Biological Cybernetics
https://doi.org/10.1007/s00422-023-00972-x

Biological Cybernetics

ORIGINAL ARTICLE

Divisive normalization processors in the early visual system of the *Drosophila* brain

Aurel A. Lazar[1] · **Yiyin Zhou**[1,2]

Received: 31 December 2022 / Accepted: 6 August 2023

Abstract
Divisive normalization is a model of canonical computation of brain circuits. We demonstrate that two cascaded divisive normalization processors (DNPs), carrying out intensity/contrast gain control and elementary motion detection, respectively, can model the robust motion detection realized by the early visual system of the fruit fly. We first introduce a model of elementary motion detection and rewrite its underlying phase-based motion detection algorithm as a feedforward divisive normalization processor. We then cascade the DNP modeling the photoreceptor/amacrine cell layer with the motion detection DNP. We extensively evaluate the DNP for motion detection in dynamic environments where light intensity varies by orders of magnitude. The results are compared to other bio-inspired motion detectors as well as state-of-the-art optic flow algorithms under natural conditions. Our results demonstrate the potential of DNPs as canonical building blocks modeling the analog processing of early visual systems. The model highlights analog processing for accurately detecting visual motion, in both vertebrates and invertebrates. The results presented here shed new light on employing DNP-based algorithms in computer vision.

Keywords Motion detection · Divisive normalization · *Drosophila* · Phase processing · Gain control

1 Introduction

Robust motion detection is the key first processing step for insects to safely navigate complex environments. Current state-of-the-art computer vision algorithms achieve good performance under demanding navigation conditions. However, under extreme conditions, their performance often quickly degrades (Mathis et al. 2016; Li et al. 2018). Surprisingly, however, the early vision system of the fruit fly is remarkably accurate at detecting motion in complex environments under a vast range of light intensity conditions. The logic of computation in the fly visual system is substantially different from the traditional methods of computation employed by current man-made counterparts. This enables the fly to navigate through terrains with rapid light intensity changes, even though it only possesses a single photoreceptor type (van Hateren 1997).

Modern optic flow algorithms may take seconds or even minutes to compare two consecutive frames (Baker et al. 2011; Menze et al. 2018). While the processing speed was recently improved upon by using deep neural network-based algorithms, the cost of training time and the required large amounts of training data remain excessive. For the low level tasks such as elementary motion detection, fly vision is far more efficient, faster and more robust without loss of precision during events that are critical for survival, such as rapid predator attacks taking place on short time scales (hundreds of milliseconds). Strikingly, in fruit flies, like in many other insects and mammals, processing delays are minimal. It only takes 3 synapses from photoreceptors to reach the neurons responsible for detecting low-level directional motion with minimal energy expenditure (Sy et al. 2013)

Communicated by Benjamin Lindner.

The authors' names are listed in alphabetical order.

This article is published as part of the Special Issue on "What can Computer Vision learn from Visual Neuroscience?.

✉ Aurel A. Lazar
aurel@ee.columbia.edu

Yiyin Zhou
yiyin@ee.columbia.edu; yiyin.zhou@fordham.edu

[1] Department of Electrical Engineering, Columbia University, New York, NY 10027, USA

[2] Present Address: Department of Computer and Information Science, Fordham University, New York, NY 10023, USA

Published online: 13 September 2023

yyz :(see Fig. 1A); an efficient computational principle of motion detection seems to be at work.

This calls for developing biologically informed robust motion detection algorithms. Two half-century-old computational theories of motion detection, namely the Reichardt motion detector (Hassenstein and Reichardt 1956) and Barlow–Levick motion detector (Barlow and Levick 1965), have dominated the field. Recent studies have unveiled the basic anatomical structure of the fly's motion detection pathways (Yang and Clandinin 2018; Borst et al. 2020). While these and other studies did rapidly advance our understanding of motion detection in the early vision system of the fruit fly, the underlying models have yet to be successful in capturing the surprising robustness of fly vision. In Lazar et al. (2016), we compared the two prevailing models of fly motion detection with a more complex phase-based algorithm that we devised. Under different luminance and contrast conditions, we demonstrated that (i) none of the three algorithms could fully account for motion in natural scenes, and (ii) the detection of motion was not robust at low luminance/contrast levels. This suggests fundamental limits in current modeling approaches to visual motion detection that are narrowly focused on simple feedforward motion detection mechanisms. The latter do not match the vastly superior performance of the motion detection circuits in flies.

There is strong evidence showing that divisive normalization may contribute to gain control in olfaction (Olsen et al. 2010), vertebrate retina (Beaudoin et al. 2007), primary visual cortex (Carandini and Heeger 1994), primary auditory cortex (Rabinowitz et al. 2011) and sensory integration (Ohshiro et al. 2017). Feedforward divisive normalization has been proposed as a model of canonical neural computation (Carandini and Heeger 2012), and used in nonlinear image representation for achieving perceptual advantages (Lyu and Simoncelli 2008). This computation is key to many sensory processing circuits underlying adaptation and attention. Feedforward normalization is also frequently used in deep neural networks (Goodfellow et al. 2016; Ioffe and Szegedy 2015).

In Lazar et al. (2020), we presented a class of divisive normalization processors (DNPs) that operate in the time and the space-time domain. A DNP example is shown in the left block of Fig. 1B. Each DNP channel exhibits Volterra processors (VP) in a feedforward and local feedback divisive normalization branch. In addition, a multi-input Volterra processor (MVP) provides global feedback. With input from all channel outputs, the MVP provides feedback into each channel. This type of MIMO circuit architecture has been observed in many neural systems (Lazar et al. 2022b).

Stimuli processed by DNPs can be faithfully recovered from the output (Lazar et al. 2022a), suggesting that no information is lost during processing. We posit that a feedforward/feedback divisive normalization processor is embedded in every layer of the motion detection pathway (see Fig. 1A), including the lamina, medulla and a single layer in the lobula.

In this paper, we demonstrate that two key processing steps in the motion detection pathway, including the elementary motion detector and the intensity and contrast gain control mechanism, can be effectively modeled with DNPs. Two cascaded DNPs depicted in Fig. 1B implementing intensity and contrast gain control and elementary motion detection, respectively, can model the robust motion detection realized by the early visual system of the fruit fly brain. This suggests that, despite its nonlinearity, the class of DNPs can be used as computational building blocks in early sensory processing (Lazar et al. 2023).

This paper is organized as follows. In Sect. 2, we model the local phase-based motion detector devised in Lazar et al. (2016) using DNPs, and present an improved method for extracting motion velocity and derive an accuracy condition. In Sect. 3, we characterize the I/O of the DNP modeling the intensity and contrast gain control of the photoreceptor/amacrine cell layer, and evaluate its performance. In Sect. 4, we extensively evaluate the DNP for motion detection and the end-to-end motion detection performance with intensity and contrast gain control in natural, dynamic environments.

2 Modeling motion detection with divisive normalization processors

In this section, we model the local phase-based motion detector (Lazar et al. 2016) as a DNP. This formulation provides a new insight into the structure of the local phase-based motion detector, and how phase computation can be carried out in neural circuits. With the DNP model of motion detection, we can directly relate the phase-based motion detector to existing bio-inspired motion detection models such as the Reichardt motion detector (Hassenstein and Reichardt 1956) and the motion energy model (Adelson and Bergen 1985). We then provide a new way for extracting both direction and velocity information as well as a new estimate for motion, and discuss the condition for the estimate to be accurate. Finally, we illustrate the motion detector's sensitivity to brightness and contrast levels and demonstrate its limitations.

2.1 Divisive normalization underlying the local phase-based motion detector

Let (x_0, y_0) be an arbitrary point of reference of the visual field. The Fourier transform of the stimulus $u(x, y, t)$ projected onto the Gaussian window function centered at (x_0, y_0) and denoted by $w_{x_0,y_0}(x, y) = e^{-((x-x_0)^2+(y-y_0)^2)/2\sigma^2}$ is given by

