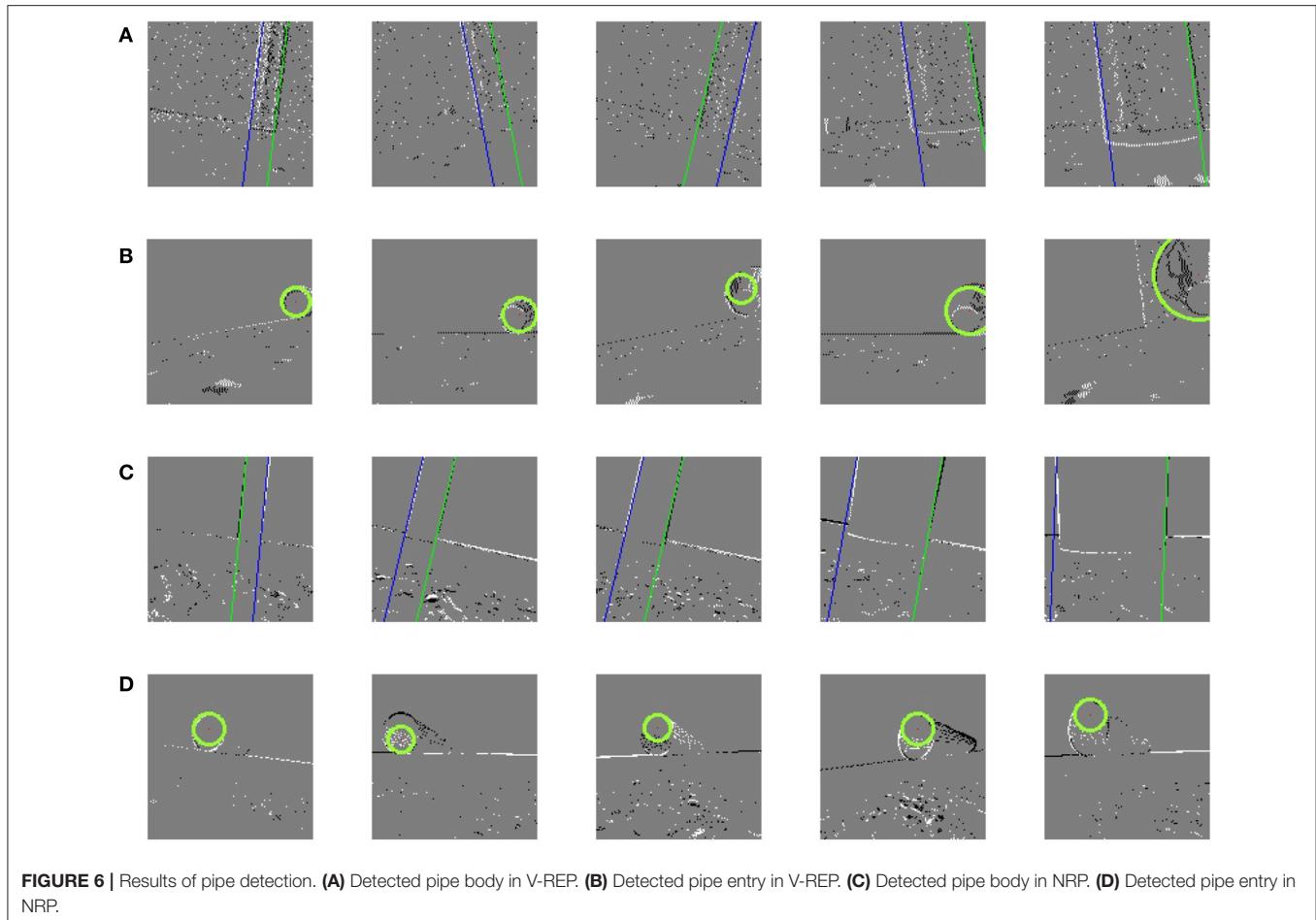
#### Algorithm 3: Position estimation and pipe tracking

```

1: for pipe  $P_i(t_i, \theta_i, w_i, pos_i)$  do
2:   if  $l = Size(pipe)$  is known then
3:     Calculate the distance  $d = f \cdot l \cdot w_i^{-1}$ 
4:   else
5:     if  $i \geq 10$  then
6:       Calculate the distance  $d = v \cdot (t_i - t_{i-10}) \cdot \frac{w_{i-10}}{w_i - w_{i-10}}$ 
7:     else
8:       Set the distance  $d$  to 0
9:     end if
10:   end if
11:   Calculate the offset of the pipe in DVS,  $pipeOffset = d \cdot \frac{|pos_i - 64| \cdot \tan \frac{\phi}{2}}{64}$ 
12:   Estimate the offset of the camera  $cameraOffset$ 
13:   Calculate the actual offset  $offset = pipeOffset + cameraOffset$ 
14:   Generate control parameter  $C = Sign(offset)$ , where  $-1$  means turning left,  $+1$  means turning right and  $0$  means going straight
15: end for
```

