Event-Based Target Tracking Control for a Snake Robot Using a Dynamic Vision Sensor

Zhuangyi Jiang[1(✉)], Zhenshan Bing[1], Kai Huang[2], Guang Chen[1], Long Cheng[1], and Alois Knoll[1]

[1] Department of Informatics, Technical University of Munich, Boltzmannstr. 3, 85748 Munich, Germany
{jiangz,bing,guang,chengl,knoll}@in.tum.de
[2] School of Data and Computer Science, Sun Yat-Sen University, Guangzhou, China
huangk36@mail.sysu.edu.cn

Abstract. Dynamic Vision Sensor (DVS) is a promising neuromorphic vision sensor for autonomous locomotion control of mobile robots, as the DVS acquires visual information by mimicking retina to sense and encode the world as neural signals. In this paper, we present an autonomous target detecting and tracking control approach for a snake-like robot with a monocular DVS. By using Hough transform based on the Spiking Neural Network (SNN), the target pole is detected as two parallel lines from the event-based visual input. Then a depth estimation method based on the pose and motion of the robot is proposed. Furthermore, by combining the periodic motion feature of the snake-like robot, an adaptive tracking method based on the estimated depth information is introduced. Experiments are conducted on a snake-like robot to demonstrate the practicality and accuracy of our proposed method to track a target pole dynamically with a monocular DVS.

Keywords: Target tracking · Spiking Neural Network · Dynamic Vision Sensor · Neuromorphic snake robot · Hough transform

1 Introduction

Autonomous locomotion capability, namely making decisions on how, when, and where to move, is important for mobile robots, especially for snake-like robots designed for disaster rescue [1]. A typical implementation of the autonomous locomotion is the target tracking. Sensing, deciding and acting are the three components of the target tracking procedure [2]. Most research has focused on the acting component. However, the sensors for snake-like robots and the corresponding method of decision-making have not been well explored.

Despite various vision sensors have been used, such as frame-based camera and stereo camera, there still exist many limitations and deficiencies. Conventional computer vision acquires information by examining a series of pictures

D. Liu et al. (Eds.): ICONIP 2017, Part VI, LNCS 10639, pp. 111–121, 2017.
https://doi.org/10.1007/978-3-319-70136-3_12

at a fixed frame rate, regardless whether there is a change or not in the scene. It will lead to data redundancy, information loss, and dependency on lighting conditions. On the other hand, the cameras mounted on snake-like robots are not able to work in stable positions while the robots are moving, because of the periodic orientation of the body of the snake-like robot under 3D slithering gait.

Very few literature has explored the sensing and deciding approaches applied on snake-like robots. Ponte [2] proposed a pole tracking method by using a structured light sensor that can make 3D maps of the environment and IMU sensors to estimate the pose of the robot. Pfotzer [3] proposed an autonomous navigation method for the snake-like robots with wheels. However, these methods need to sense the environment in a static gait but not in real time, because of the blur caused by the moving of robots and the huge computation of processing image and making decisions.

As a promising solution, Dynamic Vision Sensor (DVS) [4] mimics the retina and generates spikes in response to the pixel-level changes in illumination caused by movement. Compared to a conventional frame-based camera, a DVS offers great advantages in terms of data rate, speed, and dynamic range [5], especially for mobile scenes. The DVS is suitable for working in critical scenarios with either bad light conditions or moving platforms. Moreover, events generated by a DVS can be directly fed into Spiking Neural Networks (SNNs) for fast target detecting and motion control.

Furthermore, it should be considered how to acquire depth information by DVS for target localizing and tracking. Everding [6] and Piatkowska [7] both extracted the depth information by streaming the events of stereo DVS. The stereo DVS camera is a good choice, however, it requires additional space than monocular DVS, which makes it difficult to be mounted on a snake-like robot with the small head module. Besides, there exists an extra blind area near the stereo DVS, so that the depth of objects in this area cannot be extracted.