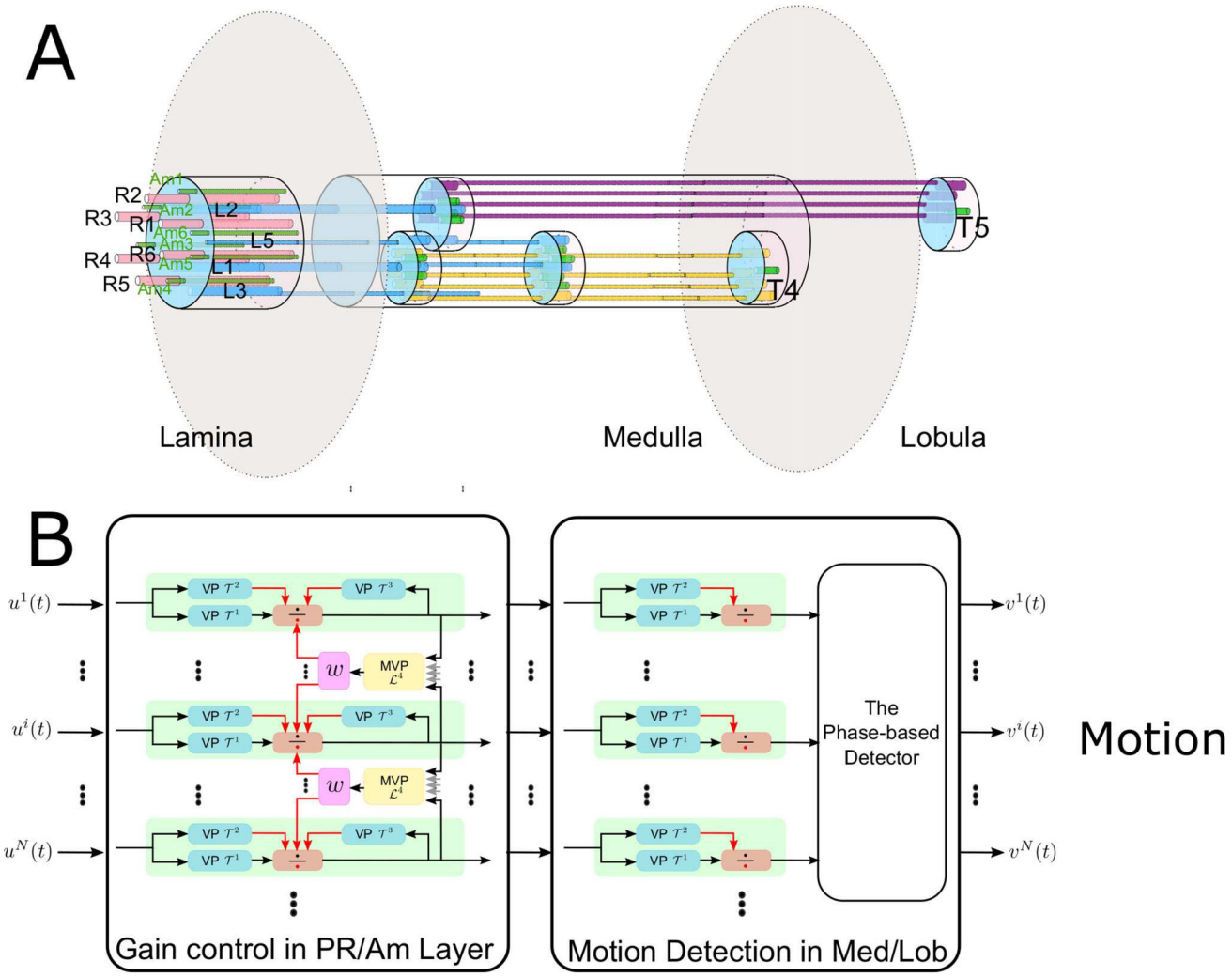

Fig. 1 End-to-end motion detection pathways in the early vision system of the fruit fly and a cascade of two DNPs modeling the motion detection circuit. **A** Canonical neural circuits with components embedded into the motion detection pathway. Canonical circuits in each neuropil are indicated by wide, long cylinders. Flat cylinders represent the intersection of canonical circuits with processing layers such as strata in the Medulla and layers in the Lobula. Thin, long cylinders represent the neurites of (pink) photoreceptors, (blue) Lamina output neurons, (yellow) ON-pathway neurons, (violet) OFF-pathway neurons, (green) neurites of wide-field neurons that innervate multiple canonical circuits in the same stratum/layer. **B** A cascade of two DNP blocks modeling the motion detection pathways in the fly eye. The first DNP block (left) models gain control of visual stimuli in the photoreceptor/amacrine cell layer, corresponding to the left cylinder in (**A**). The second block (right) models the motion detection in the Medulla/Lobula, corresponding to the right cylinder in (**A**)

$$\begin{aligned} U_{x_0,y_0}(\omega_x,\omega_y,t) &= \int_{\mathbb{R}^2} u(x,y,t) w_{x_0,y_0}(x,y) \\ &\quad e^{-j(\omega_x(x-x_0)+\omega_y(y-y_0))}\mathrm{d}x\mathrm{d}y \\ &= a_{x_0,y_0}(\omega_x,\omega_y,t) + j b_{x_0,y_0}(\omega_x,\omega_y,t) \\ &= A_{x_0,y_0}(\omega_x,\omega_y,t) e^{j\phi_{x_0,y_0}(\omega_x,\omega_y,t)}, \end{aligned} \tag{1}$$

where $a_{x_0,y_0}(\omega_x,\omega_y,t)$ and $b_{x_0,y_0}(\omega_x,\omega_y,t)$, respectively, are outputs of a quadrature pair of Gabor receptive fields with input stimulus $u(x,y,t)$. Furthermore, $A_{x_0,y_0}(\omega_x,\omega_y,t)$ and $\phi_{x_0,y_0}(\omega_x,\omega_y,t)$ are, respectively, the *local amplitude* and *local phase* defined at location (x_0,y_0) as a function of the frequency pair (ω_x,ω_y) and time (t).

The local phase of the stimulus at position (x_0,y_0) and frequency (ω_x,ω_y) defined in Eq. (1), can be computed as (see the Appendix A.1 for the definition of arctan2)

$$\begin{aligned} &\phi_{x_0,y_0}(\omega_x,\omega_y,t) \\ &\quad = \arctan2\left(b_{x_0,y_0}(\omega_x,\omega_y,t), a_{x_0,y_0}(\omega_x,\omega_y,t)\right), \end{aligned} \tag{2}$$

The time derivative of the local phase $\phi_{x_0,y_0}(\omega_x,\omega_y,t)$ amounts to

$$\frac{\mathrm{d}\phi_{x_0,y_0}}{\mathrm{d}t}(\omega_x,\omega_y,t) = \frac{\mathcal{T}^1 u(x,y,t)}{\mathcal{T}^2 u(x,y,t)}, \tag{3}$$

where

$$\mathcal{T}^1 u(x, y, t) = a_{x_0,y_0}(\omega_x, \omega_y, t)\frac{\mathrm{d}b_{x_0,y_0}}{\mathrm{d}t} - b_{x_0,y_0}(\omega_x, \omega_y, t)\frac{\mathrm{d}a_{x_0,y_0}}{\mathrm{d}t} \tag{4}$$

and

$$\mathcal{T}^2 u(x, y, t) = a^2_{x_0,y_0}(\omega_x, \omega_y, t) + b^2_{x_0,y_0}(\omega_x, \omega_y, t) + \varepsilon. \tag{5}$$

Here ε is a small constant added to avoid division by zero. In other words, the change of phase at location (x_0, y_0) as a function of the frequency (ω_x, ω_y) is a feedforward DNP. Thus, the phase-based motion detector circuit briefly reviewed in Appendix A.2 is a Divisive Normalization Processor. Note also that in Appendix A.2, the time derivative of the local phase $\phi_{x_0,y_0}(\omega_x, \omega_y, t)$ is directly computed in (36).

We also note the striking similarity between the functional form of $\mathcal{T}_1$ and the *elaborated* Reichardt detector, briefly reviewed in Appendix A.1, long considered to be a model of elementary motion detection in insects (Hassenstein and Reichardt 1956). The numerator also takes the exact form of opponent energy in the motion energy model (Adelson and Bergen 1985) when the temporal filter is properly chosen. Note that the motion energy model is equivalent to the elaborated Reichardt detector (van Santen and Sperling 1985). The former has been widely used as a model of motion detector in the mammalian brain (Grzywacz et al. 1990; Simoncelli and Heeger 1998; Burge and Geisler 2015).

Due to the nature of second-order Volterra processing of the Reichardt and the motion energy detectors, their responses strongly depend on the brightness and contrast of the input visual scene. While the motion energy detector extracts the amplitude of a visual scene for certain *spatio-temporal* frequencies, the divisive normalization processor extracts the gradient of phase at certain *spatial* frequencies that are independent of the amplitude. Note that processing of the latter is still spatio-temporal.

2.2 Estimating the magnitude of velocity and the direction of motion

In Lazar et al. (2016), we devised a criterion to detect motion and the direction of motion based on the Phase Motion Indicator (PMI), a construct built upon the Radon transform of the derivative of phase across all frequencies. We extend the construct here to extract both the direction and velocity of motion, and derive the condition when velocity can be estimated accurately.

We first remind the reader about the method of computation of the Radon transform of the derivative of the phase. We evaluate the Radon transform of $\frac{\mathrm{d}\phi_{x_0,y_0}}{\mathrm{d}t}(\omega_x, \omega_y, t)$ over a circular bounded domain $C = \left\{(\omega_x, \omega_y) | \omega_x^2 + \omega_y^2 \le r^2\right\}$ as

$$\left(\mathcal{R}\frac{\mathrm{d}\phi_{x_0,y_0}}{\mathrm{d}t}\right)(\rho, \theta, t) = \int_{\mathbb{R}} \frac{\mathrm{d}\phi_{x_0,y_0}}{\mathrm{d}t}(\rho\cos\theta + s\sin\theta, \rho\sin\theta - s\cos\theta, t) \cdot \cdot \mathbb{1}_C(\rho\cos\theta + s\sin\theta, \rho\sin\theta - s\cos\theta)\, ds, \tag{6}$$

where r, $0 \le r \le \pi$ rad/pixel is the maximum frequency and π is the maximum bandwidth of the visual field. Furthermore, $-r \le \rho \le r$, $0 \le \theta < \pi$, and

$$\mathbb{1}_C(\omega_x, \omega_y) = \begin{cases} 1, & \text{if } (\omega_x, \omega_y) \in C \\ 0, & \text{otherwise.} \end{cases} \tag{7}$$

Here, ρ and θ determines the line L over which the integral is computed. ρ is the distance (can be both positive and negative) of the line L from the origin, and θ is the angle the normal vector to L makes with the ω_x-axis. s is the coordinate on the line L with the origin set to the point with the shortest distance to the origin.

The reason to use only the values of the derivative of phase over the circular domain is to consider the same maximum frequency r in all directions. As we will see later, this maximum frequency will determine the highest velocity that can be correctly estimated.

If motion occurs around (x_0, y_0) at a velocity of $\mathbf{v}(t) = (v_x(t), v_y(t))$, the phase gradient is approximately

$$\frac{\mathrm{d}\phi_{x_0,y_0}}{\mathrm{d}t}(\omega_x, \omega_y, t) = -v_x(t)\omega_x - v_y(t)\omega_y$$

(see also Appendix A.2). Then,

$$\left(\mathcal{R}\frac{\mathrm{d}\phi_{x_0,y_0}}{\mathrm{d}t}\right)(\rho, \theta, t) = \int_{\mathbb{R}} \left[-v_x(t)\,(\rho\cos\theta + s\sin\theta) - v_y(t)\,(\rho\sin\theta - s\cos\theta)\right] \cdot \cdot \mathbb{1}_C\,(\rho\cos\theta + s\sin\theta, \rho\sin\theta - s\cos\theta)\,\mathrm{d}s. \tag{8}$$

We note that since

$$\int_{\mathbb{R}} s\,\mathbb{1}_C\,(\rho\cos\theta + s\sin\theta, \rho\sin\theta - s\cos\theta)\,\mathrm{d}s = 0, \tag{9}$$

we have

$$\left(\mathcal{R}\frac{d\phi_{kl}}{dt}\right)(\rho, \theta, t) = \rho\left(-v_x(t)\cos\theta - v_y(t)\sin\theta\right) \cdot \mathfrak{c}(\rho, \theta) = -\rho\sqrt{v_x^2(t) + v_y^2(t)}\cos(\theta - \arctan2(v_y(t), v_x(t))) \cdot \mathfrak{c}(\rho, \theta), \tag{10}$$

where

$$\mathfrak{c}(\rho,\theta)=\int_{\mathbb{R}}\mathbb{1}_C\left(\rho\cos\theta+s\sin\theta,\rho\sin\theta-s\cos\theta\right)ds. \tag{11}$$