#### 2.5. GPU Acceleration

For artificial SNNs, the neurons will update their states only when input spikes arrive asynchronously. When SNNs are



**FIGURE 6 |** Results of pipe detection. **(A)** Detected pipe body in V-REP. **(B)** Detected pipe entry in V-REP. **(C)** Detected pipe body in NRP. **(D)** Detected pipe entry in NRP.

implemented on CPU or neuromorphic chips, the neurons can update asynchronously as well. However, the general CPU is unable to deal with the large quantity of communication between neurons and real-time state updating of neurons. The neuromorphic simulators and chips still have some drawbacks on running a large SNN. Therefore, we tried to accelerate our SNN by using a GPU. By providing a uniform clock, the neurons could update synchronously in a short time period, such as frame-based image processing.

### 3. EXPERIMENTS

Our tracking framework was evaluated both in NRP and V-REP, which is a robot-brain simulator and a robotics simulator, respectively. To begin with, we reconstructed the trajectory of the head module of the snake robot for obtaining the camera offset perpendicular to the forward direction. Then, two scenes were built. One of them only had a standing pipe while another one contained a lying pipe. Finally, experiments were conducted on the aforementioned scenes. Meanwhile, the SNN was performed on a CPU by only using a single thread and a CUDA GPU, respectively.

#### 3.1. Simulation Environments

The Neuromorphic Platform (NRP) (Roehrbein et al., 2016; Falotico et al., 2017) is an integrated simulation platform to facilitate a direct link between robotics and neuroscience. In its Gazebo-based world simulator, we built a modular wheel-less snake robot and a simple environment in which there was only one pipe, as shown in **Figure 4**. we built two SNNs as well, respectively for pipe body detection and pipe entry detection. By using the Robot Operating System (ROS) as a communicating middle-ware, our snake robot and the SNNs could exchange data and commands through ROS topics. While V-REP (Rohmer et al., 2013) is a simulator only for robotics in which we implemented SNNs outside and connected the snake robot and SNNs by utilizing Remote APIs. Although the NRP and the V-REP are both robotics simulators, models of the snake robot are a little bit different in the number of modules, the controller parameters and so on.

#### 3.2. Results and Discussions

We conducted experiments in V-REP and NRP, respectively, considering four situations, which are (1) tracking pipe body on the left side of the snake robot, (2) tracking pipe body on the right side of the snake robot, (3) tracking pipe entry on the left side of