In this paper, we present an autonomous pole detecting and tracking approach for a snake-like robot based on monocular DVS. First, we extract the line features of the target pole by streaming and processing event sequences from DVS with an SNN based on Hough Transform [5,8,9]. Two parallel lines are detected as pole boundaries when the membrane potential of the neurons in the SNN exceeds the pre-defined threshold. Then, a depth estimation method based on the pose and motion of the robot is proposed with the monocular DVS instead of the stereo DVS, which calculates the depth by the change of the width of the pole in pixels and the displacement of the snake-like robot in the forward direction. Furthermore, an adaptive tracking method based on the depth estimation is introduced. According to the pose of the robot and the offset of the target in the field of DVS, the relative depth and orientation between the target and the robot are estimated as the parameters for adaptive tracking control which uses a series of control signals of turning left or turning right. Finally, a set of target pole tracking experiments for the snake-like robots is conducted to demonstrate the accuracy and practicality of the proposed method.

The rest of the paper is organized as follows. Section 2 describes the relative background knowledge, including DVS camera and spiking neural network. Section 3 is the overview of the neural snake-like robot and the tracking system. Section 4 presents a spiking neural network designed for pole detection and pole detecting algorithm. The position estimation algorithm and tracking method are illustrated in Sect. 5. Section 6 shows the results of experiments conducted on a snake-like robot. Section 7 concludes this paper.

2 Background

The dynamic vision sensor (DVS) [10] is a silicon retina. Instead of wastefully sending entire images at fixed frame rates, only the local pixel-level changes caused by movement in a scene are transmitted. It records the change in illumination as events in real time. Once the intensity change exceeds a threshold, a positive or negative event will be generated to represent the change of dark-to-bright or bright-to-dark. An event is a 4-tuple *(t,x,y,p)*, where t is the timestamp of the event, x and y are the position of the event in pixels, and p is the polarity which is binary $(+/-)$. The DVS used in this paper has a 128×128 spatial resolution and $1\,\mu s$ temporal accuracy.

Spiking neural network (SNN) is the third generation of neural network models, increasing the level of realism in a neural simulation [11]. Each Spiking Neuron [12] has some spike inputs and a spike output. A spiking input causes an increase or decrease of the neuron's Membrane Potential (MP). At the meantime, the MP is always decaying by a fixed rate. Whenever the MP exceeds the positive or negative threshold, a spike with the corresponding polarity is generated in the output. Then the MP is reset to zero and the neuron enters a refractory period, during which MP remains zero and input spikes are ignored.

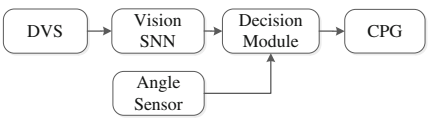


Fig. 1. The model of the neuromorphic snake robot for target tracking.

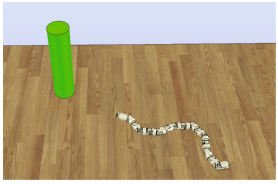


Fig. 2. The snake-like robot equipped with Dynamic Vision Sensor (DVS128)

3 Neuromorphic Snake Robot for Target Tracking

The method proposed in this paper for target detecting and tracking is tested on a neuromorphic snake-like robot we have developed. A neuromorphic snake-like robot model is designed in the robot simulator V-REP, consisting of 15 actuated

modules and a head module mounting the DVS128 sensor as shown in Fig. 2. All the modules are connected alternately with the lateral and dorsal planes. Each module with angle sensor is allowed 180° rotation. The monocular DVS in the head module sends back the real-time event stream to the host server via remote API functions of V-REP. Our snake-like robot is developed with 3D locomotion capabilities by the Central Pattern Generator (CPG) [13] which generates parameter of the motion equation for locomotion control, so that the autonomous locomotion control can be implemented simply by sending a series of commands of turning left or right. The slithering gait [14] for target tracking has been implemented to make the snake-like robot move forward fast and have a head module with relatively stable direction, so that the DVS camera mounting in the head module could obtain the valid data.

The framework of our neuromorphic snake robot for target tracking is divided into 5 components as shown in Fig. 1, including the DVS camera for sensing the environment, the angel sensors for pose estimation, the vision SNN for event-based object detection, the decision module for motion decision and the CPG-based control module.