The PMI is defined as

$$\mathrm{PMI}_{x_0,y_0}(t)=\max_{\theta\in[0,\pi)}\int_{-r}^{r}\left|\frac{(\mathcal{R}\frac{\mathrm{d}\phi_{kl}}{\mathrm{d}t})(\rho,\theta,t)}{\mathfrak{c}(\rho,\theta)}\right|d\rho, \tag{12}$$

with

$$\hat{\theta}_{x_0,y_0}(t)=\operatorname*{argmax}_{\theta\in[0,\pi)}\int_{-r}^{r}\left|\frac{(\mathcal{R}\frac{d\phi_{x_0,y_0}}{dt})(\rho,\theta,t)}{\mathfrak{c}(\rho,\theta)}\right|d\rho. \tag{13}$$

If (10) holds, the PMI amounts to

$$\begin{aligned}\mathrm{PMI}_{x_0,y_0}(t)&=2\int_0^r\rho\sqrt{v_x^2(t)+v_y^2(t)}d\rho\\&=r^2\sqrt{v_x^2(t)+v_y^2(t)}.\end{aligned} \tag{14}$$

Thus, the PMI is proportional to the magnitude of the motion velocity if motion occurs around the point (x_0, y_0). Furthermore, the angle maximizing the PMI in (12) is given by (see also Eq. (10))

$$\hat{\theta}_{x_0,y_0}(t)=\arctan2(v_y(t),v_x(t))\ \text{modulo}\ \pi. \tag{15}$$

Therefore, $\hat{\theta}_{x_0,y_0}(t)$ is the angle of motion. The direction of motion along the angle $\hat{\theta}_{x_0,y_0}(t)$ is determined by the sign of $(\mathcal{R}\frac{d\phi_{kl}}{dt})(\rho,\hat{\theta}_{x_0,y_0},t)$ for $\rho>0$. According to (10), if this sign is -1, then the direction of motion is $\hat{\theta}_{x_0,y_0}(t)$. If this sign is 1, then the direction of motion is $\hat{\theta}_{x_0,y_0}(t)+\pi$.

It is important to recognize that the planar structure in $\frac{\mathrm{d}\phi_{x_0,y_0}}{\mathrm{d}t}$ is limited to certain frequencies, due to phase wrapping. For example, translation along the x-axis between two consecutive frames by k units will result in a phase shift at frequency ω_x by $k\omega_x$. If $k\omega_x>\pi$, then the detected phase change is indistinguishable from $k\omega_x$ modulo 2π. This is the reason why we use only the values of the derivative of phase restricted in a circular domain. Thus, the maximum velocity of motion that can be accurately extracted is $kr\leq\pi$ or $k\leq\frac{\pi}{r}$ in pixels per frame, where r is the maximum frequency ω_x can take within the domain C.

In the neural circuit of early visual system of the fruit fly brain, the inputs and all processing are in the analog domain. That is, neurons communicate through graded potential values rather than spikes. Therefore, in such an analog circuit, phase aliasing does not occur.

2.3 Intensity and contrast level sensitivity of local phase-based motion detectors

In this section, we demonstrate the effectiveness of the motion detector operating on visual stimuli with a wide range of brightness and contrast levels. We then show that its performance will be limited by how well visual stimuli are encoded in the photoreceptors.

We first tested the phase-based motion detector using a video sequence of the change detection dataset (Goyette et al. 2012). Fig. 2(top row) shows 5 representative frames of the original monochrome video sequence. From this sequence, we constructed a video sequence in which the light intensity was artificially increases by a factor of 10 after every 100 frames (see Appendix B for the detailed procedure). A sample frame for each brightness level of the new video sequence is depicted in Fig. 2(2nd row). The pixels of the frames in the first and second rows of Fig. 2 are in one-to-one correspondence.

Example results of evaluating the local phase-based motion detection algorithm are shown in Fig. 2(2nd row). The red arrows indicate the motion detected at the chosen time instance.

In the next example, by increasing the Michelson contrast (Michelson 1927) of the video sequence with a fixed brightness level, we tested the performance of the local phase-based motion detector under different contrast levels. We created a video sequence in which the local contrast was artificially increased by a factor of 2 after every 100 frames (see Appendix B for the detailed procedure). A sample frame at different contrast levels for the video sequence is depicted in Fig. 2(3rd row). Red arrows indicate the motion detected at the chosen time instance.

Figure 2 demonstrates that the local phase-based motion detector can robustly detect motion emerging in visual fields with a wide range of brightness and contrast levels. The underlying assumption here is that the light sensor is ideal and has no saturation level. In reality, fly photoreceptors have a limited dynamic range. Once the stimulus is large enough, the output of a photoreceptor becomes saturated. This leaves no room for a motion detector to operate accurately. For example, the response of fly photoreceptors (and vertebrate cones) to different brightness levels typically follows a sigmoid (Sterling and Laughlin 2015). In Fig. 3 after encoding the video recording with a typical sigmoidal function with a linear range covering 2 orders of magnitudes, the motion detector can no longer detect motion robustly at brightness saturation levels.

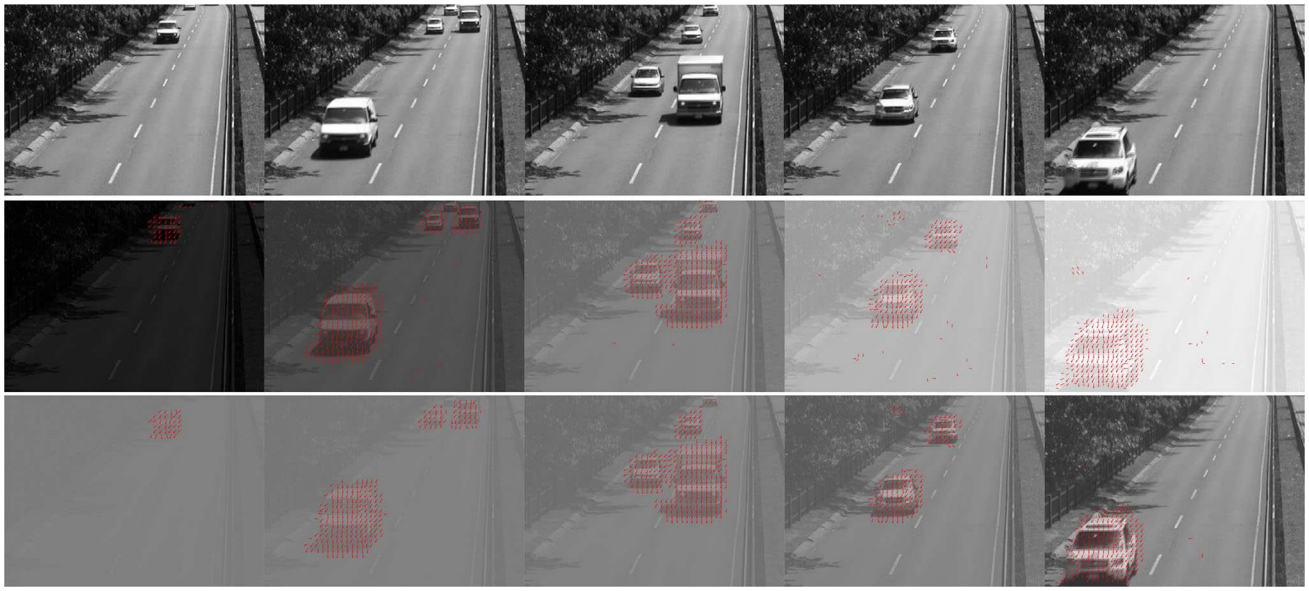

Fig. 2 Phase-based motion detector operating on a visual stimulus under different brightness and contrast levels. (top row) The original video sequence with 8-bit pixel precision (Goyette et al. 2012). 5 representative frames of the video sequence are displayed. (2nd row) Each of the 5 frames exhibits one of the five light intensity levels that are increasing by a factor of 10 in each column. The detected motion is indicated by the red arrows. (3rd row) Each of the 5 frames exhibits one of the five contrast levels that are increasing by a factor of 2 in each column (see the text for more details). The detected motion is indicated by the red arrows

Fig. 3 Motion detection applied onto the same video sequence as in the second row of Fig. 2 after preprocessing by a sigmoidal function. The motion detector can no longer detect motion at very low and very high brightness levels

3 Modeling intensity and contrast gain control with divisive normalization processors

In order for the motion detector described in Sect. 2 to operate in a dynamically changing environment with a wide range of light intensity and contrast levels, there is a need for controlling the output range of the photoreceptors (Sterling and Laughlin 2015). In this section, we present a DNP modeling the intensity and contrast gain control at the photoreceptor/amacrine cell layer in the fruit fly lamina.

3.1 Divisive normalization processors modeling the photoreceptor/amacrine cell layer

Photoreceptors alone (see Fig. 1) cannot achieve effective intensity and contrast gain control if they operate independently. To incorporate spatial information, neurons interconnecting photoreceptors in a neighborhood are needed. Amacrine cells are a perfect candidate. They are interneurons local to the Lamina neuropil, and their processes innervate multiple cartridges. They form reciprocal synapses with photoreceptors in these cartridges.

To model the interaction of photoreceptor axon terminals and amacrine cells, we have introduced divisive normalization processors in the space-time domain (Lazar et al. 2020). Figure 4 depicts a schematic diagram of the spatio-temporal DNP modeling the photoreceptor/amacrine cell layer. Here, the spatio-temporal DNP consists of parallel temporal DNPs with the added cross-channel feedback normalization/gain control provided by the MVP blocks. The temporal DNP blocks model the local photoreceptor/amacrine cell interaction and the MVP blocks model the spatio-temporal photoreceptor/amacrine cell feedback in a restricted spatial neighborhood of the photoreceptor/amacrine cell layer.