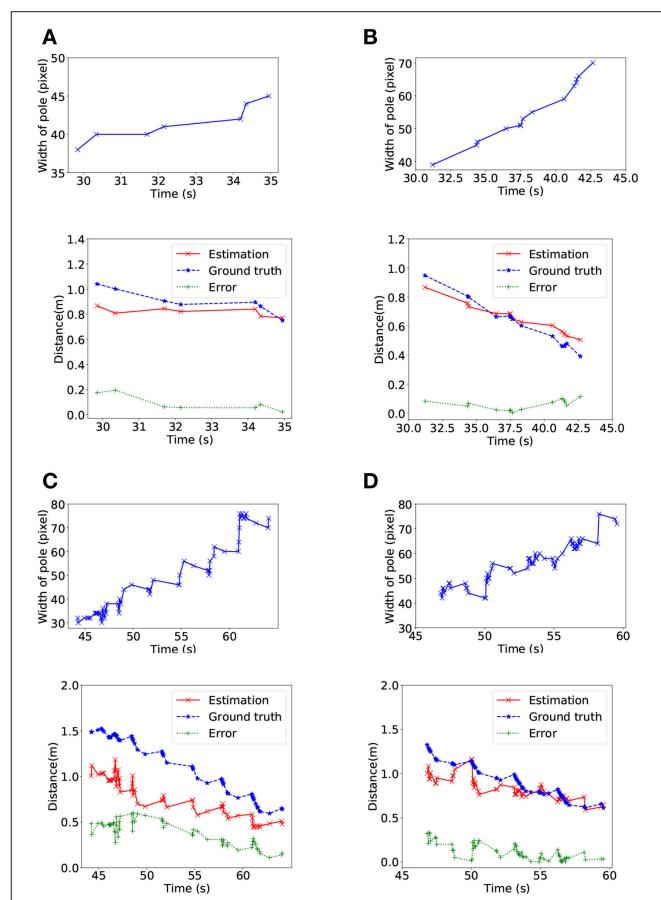
the snake robot and (4) tracking pipe entry on the right side of the snake robot.

First, we fitted the camera offset by using the head joint position. As shown in **Figure 5**, the blue curve is an approximately sine curve that represents the actual offset of the DVS camera, the red one is the offset analyzed by FFT and the green one is the reconstructed offset. We not only estimated the offset according to the Equation (6), but also applied a mean filter on the timeline to smooth the estimation result. Therefore, the horizontal offset of the DVS camera was directly obtained in real time under a low estimation error.

Then, for the pipe detection, we put a hollow pipe at different perspectives of the DVS camera. The snake robot always started moving from the same initial position for each situation, but the initial position in V-REP is different from that in NRP. During the detecting procedure, the event sequences generated by the DVS were fed into the vision SNN. Events were asynchronously processed, but we only detected pipe once in each time slot according to the method discussed in section 2.3, and the image and the position of the pipe were recorded and a part of them was shown in **Figure 6**. The standing pipe was represented by a blue and a green line and the lying pipe was represented by a green circle. The precision of the standing pipe detection is higher than the lying pipe detection because of the limitation of the network size, especially for circle detection. More neurons means greater ability to recognize the much finer structure of the target. Additionally, it suffered from worse precision when the snake robot got close to the target, especially the lying pipe. At the beginning, the circle looks dense and easily distinguished, however it looks noisier because more details of the target which generated noise spikes were seen. If we increased the firing threshold of the membrane potential to increase the precision of target detection, however, the firing rate of output neurons would reduce so significantly that there were not enough output spikes generated.

After that, the relative distance between the DVS camera and the target pipe was estimated by using Algorithm 3. As the snake robot moving toward the pipe, the width of the pipe increased in the image plane. As shown in **Figures 7A,B**, the relative distance decreased when the width of pipe increased and the average error was around 0.1 m. Nevertheless, the error of **Figures 7C,D** were much higher than for standing pipe tracking. The reason is that the firing rate was much higher and the difference of width was smaller than parallel lines detection when we detected the circle. Additionally, in all cases, larger error also occurred at the early time period of the simulating experiment. We only evaluated the precision on the data derived from V-REP because we assumed the size of targets was known in NRP to avoid introducing too much error.