4 SNN-Based Pole Detection by DVS

The pole is one of the most common obstacles indoor. It usually has a textureless and smooth surface so that it is regarded as two lines in each address-event frame. In this paper, an SNN corresponding to the Hough transform parameter space is designed. The events in the stream are fed into the SNN to detect the pole.

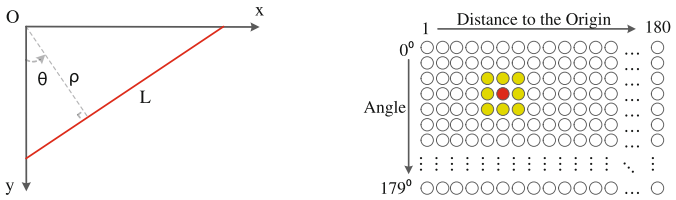
4.1 Vision SNN for Line Detection

According to the Hough transform, assuming $\boldsymbol{n} = (\sin\theta, \cos\theta)$ as the normal vector perpendicular to the line L and ρ as the normal distance from the line to the origin, for every point $\boldsymbol{p} = (x, y)$ on the line:

$$\rho = \boldsymbol{n} \cdot \boldsymbol{p} = x \sin\theta + y \cos\theta. \tag{1}$$

Equation (1) maps every point (x, y) from Cartesian coordinate into parameter space (θ, ρ) as a sinusoidal curves as shown in Fig. 3(a).

A 2D SNN corresponding to the parameter space of Hough transform is built up as shown in Fig. 3(b) for line detection, which consists of 180×180 spiking neurons. One dimension of the SNN is for angle θ and the other is for distance ρ, the range of θ is from 0° to 179°, the range of ρ is from 1 to $128\sqrt{2} \approx 180$ pixels. Each neuron of the SNN represents a line, or a point (θ, ρ) in the parameter space. The leaky integrate-and-fire (LIF) neuron model is used in the SNN. We use Algorithm 1 to update a spiking neuron. For every time slot, the MP of the spiking neuron decreases at a constant rate λ as the decay. When an input spike arrives, the absolute value of MP increases s_i. Then if the MP exceeds the positive threshold or the negative one, an output spike δ is generated and the fired spiking neuron will be reset.



(a) A line in the Cartesian coordinate (b) A 180×180 SNN

Fig. 3. θ is limited in $[0°, 180°)$ while ρ is limited in $[1, 180]$. The central neuron in (b) is corresponded to the line in (a), which is connected to all neurons in neighbor for local inhibition.

Algorithm 1. Updating of a spiking neuron ($\lambda = 0.3$/ms, $v_{th} = 40$)

for input spike s_i at t_i **do**
  $v_i \leftarrow sign(v_{i-1}) \cdot \max(|v_{i-1}| - \lambda(t_i - t_{i-1}), 0)$
  $v_i \leftarrow v_i + s_i$
  if $|v_i| \geq v_{th}$ **then**
    Generate output spike $\delta = sign(v_i)$ at t_i
    Reset all connected neurons
    $v_i \leftarrow 0$

Meanwhile, a local inhibition strategy is applied to suppress the noise. Every spiking neuron is connected to those in its neighborhood. Once a line is detected, a spiking neuron will fire and all the spiking neurons connected will be reset as 0. In this paper, the size of the neighborhood for local inhibition strategy is 7×7, which means the angle range is $\pm 3°$ and the distance range is ± 3.

4.2 SNN-Based Pole Detection

The vertical edge on both sides of the pole can be detected as two parallel lines. In the certain indoor environment, while the DVS camera is moving in the direction perpendicular to the pole, the change of lightness is opposite on both sides of the pole. The polarity of the events on the one side is positive, but negative on the other side. Once the DVS camera moves to the opposite direction, the polarity of the events on the two side would reverse. The two lines with opposite polarity can be considered as a pole. Furthermore, there should not be any other lines between the two sides of the pole which is textureless. Therefore, three conditions for line detection are found out as below:

- The polarities of two lines are different.
- The two lines are parallel or the difference of the angle is tiny.
- The distance between the two lines is the minimum in all pairs of line.