Given the input to each of the photoreceptors $u^i(t), i = 1, 2, \ldots, N$, the output of the spatio-temporal DNP $v^i(t), i =$

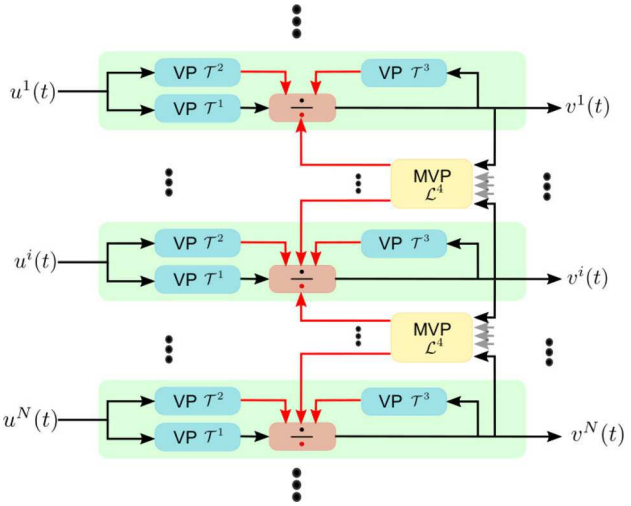

Fig. 4 A diagram of the spatio-temporal DNP modeling the photoreceptor/amacrine cell layer. The DNP consists of N channels. In channel i, the input $u^i(t)$ is processed by two Volterra Processors (VPs), $\mathcal{T}^1$ and $\mathcal{T}^2$, and fed into a division unit. The output $v^i(t)$ is processed by a third VP $\mathcal{T}^3$ and fed into the same division unit. Outputs from channels in a neighbourhood are jointly processed by a Multi-input Volterra Processor (MVP) and fed into the division units of all the corresponding channels. The black arrow inputs to the division unit are passed to the numerator, and the red arrow inputs are summed in the denominator

$1, 2, \ldots, N$, satisfies the equations:

$$v^i(t) = \frac{\mathcal{T}^1 u^i}{\mathcal{T}^2 u^i + \mathcal{T}^3 v^i + \mathcal{L}^4 \mathbf{v}}, \tag{16}$$

for $i = 1, 2, \ldots, N$, where

$$\begin{aligned}\left(\mathcal{T}^l u^i\right)(t) &= b^l + \int_{\mathbb{R}} h_1^l(s) u^i(t-s)\mathrm{d}s \\ &\quad + \int_{\mathbb{R}^2} h_2^l(s_1, s_2) \\ &\quad u^i(t-s_1)u^i(t-s_2)ds_1 ds_2, \quad l = 1, 2,\end{aligned} \tag{17}$$

$$\begin{aligned}\left(\mathcal{T}^3 v^i\right)(t) &= b^3 + \int_{\mathbb{R}} h_1^3(s) v^i(t-s)\mathrm{d}s \\ &\quad + \int_{\mathbb{R}^2} h_2^3(s_1, s_2) \\ &\quad v^i(t-s_1)v^i(t-s_2)\mathrm{d}s_1\mathrm{d}s_2,\end{aligned} \tag{18}$$

and

$$\begin{aligned}(\mathcal{L}^4 \mathbf{v})(t) = b^4 &+ \sum_{i=1}^{N}\left(\int_{\mathbb{R}} h_1^{i4}(s) v^i(t-s)\mathrm{d}s\right) \\ &+ \sum_{i=1}^{N}\sum_{j=1}^{N}\left(\int_{\mathbb{R}^2} h_2^{ij4}(s_1, s_2) v^i(t-s_1)v^j(t-s_2)\mathrm{d}s_1\mathrm{d}s_2\right),\end{aligned} \tag{19}$$

with $\mathbf{v}(t) = \left[v^1(t), v^2(t), \ldots, v^N(t)\right]^T$. Here, $b^l, l = 1, 2, 3, 4$, are zero'th-order Volterra kernels, $h_1^l, l = 1, 2, 3$, and $h_1^{i4}, i = 1, 2, \ldots, N$, are first-order Volterra kernels, and $h_2^l, l = 1, 2, 3$, and $h_2^{ij4}, i, j = 1, 2, \ldots, N$, are second-order Volterra kernels.

In what follows, we will evaluate the I/O mapping of each DNP type, building upon the ones shown in Fig. 4. We first consider a temporal DNP block with only the feedforward terms. The diagram of the feedforward DNP blocks, shown in Fig. 4, is described by the I/O pair (u^i, v^i) and

$$v^i(t) = \frac{\mathcal{T}^1 u^i}{\mathcal{T}^2 u^i}. \tag{20}$$

For an input with constant intensity value I, the steady state response of the DNP is given by

$$v^i(I) = \frac{a_0 + a_1 I + a_2 I^2}{c_0 + c_1 I + c_2 I^2}, \tag{21}$$

where $a_0 = b^1$, $a_1 = \int_{\mathbb{R}} h_1^1(t)\mathrm{d}t$, $a_2 = \int_{\mathbb{R}^2} h_2^1(t_1, t_2)dt_1 dt_2$ are, respectively, the DC components of the first and second-order Volterra kernels of $\mathcal{T}^1$, and $c_0 = b^2$, $c_1 = \int_{\mathbb{R}} h_1^2(t)\mathrm{d}t$, $c_2 = \int_{\mathbb{R}^2} h_2^2(t_1, t_2)dt_1 dt_2$ are, respectively, the DC components of the first and second-order Volterra kernels of $\mathcal{T}^2$. For simplicity, to ensure that the denominator is always positive, we also assume that the coefficients $a_0, a_1, a_2, c_0, c_1, c_2$ are positive. We consider $v^i(I)$ to be normalized and to take values between 0 and 1. This can be achieved by working with $c_2/a_2 \cdot v^i(I)$ or simply with $v^i(I)$ by setting $c_2 = a_2$. Consequently, the steady state response of a temporal feedforward DNP is a sigmoidal function of $\log_{10}(I)$ with an output range between 0 and 1 (see also Appendix C.1). The steady-state response curve is shown in Fig. 5A for a few values of the a_2/a_1 ratio. We note that the slope of the sigmoid gradually increases from a first-order Volterra kernel ($a_2 = 0$) to a second-order Volterra kernel ($a_1 = 0$). Here, by choosing $c_1 \geq a_1$, we ensured that $v^i(I)$ is a monotonically increasing function (see also Appendix C.1).

By adding the local-feedback term $\mathcal{T}^3 v^i$, we obtain the DNP block with feedforward and local feedback depicted in Fig. 4. The output of this temporal DNP can be expressed as

$$v^i(t) = \frac{\mathcal{T}^1 u^i}{\mathcal{T}^2 u^i + \mathcal{T}^3 v^i}, i = 1, 2, \ldots, N. \tag{22}$$

Its steady-state response is given by

$$v^i(I) = \begin{cases} \frac{-C+\sqrt{C^2-4d_1A}}{2d_1} & \text{if } d_2 = 0, \\ -\frac{1}{3d_2}\left(d_1 + D\frac{-1+\sqrt{-3}}{2} + \frac{\Delta_0}{D\frac{-1+\sqrt{-3}}{2}}\right) & \text{if } d_2 \neq 0, \Delta > 0, \\ -\frac{1}{3d_2}\left(d_1 + D + \frac{\Delta_0}{D}\right) & \text{if } d_2 \neq 0, \Delta \leq 0 \end{cases} \tag{23}$$

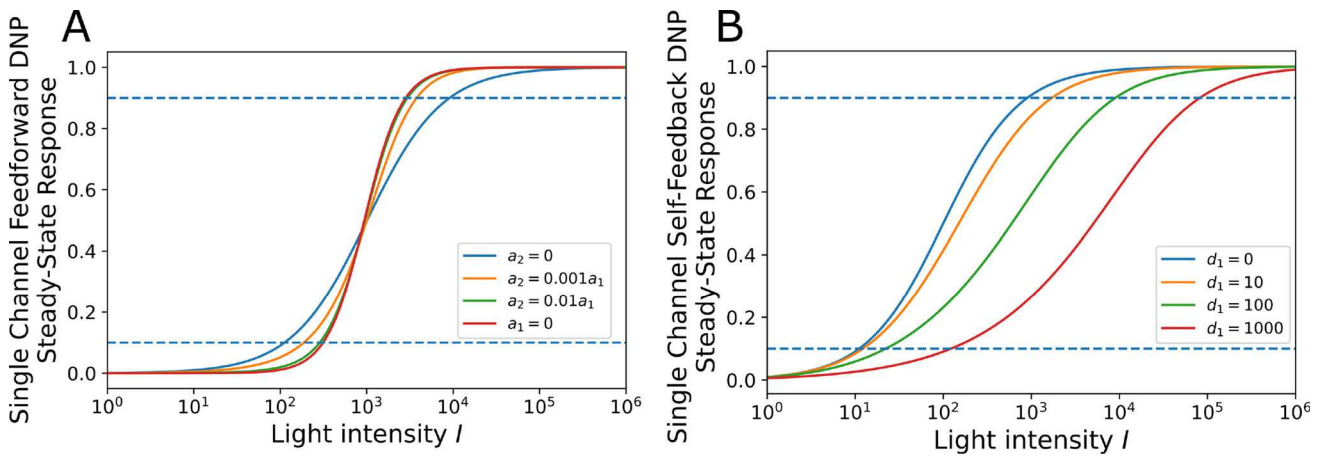

Fig. 5 The steady-state response curve as a function of the parameters of a temporal DNP. **a** For the feedforward DNP block shown in Fig. 4, increasing the ratio between a_2 and a_1 increases the slope of the sigmoid. **b** For the DNP block with feedforward and local feedback as shown in Fig. 4, increasing the feedback strength decreases the slope of the sigmoid. Dashed lines indicate the 0.1 and 0.9 levels

where

$$A = -(a_0 + a_1 I + a_2 I^2), \tag{24}$$

$$C = (c_0 + d_0 + c_1 I + c_2 I^2), \tag{25}$$

$$D = \sqrt[3]{\frac{\Delta_1 + \sqrt{-\Delta}}{2}}, \tag{26}$$

$$\Delta = \frac{4\Delta_0^3 - \Delta_1^2}{27 d_2^2}, \tag{27}$$

$$\Delta_0 = d_1^2 - 3 d_2 C, \tag{28}$$

$$\Delta_1 = 2 d_1^3 - 9 d_2 d_1 C + 27 d_2^2 A, \tag{29}$$

with $d_0 = b^3$, $d_1 = \int_{\mathbb{R}} h_1^3(s)ds$ and $d_2 = \int_{\mathbb{R}^2} h_2^3(s_1, s_2)\, ds_1 ds_2$. Again, we assume that the coefficients $d_i > 0, i = 1, 2, 3$. The effect of the feedback on the steady-state response is depicted in Fig. 5B. By increasing d_1 (and/or d_2), the slope of the sigmoid decreases.

Combined with the ratio between a_1 and a_2, we see that a temporal DNP with feedforward and local feedback exhibits a range of gradients of the sigmoid for different parameter choices. This key feature underlies the contrast gain control mechanism exerted by DNPs.

Finally, the output of each channel also depends on the global feedback generated by the MVPs. The $\mathcal{L}^4\mathbf{v}$ term in the denominator of Eq. (16) leads to a shift of the steady-state response curve to the left or right (see Fig. 6 and the discussion below) depending on the strength of the global feedback, that in turn, depends on the inputs to all channels.