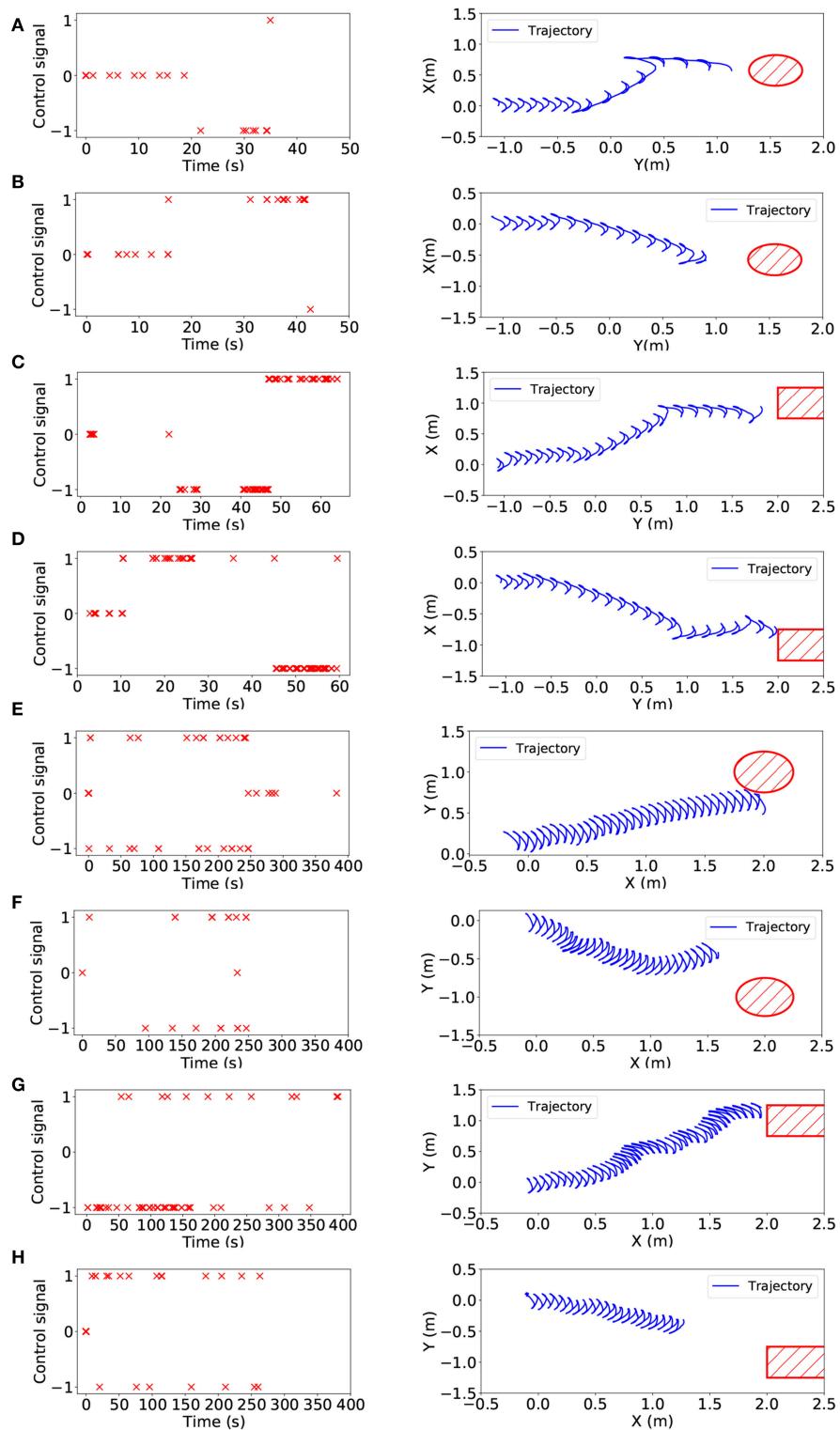
Furthermore, we estimated the actual offset of the target pipe and generated control signals so that our snake robot was able to achieve pipe tracking, as shown in **Figure 8**. **Figures 8A–D** showed the final trajectory of the snake robot for tracking in V-REP, the shapes in red were the actual position of the target pipe. Then, **Figures 8E–H** showed the final trajectory of the snake



**FIGURE 7 |** Relative position estimating in V-REP. For each case, the upper line shows the increase of the width of the target pipe in pixels and the lower line shows the distance between the head of the snake robot and the target pipe. In (A) and (C), the target pipe is on the left side of the snake robot, and the target pipe is on the right side in (B) and (D).

robot for tracking in NRP. All the results demonstrated that our tracking framework based on SNN was valid and effective. The snake robot was able to find the target pipe and approach it by performing a series of motion, including turning left, turning right and go straight. The trajectories shown in **Figure 8** were not representative of a smooth curve because of the swing of the snake robot, but the trend of motion is still correct. Another feature about the curves was that there were several obvious turning points in V-REP but not in NRP.