In this paper, all the lines are tested, which are detected by the SNN in each time slot. The pair of lines met the above conditions are found out. The target

pole can be detected by Algorithm 2 and represented as a 4-tuple $P(t, \theta, w, l)$, where t is the timestamp, θ is the angle of the pole, the w is the width of the pole and the l is the offset to the left side of the view field. Moreover, the procedure in Algorithm 2 can be vectorized and accelerated by parallel methods, such as multi-thread, GPU and neuromorphic chip.

5 Depth Estimation and Adaptive Pole Tracking

When the pole is detected by the SNN, we can estimate the offset of the pole on the x-axis and the distance between the DVS camera and the pole on the z-axis. As shown in Fig. 4(a), In a period of time $\Delta t = t_2 - t_1$ the decrease of the distance $\Delta d = |d_2 - d_1|$ on the z-axis is relative to the increase of the width of the pole $\Delta w = |w_2 - w_1|$ in DVS. Assuming the robot moves at a constant speed, the Δd can be estimated by multiplying the time elapsed and the speed. Due to our snake-like robot moving slowly, the speed forwards is approximately constant. On the other hand, the distance d in meters is inversely proportional to the width of the pole w in pixels, the scale factor is the focus length f in pixels multiplying the width l in meters. According to Eqs. (2), (3) and (4), we

Algorithm 2. Event-based pole detecting in the SNN

for event $e_i = (t_i, x_i, y_i, s_i)$ in each time slot **do**
  for every angle θ_j in SNN **do**
    Calculate distance $\rho_{\theta_j} = argmin\ |\rho - x_i \sin\theta_j - y_i \cos\theta_j|$
    Excite neuron $N(\theta_j, \rho_{\theta_j})$ at t_i with s_i (algorithm 1)
    Find out two output spikes met the pole conditions
    if a pole exits **then**
      Output the pole $P(t, \theta, w, l)$

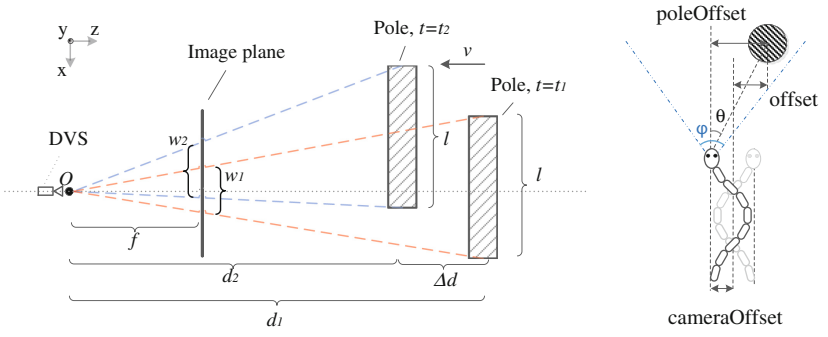


Fig. 4. (a) The geometric relationship between the DVS camera and the pole. When t = t1 the pole is on the right position. After the DVS camera moving at a certain velocity v, the pole is on the left position when $t = t_2$. (b) The periodic motion of the DVS camera while the snake-like robot moving forwards.

Algorithm 3. Position estimation and adaptive tracking for target (v = 0.4 m/s)

for pole $p_i(t_i, \theta_i, w_i, l_i)$ **do**
  if $i \geq 10$ **then**
    Calculate the distance, $d = v \cdot (t_i - t_{i-10}) \cdot \frac{w_{i-10}}{w_i - w_{i-10}}$
    Calculate the offset of the pole in DVS, $poleOffset = d \cdot \frac{|l_i - 64| \cdot \tan\frac{\phi}{2}}{64}$
    Calculate the offset of the camera $cameraOffset$
    $offset = poleOffset + robotOffset$
    Trun left when $offset < 0$
    Trun right when $offset > 0$
    Go straight when $offset == 0$

can calculate the distance d_2 depending on the displacement in a time period and the change of the pole's width in pixels.

$$\Delta d = |d_2 - d_1| = v \cdot \Delta t, \tag{2}$$

$$d = f \cdot l \cdot w^{-1}, \tag{3}$$

$$d_2 = \frac{f \cdot l}{w_2} = \frac{w_1 \cdot v \cdot \Delta t}{\Delta w}. \tag{4}$$

The snake-like robot moves slowly and the Δt and the Δw between two consecutive output spikes are tiny. Therefore, the error of distance calculated by the Eq. (4) is remarkable. To reduce the error, two discrete output spikes are selected for distance estimating and the interval is 10 spikes in this paper.