In Fig. 6, we depict the mapping of a 16×16 image block at three different light intensity levels, with a factor of 10 between each two consecutive levels. They are shown under the I-axis in both Fig. 6A and B. Without the MVP feedback term, the steady-state response curve of each pixel is fixed on the blue continuous curve for all three light intensity levels (see Fig. 6A). With the MVP, the mapping shifts to the orange curve for the image patch scaled by a factor of 10, and to the green curve for the image patch scaled by a factor of 100. The respective output of the DNPs without or with MVP are shown on the right of Fig. 6A and B, respectively. Figure 7(2nd and 3rd column) shows the outputs of the DNP with the same parameters without and with MVP, respectively, processing an entire image at 3 different brightness levels.

3.2 Adaptive feedback intensity gain control

From the analysis of the previous section, if the input is scaled up by a factor, the output of $\mathcal{L}^4\mathbf{v}$ must be scaled up by approximately the same amount. However, since the value of the output is restricted between 0 and 1, scaling up the MVP feedback is necessary.

By noticing the role that the MVP plays in the I/O curve of the DNP, we devised a DNP with a dynamic feedback term controlling the I/O curve:

$$v^i(t) = \frac{\mathcal{T}^1 u^i}{\mathcal{T}^2 u^i + \mathcal{T}^3 v^i + w}, \tag{30}$$

where

$$\frac{\mathrm{d}w}{\mathrm{d}t} = \alpha \left[\mathcal{L}^4 \mathbf{v}(t) - 0.5(b^4 + r_1 + r_2) \right], \tag{31}$$

with $\alpha > 0$, $r_1 = \sum_{i=1}^{N} \int_{\mathbb{R}} h^{i4}(t)dt$ and $r_2 = \sum_{i=1}^{N} \sum_{j=1}^{N} \int_{\mathbb{R}^2} h^{ij4}(t_1, t_2)dt_1 dt_2$. Eq. (31) centers the steady-state the outputs of the DNP channels to a mean of 0.5, the middle

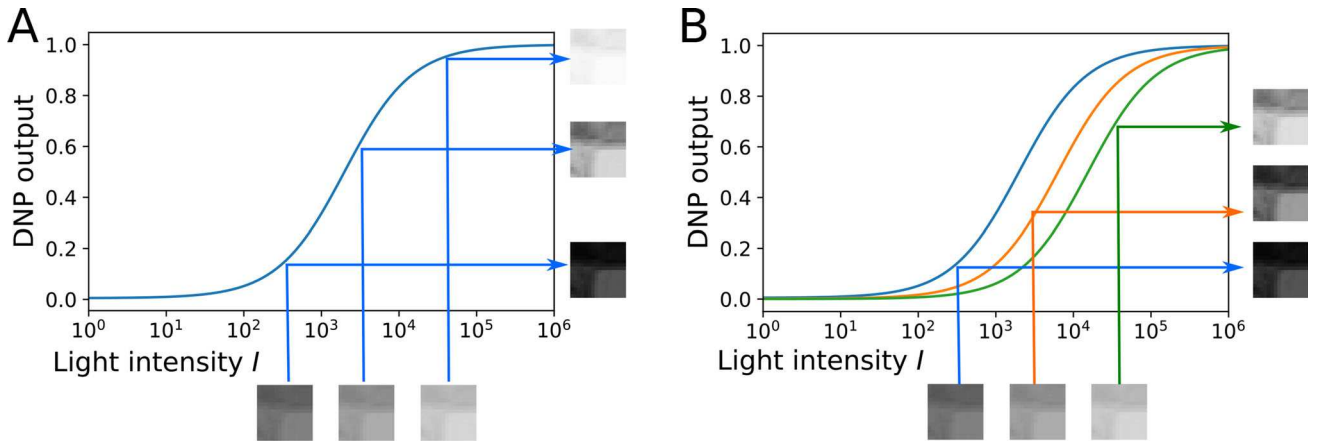

Fig. 6 The effect of the MVP feedback. **A** The steady-state response curve of the DNP without global MVP feedback. The single curve maps three 16 × 16 image patches at different brightness levels (bottom) into the output on the right. **B** With the MVP, the steady-state response curve shifts to the orange curve when mapping the image patch in the middle and to the green curve when mapping the image patch on the right. This allows the DNP to map a larger range of light intensity values onto the linear part the response curve, increasing the local contrast

point of the sigmoid curve. Consequently, the output of the photoreceptor will be distributed on both sides of the center of the sigmoid and will mostly operate in the linear range of the sigmoid.

Figure 8 shows a schematic diagram of the DNP introduced above. We note that Eq. (31) suggests an additional processor in the feedback loop. Such a feedback loop may well be implemented in the molecular domain. For example, photoreceptors operate at a baseline intracellular calcium concentration level. Calcium accumulation can cause this baseline to increase by a factor of more than 1,000 and thereby shift the response curve of the photoreceptor (Song et al. 2012; Sterling and Laughlin 2015).

Figure 7(4th column) displays the output of the adaptive feedback DNP for the inputs in Fig. 7(1st column). We note that the output of this DNP is approximately the same for all 3 inputs with different brightness levels. The DNP also enhances the contrast at the edges.

4 Evaluation of motion detection with gain control preprocessing

In this section, we combine the results in the previous two sections, and model the overall motion detection of the fruit fly early visual system consisting of the retina, lamina and medulla (see Fig. 1A) with a cascade of two DNP processing blocks, as shown in Fig. 1B. The DNP in the first block models the intensity and contrast gain control at the photoreceptors/amacrine cells layer. The second block consists of a DNP for detecting phase changes followed by a phase-based motion detector. This block models the elementary motion detection by the fruit fly medulla/lobula T4/T5 neurons using local phase information.

4.1 Evaluation of motion detection with DNPs

Existing benchmark optic flow datasets are not suitable for evaluating phase-base motion detection algorithms. They either have very few frames (Baker et al. 2011; Geiger et al. 2012; Dosovitskiy et al. 2015) or have a too large displacement between frames (Butler et al. 2012; Mayer et al. 2016) to allow phase changes to be correctly estimated.

In what follows, we will evaluate the phase-based motion detector and compare its performance with other algorithms using videos captured by the authors in a local park. We note that inputs to the fly photoreceptors are analog, and the visual processing in the early visual system of the fruit fly brain is also “analog" as well. To approximate the analog visual world, we captured videos at the highest mobile phone frame rate.

4.1.1 Qualitative evaluation of phase-based motion detectors

To evaluate the motion detector performance, we used two types of video recordings. The first type were shot at 240 frames per second (fps) using a Samsung S23 Plus mobile phone at Full High Definition (FHD) 1920 × 1080 resolution. Each color frame was converted to grey-scale luminance and resized to 480 × 270 before fed into the phase-based motion detector and other motion detection algorithms for comparison. The input pixel values were already gamma corrected when the video sequence was shot with a bit-depth of 8 bit.

Since there was no ground truth for motion in these visual sequences, we visually compared the performance of motion detection using (1) the phase-based motion detection algorithm, (2) the algorithm of Shi and Luo (2018) based on the

Fig. 7 Comparison between DNP processed images at 3 different brightness levels. (1st column) Input image. (2nd column) Output of the DNP without global feedback. (3rd column) Output of the DNP with global feedback. (4th column) Output of the DNP with adaptive feedback. The image in each consecutive row has its brightness scaled by a factor of 10

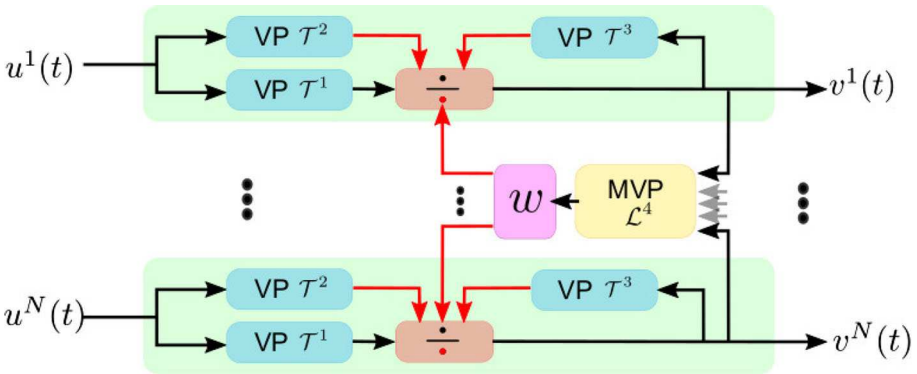

Fig. 8 The schematic diagram of the adaptive feedback DNP model with an additional processor w in the feedback loop. See also the notation in Fig. 4–here, for simplicity, only two channels and one MVP are depicted

motion energy model inspired by the mammalian brain, and (3) the MR-Flow algorithm (Wulff et al. 2017) and (4) the top performing RAFT algorithm in the MPI-Sintel benchmark (Butler et al. 2012). Note that the more recent RAFT algorithm is based on artificial neural networks (Shah and Xuezhi 2021). Motion detection was performed at every other pixel using method (1), on frames subsampled to 240 × 135 by method (2), and on the full 480 × 270 frames by methods (3) and (4). Therefore, the detected motion using methods (3) and (4) had four times the density of the motion detected by methods (1) and (2). All models mentioned above were used in all experiments with the same set of parameters.

For methods (1), (2) and (3) we used consecutive frames when performing motion detection. For method (4), however, we did not obtain any meaningful output due to the small displacement between frames. Therefore, the optic flow algorithm was computed using frames that are 1/30 second apart.