Finally, we compared the performance of the proposed SNNs on CPU and GPU, respectively. As shown in **Table 1**, the GPU was able to accelerate the detection procedure for each case, especially for lying pipe detection. Due to the three-dimensional parameter space for lying pipe detection, the GPU could achieve higher speedup than standing pipe detection. Moreover, in cases 3 and 4, the pipe detection could be performed in real time by using the GPU. In addition, the simulating experiments conducted in NRP showed higher frame rate than that of V-REP, especially when the GPU was utilized. The possible reason is that the V-REP spent more time in data transferring between the snake robot and the SNN.



**FIGURE 8 |** Left: Control signals of turning left(-1), turning right(1) and going straight(0) for the snake-like robot. Right: The overhead view of the trajectory of the head module while tracking. From (A–H), eight experiment results are shown which are left-side standing pipe tracking in V-REP, right-side standing pipe tracking in V-REP, left-side lying pipe tracking in V-REP, right-side lying pipe tracking in V-REP, left-side standing pipe tracking in NRP, right-side standing pipe tracking in NRP, left-side lying pipe tracking in NRP, right-side lying pipe tracking in NRP.

**TABLE 1 |** The frame rate of pipe tracking on CPU and on GPU, respectively.

Case	Hardware	CPU(fps)	GPU(fps)	Speedup
Standing pipe in V-REP	Intel i7-5500U/Nvidia Geforce 940 m	0.21	1.71	8.14
Lying pipe in V-REP	Intel i7-5500U/Nvidia Geforce 940 m	0.33	3.22	9.76
Standing pipe in NRP	Intel i7-4770/Nvidia Geforce GTX 645	1.63	40.74	24.99
Lying pipe in NRP	Intel i7-4770/Nvidia Geforce GTX 645	0.36	22.35	62.08

## 4. CONCLUSION

In this work, we proposed a pipe-like object detecting and tracking approach by combining the DVS and SNNs, and successfully performed on a wheel-less snake robot. The target pipe was detected by dealing with the asynchronous address-event stream obtained from a DVS. Then, an autonomous tracking method was present according to the relative position between the snake robot and the target. Furthermore, the performances of the proposed SNNs were estimated on CPU and GPU. The experiments demonstrated the efficacy of our tracking approach based on SNNs and showed the practicality and accuracy of the autonomous tracking method. Comparing the performances of our SNN model on CPUs and on GPUs, respectively, the SNN model showed the best performance on a GPU while displayed the highest precision on a CPU. However, there are still some drawbacks to our approach. The

prime one is that the performance of tracking is sensitive to the noise and the error in detection and position estimation.

## AUTHOR CONTRIBUTIONS

ZJ and AK brought up the core concept and architecture of this manuscript. ZJ and ZB designed the experiments. ZJ, ZB, and KH wrote the paper.

## FUNDING

This work is supported by a scholarship from the China Scholarship Council (CSC) under Grant No. 201606270201. Meanwhile, it is also supported by the German Research Foundation (DFG) and the Technical University of Munich (TUM) in the framework of the Open Access Publishing Program.