Actually, the pole cannot be detected in each time slot by our SNN, so that it is hard to track the target pole at a fixed frequency. Therefore, an adaptive method is proposed that the robot makes a decision to turn left or turn right immediately when the relative position of the target pole is estimated. The relative position, that is the offset to the symmetry axis while moving, consists of the offset of the pole in DVS and the offset of the camera mounted on the robot's head. Considering the mobile snake-like robot, as shown in Fig. 4(b), the offset of the pole is calculated by the ratio of $\tan\theta$ to $\tan\frac{\phi}{2}$ ($\phi = 65°$) and the ratio of l to half of the resolution of DVS. Then the offset of the camera is calculated just according to the periodic motion of the head module in the horizontal direction. The trajectory of the camera can be extracted so that the relationship between the angle of the head module and the amplitude can be calculated by FFT. Especially, the trajectory can be extracted directly in the simulator. Finally, the command of turning left, turning right or going straight is sent to the robot control module implemented by the Central Pattern Generator (CPG). Algorithm 3 is performed for each pole p_i. The distance is calculated by Eq. (4) and $Offset$ is calculated by $poleOffset$ and $cameraOffset$. Finally, the movement direction is decided according to $offset$.

6 Experiments

The proposed method was evaluated in the simulator V-REP, an indoor scene was built up with a pole in front of a snake-like robot mounted a DVS in the head module. Two cases were tested, the pole was on the left side as the case 1 and the pole was on the right side as the case 2. In case 1, the initial position of the left pole is (−0.574, 1.550, 1.200), the initial position of the DVS is (−0.021, −0.920, 0.046). In case 2, the initial position of the right pole is (0.574, 1.550, 1.200), the initial position of the DVS is (−0.021, −0.920, 0.046). The robot moves forwards at the speed of 0.04 m/s, while the head module is rotating and swinging periodically. The event sequences that are obtained from the DVS in V-REP inspired the vision SNN which is implemented in Python. The trajectory of the head module of the snake-like robot, the angle of the servo and the relative distance are recorded while simulating.

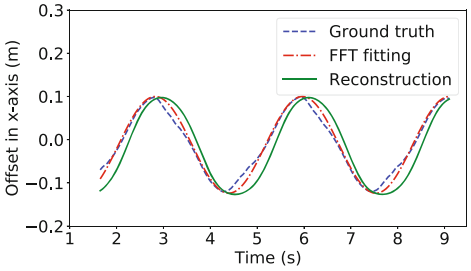


Fig. 5. The motion estimation of the head of snake-like robot while moving forwards.

The pole was detected in each time slot, the image and the position of the pole in case 2 were recorded and some of them were shown in Fig. 6. By using Algorithm 2, the pole was exactly detected at different distances. With the snake-like robot moving towards the pole, the width of the pole increased, as shown in Fig. 7. The offset of the pole to the left side of the image was obtained by calculating the center of two lines.

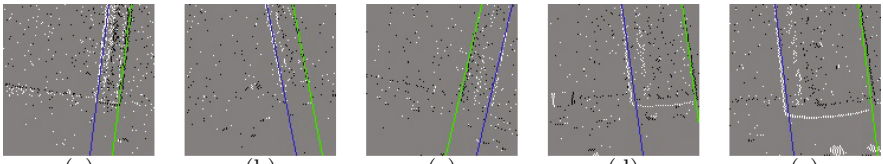
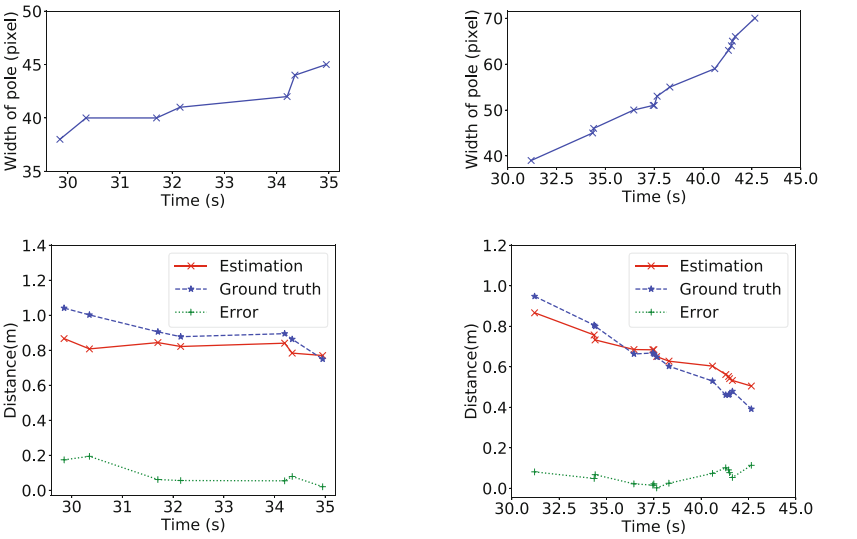