Figure 9(1st column) shows a sample frame of the natural visual sequence with a squirrel moving downward on a bench. A small background movement due to hand movement while shooting the video is also discernible. The phase-based motion detector captured both the movement of the squirrel and the background (2nd column). Its performance is comparable to the MR-Flow (4th column) and the RAFT algorithms

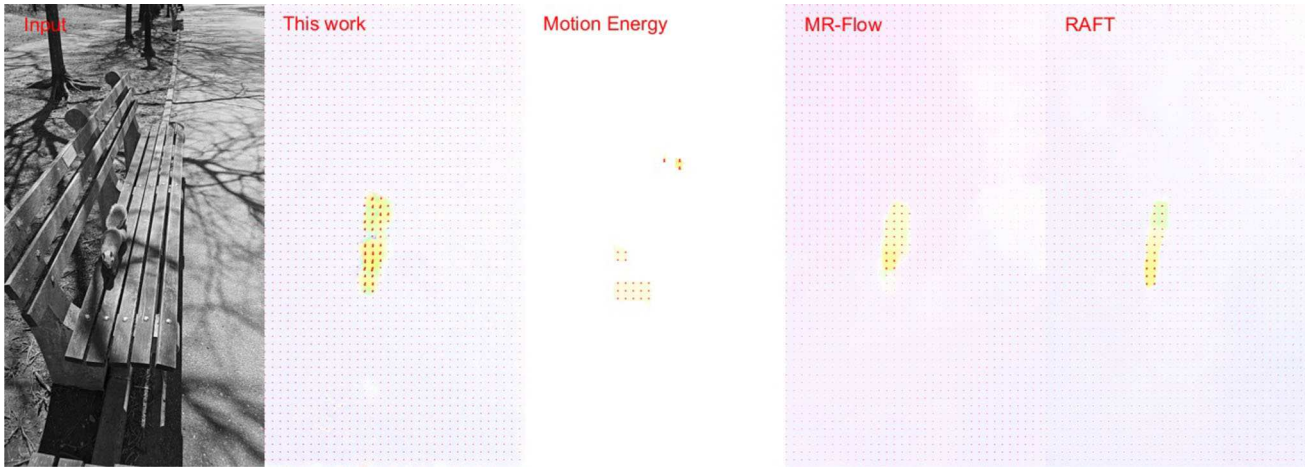

Fig. 9 Comparison of motion detected in a visual sequence shot at 240 fps. (1st column) A sample frame of a 240 fps visual sequence. (2nd column) Motion detected by the phase-based motion detector. (3rd column) Motion detected by the motion energy algorithm (Shi and Luo 2018). (4th column) Motion detected by the MR-Flow algorithm (Wulff et al. 2017). (5th column) Motion detected by the RAFT algorithm (Teed and Deng 2020). Direction of detected motion is indicated by the common color coding convention (Baker et al. 2011) as well as by arrows on a sparser grid. See Video S1 for the full video

(5th column), if not better, and much better than that of the motion energy model (3rd column). The full video is available as Video S1 in the Supplementary Materials. Additional examples can be found in Video S2, 3.

Table 1 specifies the code execution environment of the 4 different algorithms. The processing of the phase-based motion detector is highly efficient on GPUs and can process 170 frames per second. While the motion energy model can be executed at 480 frames per second, we note that the algorithm jointly processes batches of frames. Since the motion energy model requires temporal filtering, it will be slowed down when implemented frame by frame in real-time applications.

The second type of video sequences were shot at 60 fps using the same mobile phone at FHD resolution. The raw sensor data had 13 bit bit-depth. We used only the green color channel of the raw frames to ensure that the pixel values are approximately linear with respect to the light intensity of the visual scenes. Each frame was then resized to 480 $\times$ 270 before fed into motion detectors. These videos exhibit rapid changes in light intensity and contrast across the scene.

Figure 10(1st column) shows a sample frame with a squirrel moving up on a tree trunk in the shadow with the rest of the scene exposed to direct sunlight. To better display the input, Fig. 10(2nd column) depicts the same frame with a gamma correction of 2.2. The phase-based motion detector (3rd column) successfully detected both movement of the squirrel and the background. The motion energy model (4th column) detected the background movement (which is larger than in Fig. 9, but fails to detect the squirrel's motion under the shadow. The MR-Flow algorithm (5th column) and the RAFT algorithm (6th column) provide a smoother background motion but fails to detect the moving squirrel. This is likely due to the fact that the video frames fed into the MR-Flow algorithm had only 8 bit bit-depth and did not have enough precision to discern the image of the squirrel under the shadow. The full video is available as Video S4 in the Supplementary Materials. Additional videos can be found in Video S5, 6.

The results presented so far indicate that the phase-based motion detector performs well under natural light conditions that can significantly vary in the same scene. In Supplementary Video S7–9, we show how the phase-based motion detector performs on video sequences that are subject to additive white Gaussian noise with SNRs ranging from 30 dB to 10 dB. Note that the area with lower contrast is disproportionally affected by noise.

4.1.2 Quantitative evaluation of phase-based motion detectors

In order to evaluate the phase-based motion detector quantitatively, we used the same mobile phone to shoot raw images at a bit-depth of 14 bits and a resolution of 8000 $\times$ 6000. We again only used the green color channel from these raw images. A sample image and its gamma corrected version are shown in Fig. 11.

For each scene, we chose a 256 $\times$ 256 window and moved the window by a specified number of pixel per frame (from 0.25 to 2 with an increment of 0.25), similar to the approach used in Shi and Luo (2018). For speeds of a fractional pixel per frame, we used a cubic spline interpolation. We

Table 1 Comparison of the processing speed of different motion detection algorithms

Algorithm	Environment	Processing speed (fps)
Phase-based (this work)	1× NVIDIA V100 GPU (Python)	170
Motion energy model (Shi and Luo 2018)	1× NVIDIA V100 GPU (Python)	480
MR-Flow (Wulff et al. 2017)	2× Intel Xeon Gold 5120 CPU (14 cores @ 2.2GHz) (Python)	0.013
RAFT (Shah and Xuezhi 2021)	1× NVIDIA V100 GPU (Python)	5.33

The execution of phase-based motion detector, the MR-Flow algorithm and the RAFT algorithm was frame by frame, while the motion energy model is executed in batches of frames. Therefore, the actual real-time processing speed may be lower for the motion energy model

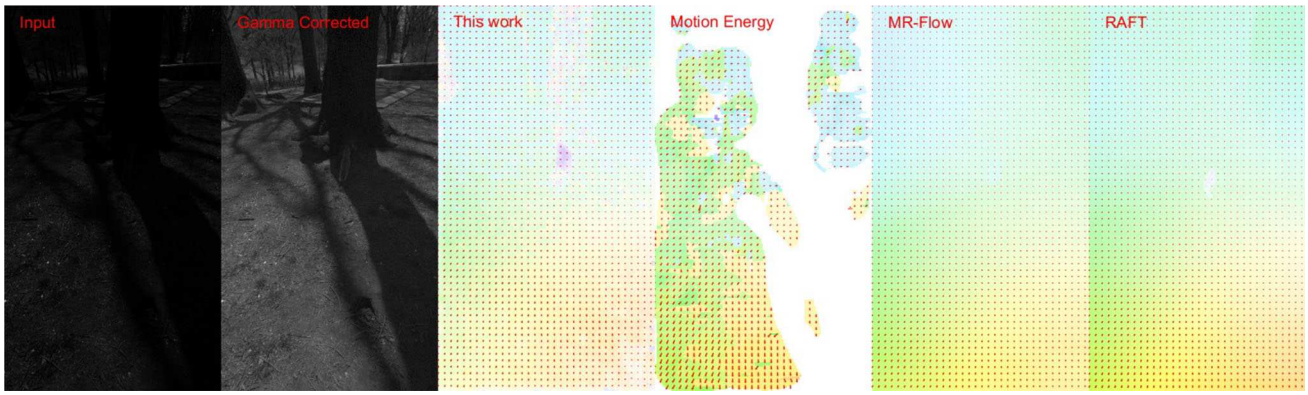

Fig. 10 Comparison of motion detected in a "raw" visual sequence shot at 60 fps. (1st column) A sample "raw" frame of a 60 fps visual sequence. Pixel values are proportional to light intensity. (2nd column) The video frame with gamma correction of 2.2. (3rd column) Motion detected by the phase-based motion detector. (4th column) Motion detected by the motion energy algorithm (Shi and Luo 2018). (5th column) Motion detected by the MR-Flow algorithm (Wulff et al. 2017). (6th column) Motion detected by the RAFT algorithm (Shah and Xuezhi 2021). Direction of detected motion is indicated by the common color coding convention (Baker et al. 2011) as well as by arrows on a sparser grid. See Video S4 for full video

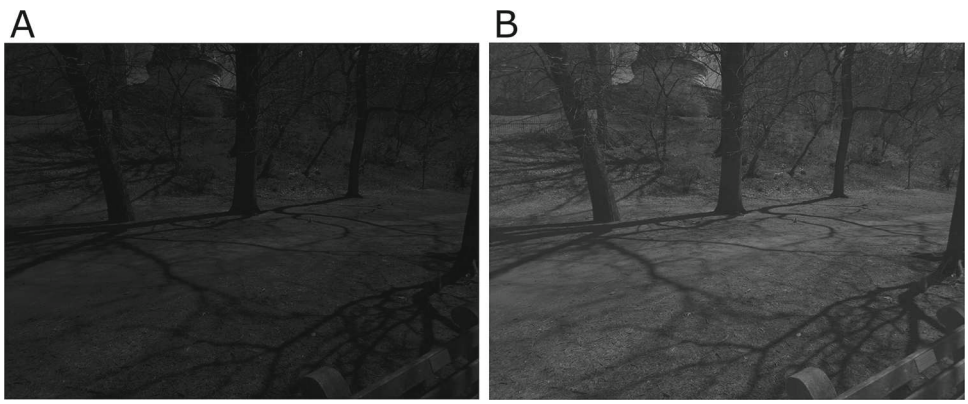

Fig. 11 Sample raw images that are used to generate full screen motion with ground truth direction

then obtained a series of video sequences with ground-truth motion direction and tested how well the phase-based motion detector can detect motion at every other pixel. In Fig. 12A, we show the average angular error (AE) in a polar plot for every tested velocity and angle. AE was largely kept under 2°, with an overall average of 1.09°. Figure 12B shows the end-point error (EPE). The EPE was largely under 0.2 pixels for speeds up to 1.5 pixel/second. We used a circular domain with $r = \frac{5}{8}\pi$, expecting that phase aliasing would occur at velocities larger than 1.6 pixel/frame. As predicted, EPE in Fig. 12B is much larger for motion velocities above 1.5.