## REFERENCES

- Bing, Z., Cheng, L., Huang, K., Jiang, Z., Chen, G., Röhrbein, F., et al. (2017). “Towards autonomous locomotion: Slithering gait design of a snake-like robot for target observation and tracking,” in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (Vancouver, BC: IEEE), 2698–2703. doi: 10.1109/IROS.2017.8206095
- Burkitt, A. N. (2006). A review of the integrate-and-fire neuron model: I. homogeneous synaptic input. *Biol. Cybernet.* 95, 1–19. doi: 10.1007/s00422-006-0068-6
- Drazen, D., Lichtsteiner, P., Häfliger, P., Delbrück, T., and Jensen, A. (2011). Toward real-time particle tracking using an event-based dynamic vision sensor. *Exp. Fluids* 51:1465. doi: 10.1007/s00348-011-1207-y
- Falotico, E., Vannucci, L., Ambrosano, A., Albanese, U., Ulbrich, S., Vasquez Tieck, J. C., et al. (2017). Connecting artificial brains to robots in a comprehensive simulation framework: the neurorobotics platform. *Front. Neurorobot.* 11:2. doi: 10.3389/fnbot.2017.00002
- Gehrig, D., Rebecq, H., Gallego, G., and Scaramuzza, D. (2018). “Asynchronous, photometric feature tracking using events and frames,” in *Proceedings of the European Conference on Computer Vision (ECCV)* (Munich), 750–765. doi: 10.1007/978-3-030-01258-8-46
- Glover, A., and Bartolozzi, C. (2016). “Event-driven ball detection and gaze fixation in clutter,” in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (Daejeon: IEEE), 2203–2208. doi: 10.1109/IROS.2016.7759345
- Hu, D. L., and Shelley, M. (2012). “Slithering locomotion,” in *Natural Locomotion in Fluids and on Surfaces* (New York, NY: Springer), 117–135. doi: 10.1007/978-1-4614-3997-4-8
- Hu, Y., Zhao, W., and Wang, L. (2009). Vision-based target tracking and collision avoidance for two autonomous robotic fish. *IEEE Trans. Indust. Electr.* 56, 1401–1410. doi: 10.1109/TIE.2009.2014675
- Jiang, Z., Bing, Z., Huang, K., Chen, G., Cheng, L., and Knoll, A. (2017). “Event-based target tracking control for a snake robot using a dynamic vision sensor,” in *International Conference on Neural Information Processing* (Springer), 111–121. doi: 10.1007/978-3-319-70136-3-12
- Jin, H., Favaro, P., and Cipolla, R. (2005). “Visual tracking in the presence of motion blur,” in *CVPR 2005. IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2005* (San Diego, CA: IEEE), 2, 18–25.
- Kheradpisheh, S. R., Ganjtabesh, M., Thorpe, S. J., and Masquelier, T. (2018). Stdp-based spiking deep convolutional neural networks for object recognition. *Neural Netw.* 99, 56–67. doi: 10.1016/j.neunet.2017.12.005
- Kueng, B., Mueggler, E., Gallego, G., and Scaramuzza, D. (2016). “Low-latency visual odometry using event-based feature tracks,” in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (Daejeon: IEEE), 16–23. doi: 10.1109/IROS.2016.7758089
- Lagorce, X., Meyer, C., Ieng, S.-H., Filliat, D., and Benosman, R. (2015). Asynchronous event-based multikernel algorithm for high-speed visual features tracking. *IEEE Trans. Neural Netw. Learn. Syst.* 26, 1710–1720. doi: 10.1109/TNNLS.2014.2352401
- Lichtsteiner, P., Posch, C., and Delbrück, T. (2008). A 128×128 120 dB 15 μs latency asynchronous temporal contrast vision sensor. *IEEE J Solid State Circ.* 43, 566–576. doi: 10.1109/JSSC.2007.914337
- Liu, H., Moeyns, D. P., Das, G., Neil, D., Liu, S.-C., and Delbrück, T. (2016). “Combined frame-and event-based detection and tracking,” in *2016 IEEE International Symposium on Circuits and Systems (ISCAS)* (Montreal, QC: IEEE), 2511–2514. doi: 10.1109/ISCAS.2016.7539103
- Ma, B., Huang, L., Shen, J., Shao, L., Yang, M.-H., and Porikli, F. (2016). Visual tracking under motion blur. *IEEE Trans. Image Process.* 25, 5867–5876. doi: 10.1109/TIP.2016.2615812
- Moeyns, D. P., Corradi, F., Kerr, E., Vance, P., Das, G., Neil, D., et al. (2016). “Steering a predator robot using a mixed frame/event-driven convolutional neural network,” in *2016 Second International Conference on Event-based Control, Communication, and Signal Processing (EBCCSP)* (Krakow: IEEE), 1–8. doi: 10.1109/EBCCSP.2016.7605233
- Mueggler, E., Huber, B., and Scaramuzza, D. (2014). “Event-based, 6-dof pose tracking for high-speed maneuvers,” in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems* (Chicago, IL: IEEE), 2761–2768. doi: 10.1109/IROS.2014.6942940