Fig. 6. The poles were detected in case 2. (a) A pole detected at 0.3 s. (b) A pole detected at 7.65 s. (c) A pole detected at 15.6 s when the robot turned right. (d) A pole detected at 36.45 s. (e) A pole detected at 42.65 s when the robot turned left.

Then the trajectory of the head module of the snake-like robot was analyzed by FFT while the robot moving straight. As shown in Fig. 5, the trajectory fitted by FFT was reconstructed, the peak amplitude was 0.112 m at the frequency 0.319. In the meantime, we reconstructed the trajectory of the head module by the angle θ extracted by V-REP. we obtained the offset of the DVS by the formula shown in Eq. (5). Further, the situation of turning was approximately treated as that moving straight.

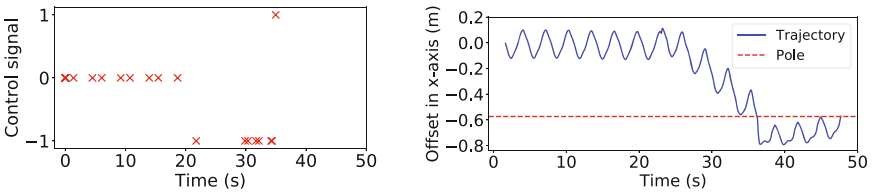
$$y = -0.124 \cdot sin\theta - 0.010. \tag{5}$$



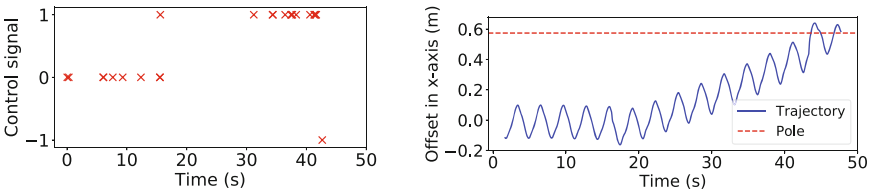
(a) Case 1: the pole is on the left side (b) Case 2: the pole is on the right side

Fig. 7. The upper graphs show the increase of the width of the target pole in pixels. The lower graphs show the distance between the snake-like robot and the target pole.

For the pole offset to the snake-like robot, the offset of the pole in the address-event image is known after the pole is detected and the real offset of the pole is proportional to offset in DVS. So the ratio of offset in DVS to the real offset is decided by the ratio of the drift angle to the field angle as shown in Fig. 4(b) and Algorithm 3. The distance between the robot to the target pole was estimated first, as shown in Fig. 7, the depth was the estimated value from the tenth time when the pole was detected. Compared to the real distance extracted in V-REP, the error value is relatively small. Then the offset was calculated in every time when the pole was detected, and the control signals to the CPG of the snake-like robot were generated. As shown in Fig. 8, in the beginning the signal was 0 which means going straight. Then The robot turned left at 21.75 s and turned right at 34.95 s in case 1. The robot turned right at 15.6 s and turned left at 42.65 s in



(a) Case 1: the pole is on the left side



(b) Case 2: the pole is on the right side

Fig. 8. Left: Control signals of turning left (−1), turning right (1) and going straight (0) for the snake-like robot. Right: The trajectory of the head module while tracking.

case 2. Finally The snake-like robot successfully arrived at the position of the target pole in the both two cases. The chart illuminates the practicality and accuracy of the method we proposed that the SNN-based pole detecting and tracking method.