4.2 Evaluation of the entire DNP cascade

In Fig. 13, we evaluate the motion detection in a dynamically changing environments scenario with light intensity ranging over 5 orders of magnitude. Figure 13 shows results of motion detection for a video sequence processed by, respectively, the DNP without the MVP feedback, with the MVP feedback,

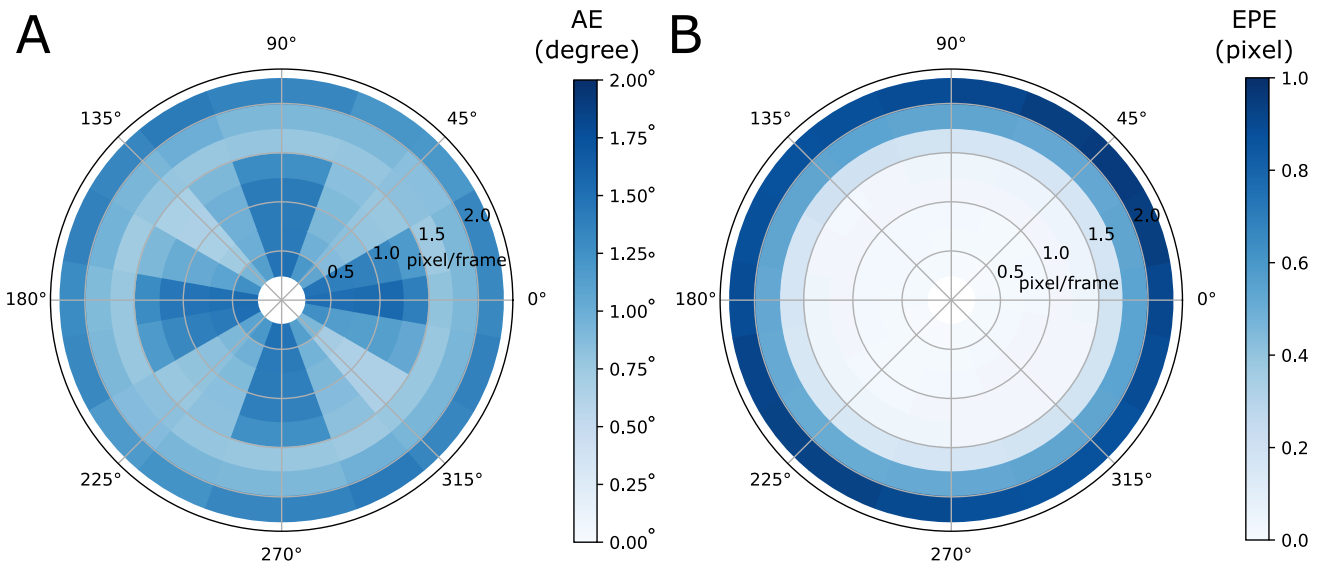

Fig. 12 Average **A** angular error and **B** end-point error of motion detected by the phase-based motion detector. The errors are provided in a polar plot. Each angle represents the error of ground-truth motion moving in the direction indicated by the angle. Each radius shows the motion with speed at 0.25 to 2.0 pixel per frame with an 0.25 increment

and with the MVP adaptive feedback blocks. The brightness of each video sequence increases by a factor of 10 after every 100 frames. A typical frame at each level of brightness is shown for each video sequence in one of the 5 rows in the first column.

In the second column, we show the responses of the DNP without the MVP feedback block. The motion detected is shown in red arrows. We can see that the phase-based motion detector fails to detect motion in the lowest and highest brightness levels, since these responses are saturated.

In the third column, we show the responses of the DNP with MVP feedback block. The motion detected from these responses is shown with red arrows. The DNP outputs are less saturated and the phase-based motion detector is able to robustly detect motion.

In the fourth column, we show the responses of the DNP described in Sect. 3.2. These responses are largely invariant to the input brightness levels. The phase-based motion detector can robustly detect motion from these outputs.

The videos corresponding to Fig. 13 can be found in Supplementary Video S10. Additional video recordings are available in Supplementary Video S11–15. The parameters for testing all these video sequences were kept the same. Therefore, the outputs of the intensity and contrast gain control DNP looks drastically different from that of the DNP without the MVP block and the DNP with the global feedback block. The outputs are in a similar range when using the DNP with the adaptive feedback block across all videos. In most of the tests, however, the final motion detector responses were similar across the three configurations. This is because the phase-based motion detector can still detect motion even if the output of the first DNP has lower contrast, as shown in Fig. 2. However, we expect that motion detection will be more accurate when using the DNP with adaptive feedback.

5 Discussion

The early visual system of the fruit fly, as the vertebrate retina, exhibits highly nonlinear processing of visual information in the analog domain. In this paper, we offered further insights into a class of analog nonlinear circuits, called the divisive normalization processors, first proposed in Lazar et al. (2020), and the roles they can play in modeling early visual system of the fruit fly. We demonstrated that two cascaded DNPs can effectively model the robust motion detection capability of the early visual system in the fruit fly. The two stages of DNPs, as schematically shown in Fig. 1B, implement intensity/contrast gain control and elementary motion detection, respectively.

We first introduced a feedforward DNP underlying the phase-based motion detector modeling motion detection in the fruit fly. The DNP modeling of the motion detection circuit allow us to directly relate the phase-based motion detector with the classical models of motion detection, such as the Reichardt motion detector and the motion energy model. We have extended our previous approach and developed a method of estimating both direction and velocity magnitude of motion and provided the condition under which the estimation of velocity is accurate. The performance of

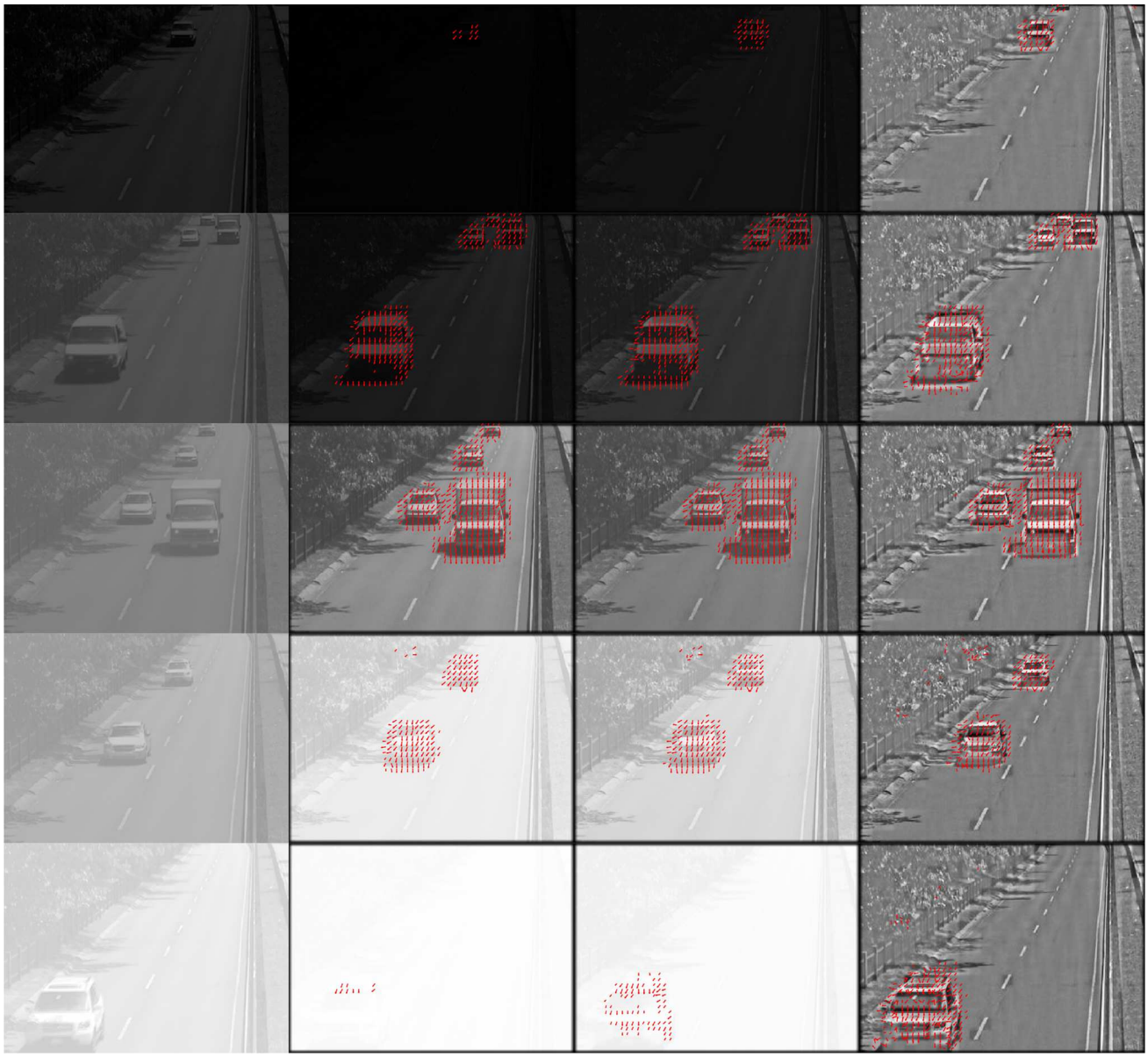

Fig. 13 Evaluation of motion detection using a cascade of two DNPs. The video sequence has 500 frames. The brightness of the video increases by a factor of 10 after every 100 frames. The resulting video sequence has brightness that spans 5 orders of magnitude. (1st column) A typical frame at a different brightness level in each row. (2nd column) DNP output without the MVP feedback block. Red arrows indicate the motion detected from the DNP output. The motion detector cannot robustly detect motion under very low (first row) or very high (last row) brightness conditions. (3rd column) DNP output with the MVP feedback block. Red arrows indicate the motion detected from the DNP output. The output of the DNP with MVP feedback is slightly less saturated than the output of DNP without the MVP feedback. The phase-based motion detector can pick up the subtle differences after the DNP processes and detects motion. (4th column) The output of the adaptive feedback DNP. Red arrows indicate the motion detected. The output of this DNP is mostly invariant of the input brightness level. See also Supplementary Video S10

the motion detection circuit was quantitatively validated and extensively compared to other motion detection algorithms, such as those based on the motion energy model, as well as optic flow algorithms.

Currently many hardware implementations of bioinspired visual motion detection are based on the motion energy model of the mammalian brain (Orchard and Etienne-Cummings 2014; Shi and Luo 2018). They require a pre-processing stage to compensate for large luminance/contrast variation. For example, in Shi and Luo (2018), a threshold is first applied onto each filtered frame to enhance and normalize contrast. However, the threshold value depends on

the values of luminance and contrast and a fixed threshold does not work well under all conditions. On the contrary, our model works robustly under widely different intensity and contrast environmental conditions using throughout the same set of parameters .

As demonstrated in the supplementary videos, the proposed motion detector is very effective on a range of motion velocities. In contrast, at very low speed, the optic flow algorithm we tested missed the movement of small objects whereas the phase-based motion detector was able to detect them. However, since the phase-based motion detector only processes local information, it inevitably cannot detect large displacements between frames as extensively present in most of the standard optic flow benchmarks that typically feature low frame rates. The same also applies to most of the models based on localized spatial filters such as the Gabor filters, including those derived from the motion energy model. While using additional processing after the elementary motion detection stage can be used to combine local motion information across space, frame-based processing does not appear to be a natural solution for living organisms.