- Ponte, H., Queenan, M., Gong, C., Mertz, C., Travers, M., Enner, F., et al. (2014). "Visual sensing for developing autonomous behavior in snake robots," in *2014 IEEE International Conference on Robotics and Automation (ICRA)* (Hong Kong: IEEE), 2779–2784. doi: 10.1109/ICRA.2014.6907257
- Rebecq, H., Horstschafer, T., and Scaramuzza, D. (2017). "Real-time visual-inertial odometry for event cameras using keyframe-based nonlinear optimization," in *British Machine Vision Conference (BMVC)* (London, UK). doi: 10.5244/C.31.16
- Roehrbein, F., Gewaltig, M.-O., Laschi, C., Klinker, G., Levi, P., and Knoll, A. (2016). "The neurorobotic platform: a simulation environment for brain-inspired robotics," in *Proceedings of ISR 2016: 47st International Symposium on Robotics* (Munich: VDE), 1–6.
- Rohmer, E., Singh, S. P., and Freese, M. (2013). "V-rep: a versatile and scalable robot simulation framework," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems* (Tokyo: IEEE), 1321–1326. doi: 10.1109/IROS.2013.6696520
- Schraml, S., Belbachir, A. N., Milosevic, N., and Schön, P. (2010). "Dynamic stereo vision system for real-time tracking," in *Proceedings of 2010 IEEE International Symposium on Circuits and Systems (ISCAS)* (Paris: IEEE), 1409–1412. doi: 10.1109/ISCAS.2010.5537289
- Seifozzakerini, S., Yau, W.-Y., Zhao, B., and Mao, K. (2016). "Event-based hough transform in a spiking neural network for multiple line detection and tracking using a dynamic vision sensor," in *BMVC* (York, UK). doi: 10.5244/C.30.94
- Wiesmann, G., Schraml, S., Litzenberger, M., Belbachir, A. N., Hofstatter, M., and Bartolozzi, C. (2012). "Event-driven embodied system for feature extraction and object recognition in robotic applications," in *2012 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)* (IEEE), 76–82. doi: 10.1109/CVPRW.2012.6238898
- Wright, C., Johnson, A., Peck, A., McCord, Z., Naaktgeboren, A., Gianforoni, P., et al. (2007). "Design of a modular snake robot," in *IROS 2007. IEEE/RSJ International Conference on Intelligent Robots and Systems, 2007* (San Diego, CA: IEEE), 2609–2614. doi: 10.1109/IROS.2007.4399617
- Wu, X., and Ma, S. (2010). "Head-navigated locomotion of a snake-like robot for its autonomous obstacle avoidance," in *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems* (Taipei, IEEE), 401–406.
- Wu, Y., Ling, H., Yu, J., Li, F., Mei, X., and Cheng, E. (2011). "Blurred target tracking by blur-driven tracker," in *2011 IEEE International Conference on Computer Vision (ICCV)* (Barcelona: IEEE), 1100–1107. doi: 10.1109/ICCV.2011.6126357

**Conflict of Interest Statement:** The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Copyright © 2019 Jiang, Bing, Huang and Knoll. This is an open-access article distributed under the terms of the Creative Commons Attribution License (CC BY). The use, distribution or reproduction in other forums is permitted, provided the original author(s) and the copyright owner(s) are credited and that the original publication in this journal is cited, in accordance with accepted academic practice. No use, distribution or reproduction is permitted which does not comply with these terms.