7 Conclusion

In this paper, we proposed a pole detecting and tracking approach based on the combination of DVS and SNN for the snake-like robot. The combination is novel to be used in autonomous locomotion. A simulating scene is built up to test our approach on the neuromorphic snake-like robot. The target pole is detected in the address-event stream obtained from DVS. An adaptive tracking method is proposed, according to the change of relative position between the robot and the target pole. Experiments demonstrate the efficacy of the SNN for pole detecting and the practicality and accuracy of the adaptive tracking method.

References

1. Liljebäck, P., Pettersen, K.Y., Stavdahl, O., Gravdahl, J.T.: Snake Robots: Modelling, Mechatronics, and Control. Springer Science & Business Media, Heidelberg (2012)
2. Ponte, H., Queenan, M., Gong, C., Mertz, C., Travers, M., Enner, F.: Visual sensing for developing autonomous behavior in snake robots. In: 2014 IEEE International Conference on Robotics and Automation (ICRA), pp. 2779–2784. IEEE (2014)
3. Pfotzer, L., Klemm, S., Rönnau, A., Zöllner, J.M., Dillmann, R.: Autonomous navigation for reconfigurable snake-like robots in challenging, unknown environments. Robot. Auton. Syst. **89**, 123–135 (2017)

4. Weikersdorfer, D., Adrian, D.B., Cremers, D., Conradt, J.: Event-based 3D SLAM with a depth-augmented dynamic vision sensor. In: 2014 IEEE International Conference on Robotics and Automation (ICRA), pp. 359–364. IEEE (2014)
5. Seifozzakerini, S., Yau, W.Y., Zhao, B., Mao, K.: Event-based hough transform in a spiking neural network for multiple line detection and tracking using a dynamic vision sensor. In: BMVC (2016)
6. Everding, L., Walger, L., Ghaderi, V.S., Conradt, J.: A mobility device for the blind with improved vertical resolution using dynamic vision sensors. In: 2016 IEEE 18th International Conference on e-Health Networking, Applications and Services (Healthcom), pp. 1–5. IEEE (2016)
7. Piatkowska, E., Belbachir, A., Gelautz, M.: Asynchronous stereo vision for event-driven dynamic stereo sensor using an adaptive cooperative approach. In: Proceedings of the IEEE International Conference on Computer Vision Workshops, pp. 45–50 (2013)
8. Wiesmann, G., Schraml, S., Litzenberger, M., Belbachir, A.N., Hofstatter, M., Bartolozzi, C.: Event-driven embodied system for feature extraction and object recognition in robotic applications. In: Computer Vision and Pattern Recognition Workshops (CVPRW), pp. 76–82. IEEE (2012)
9. Auerbach, J.: Cross tracking on the DVS using an extended hough space method. In: Telluride Nueromorphic Cognition Engineering Workshop (2009)
10. Conradt, J., Galluppi, F., Stewart, T.C.: Trainable sensorimotor mapping in a neuromorphic robot. Robot. Auton. Syst. **71**, 60–68 (2015)
11. Maass, W.: Networks of spiking neurons: the third generation of neural network models. Neural Netw. **10**(9), 1659–1671 (1997)
12. Burkitt, A.N.: A review of the integrate-and-fire neuron model: I. Homogeneous synaptic input. Biol. Cybern. **95**(1), 1–19 (2006)
13. Bing, Z., Cheng, L., Huang, K., Zhou, M., Knoll, A.: CPG-based control of smooth transition for body shape and locomotion speed of a snake-like robot. In: 2017 IEEE International Conference on Robotics and Automation (ICRA), pp. 4146–4153. IEEE (2017)
14. Bing, Z., Cheng, L., Chen, G., Röhrbein, F., Huang, K., Knoll, A.: Towards autonomous locomotion: CPG-based control of smooth 3D slithering gait transition of a snake-like robot. Bioinspir. Biomim. **12**(3), 035001 (2017)