As discussed in many of the examples, the phase-based motion detector takes advantage of very high data rates, possibly approaching the analog domain. Note that the fly early visual system processes visual scenes in the analog domain. Thus, our model suggests a possible reason why processing in both the early visual system of the fruit fly and the retina of vertebrates is analog. That is, the analog processing is necessary to accurately extract motion in the early visual systems. Hence, it is of interest to implement this algorithm in analog neuromorphic hardware (Stocker 2006) in the future.

While recent advances in optic flow algorithms based on artificial neural networks has led in benchmark tests to high accuracy in motion estimation, the amount of data required for training is still very high. Moreover, although deep learning has significantly improved the run time over traditional optic flow methods, they are still far from achieving real-time processing capability. In contrast, the phase-based motion detection algorithm requires no training and can process video sequences in real-time. The examples in the supplementary videos show that, under natural environments the phase-based motion detector can detect motion that is undetectable by algorithms based on deep learning. Most likely the datasets used to train these models did not contain such examples and the trained model cannot generalize robustly under these conditions.

We showed that the performance of the DNP modeling the very first stage of visual processing, i.e., the photoreceptor/amacrine cell layer, was critical for subsequent motion detection in dynamic environments where the light intensity can vary by orders of magnitudes. We provided a detailed characterization of the I/O of this class of DNPs with respect to different parameter choices, and gave insights into how they perform intensity and contrast gain control (Lazar et al. 2020).

Furthermore, we advanced here a DNP architecture with an adaptive feedback MVP block. We showed that the response of the DNP with adaptive MVP feedback is largely independent of the background light intensity levels in the visual scenes. Due to its adaptive nature, the circuit is robust to variation of parameters.

While the structure of DNP modeling the photoreceptor/amacrine cell layer directly corresponds to the anatomical structure of the neural circuit, as depicted in Fig. 1A and B, the phase-based motion detection circuit is more abstract. Notably, the fly motion detection circuit is separated into ON and OFF pathways. In the ON pathway (see Fig. 1 (in yellow)), visual motion resulting in a brightness increase is detected. Visual motion that causes a brightness decrease is detected in the OFF pathway (see Fig. 1 (in purple)). It has been shown that a full Reichardt motion detector can be subdivided into a two-quadrant model (Eichner et al. 2011) where one quadrant corresponds to the ON pathway processing brightness increases and the other corresponds to the OFF pathway processing brightness decreases. Mapping the proposed motion detection circuit into these pathways is of future interest.

Overall, the work presented here demonstrated the power of DNPs in modeling computation arising in visual processing circuits of the fly brain. As fly visual circuits that are involved in other visual tasks have similar connectivity features such as extensive feedback loops, we expect that divisive normalization can serve as a key computational building block in visual processing in the bee/ zebrafish vision systems and the retina of vertebrates (Sanes and Zipursky 2010). Additional types of DNPs and further understanding of their properties will shed light on employing DNP-based algorithms in computer vision.

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/s00422-023-00972-x.

Author Contributions AAL contributed to conceptualization, developing research directions, investigation, methodology, writing—original draft, writing—review and editing, resources, supervision, funding acquisition, project administration. YZ contributed to conceptualization, developing research directions, investigation, simulation, visualization, methodology, writing - original draft, writing—review and editing, funding acquisition, project administration.

Funding The research reported here was supported by the National Science Foundation under grant #2024607.

Declarations

Conflict of interest The authors declare no competing interest.

A Models of elementary motion detectors in fruit flies

We briefly review in Appendix A.1 the structure of the elementary motion detectors and in Appendix A.2, the structure of the local phase-based motion detector.

A.1 Reichardt motion detector

Modeling elementary motion detection circuits in fruit flies has been strongly influenced by the classical work of Hassenstein and Reichardt (1956) and Barlow and Levick (1965). Figure 14 depicts the *elaborated* Reichardt motion detector of van Santen and Sperling (1985). With properly chosen parameters, this motion detector is equivalent to the motion energy detector of Adelson and Bergen (1985) modeling direction sensitive complex cells in the mammalian visual cortex.

The Reichardt motion detector can explain many optomotor responses (Borst 2014). Electrophysiological observations of lobula plate tangential cells located downstream of these elementary motion detectors are sensitive to wide-field motion. They are, however, extremely sensitive as well to the overall brightness and contrast level of visual scenes. Since light intensity during day and night may vary by at least 6 orders of magnitude (Cronin et al. 2014), the Reichardt detector, by itself, cannot match the robustness of the *Drosophila* motion detection circuits.

An schematic diagram of the *elaborated* Reichardt motion detector is shown in Fig. 14. The output $v(t)$, a functional of the input stimulus $u(x, y, t)$, can be written as

$$v(t) = \int_{\mathbb{R}^6} h_1(x_1, y_1)h_2(x_2, y_2)\,[g_2(s_1)g_1(s_2) - g_1(s_1)g_2(s_2)]\cdot u(x_1, y_1, t - s_1)u(x_2, y_2, t - s_2)\mathrm{d}x_1\mathrm{d}y_1\mathrm{d}s_1\mathrm{d}x_2\mathrm{d}y_2\mathrm{d}s_2, \tag{32}$$

where h_1, h_2 denote the two spatial filters and g_1, g_2 denote the two temporal filters shown in Fig. 14. Note that the output of the *elaborated* Reichardt motion detector is a quadratic function of the input.

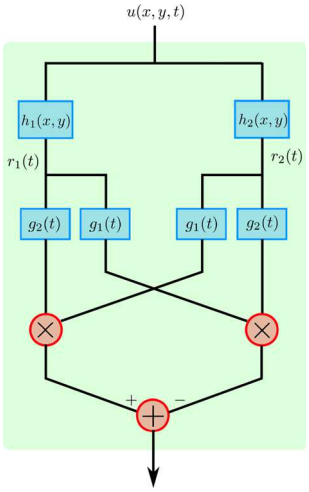

Fig. 14 A schematic diagram of the *elaborated* Reichardt motion detector

A.2 Local phase-based motion detector

In Lazar et al. (2016), we proposed a motion detector based on local phase information. With processing in the phase domain, the operation of the motion detector does not depend on the scale of the Fourier amplitude of the input stimuli. In what follows, we provide a brief overview of the local phase-based motion detector.

The local phase-based motion detector consists of 2 staged circuits, for detecting motion surrounding a reference point in the visual field. In the first stage, the local phase (defined below) and its time derivative are extracted. In the second stage, the extracted phase is processed to indicate the presence and the direction of motion. This two-stage detector can be repeated in parallel for any number of reference points covering the entire visual field.

If $\mathbf{v} = \big(v_x(t), v_y(t)\big)$ is the instantaneous motion velocity pair, the translated signal considered here amounts to,

$$u(x, y, t) = u(x - s_x(t), y - s_y(t), 0), \tag{33}$$

where

$$s_x(t) = \int_0^t v_x(s)\mathrm{d}s, \tag{34}$$

$$s_y(t) = \int_0^t v_y(s)\mathrm{d}s. \tag{35}$$

Consequently, the motion velocity and the time derivative of the phase satisfy the differential equations

$$\frac{\mathrm{d}\phi_{x_0,y_0}}{\mathrm{d}t}(\omega_x, \omega_y, t) = -\frac{\mathrm{d}s_x(t)}{\mathrm{d}t}\omega_x - \frac{ds_y(t)}{dt}\omega_y$$

$$+\mathfrak{v}_{x_0,y_0}(\omega_x,\omega_y,t), \tag{36}$$

for all $(\omega_x,\omega_y)\in\mathbb{R}^2$ with the initial condition $\phi_{x_0,y_0}(\omega_x,\omega_y,0)$, where $\mathfrak{v}$ is assumed to be a small term (Lazar et al. 2016).

A.3 Definition of arctan2

$$\text{arctan2}(y,x)=\begin{cases}\arctan\left(\frac{y}{x}\right) & \text{if } x>0,\\ \arctan\left(\frac{y}{x}\right)+\pi & \text{if } x<0 \text{ and } y\geq 0,\\ \arctan\left(\frac{y}{x}\right)-\pi & \text{if } x<0 \text{ and } y<0,\\ \frac{\pi}{2} & \text{if } x=0 \text{ and } y>0,\\ -\frac{\pi}{2} & \text{if } x=0 \text{ and } y<0,\\ \text{undefined} & \text{if } x=0 \text{ and } y=0,\end{cases} \tag{37}$$

B Constructing video sequences with different light intensity and contrast levels

Here we provide details for constructing visual scenes with different light intensity and contrast levels.

B.1 Constructing video sequences with different light intensity levels

Each pixel of the original video sequence had 8-bit precision. We note that the pixel values were already gamma corrected during recording. Prior to testing the motion detection algorithm, we performed preprocessing of the original video sequences to (1) undo the gamma correction effect by transforming the pixel values to be approximately proportional to the light intensity levels and, (2) change brightness levels. First, we scaled the value of each pixel from between 0 and 255 to between 0 and 1 and denoted the scaled video sequence by u_0. We then generated an input video sequence as $u(x,y,t)=10^{u_0(x,y,t)}$ with an approximately scaled version of the light intensity of the original visual scene. The light intensity of the video sequence was furthermore artificially increased by a factor of 10 after every 100 frames, i.e., $u(x,y,t)=10^{u_0(x,y,t)+k}$, $k=1,2,3,4$.

To help discern the contents of the visual scenes at every brightness level, Fig. 2(2nd row) shows $log_{10}(u)$ instead of u itself, with $log_{10}(1)$ displayed as pure black and $log_{10}(10^6)$ displayed as pure white.

B.2 Constructing video sequences with different contrast levels

We generated the video sequences $u_i(x,y,t)=100\cdot 10^{c^i(u_0(x,y,t)-0.5)}$, where c^i is the scaling factor for each contrast level. Note that locally, the contrast is different even within a single frame. Here the increase between contrast levels refers to the contrast difference evaluated at the same spatial position for two different time instances.

C Steady-state responses of DNPs

We derive here the steady state response of DNPs in several different configurations, including (1) temporal DNPs with feedforward normalization, (2) temporal DNPs with feedforward and feedback normalization and (3) spatio-temporal DNPs with global feedback. For each configuration, we then illustrate how the DNP parameters affect their steady-state response curve.

C.1 Steady-state I/O of temporal DNPs with feedforward normalization

We first characterize the steady-state response of temporal DNPs with feedforward normalization. The output of the temporal DNP with feedforward normalization is given by

$$v^i(t)=\frac{\mathcal{T}^1u^i}{\mathcal{T}^2u^i}, \tag{38}$$

and the steady-state response to a constant intensity value I amounts to

$$v^i(I)=\frac{a_0+a_1I+a_2I^2}{c_0+c_1I+c_2I^2}. \tag{39}$$

To ensure that the denominator is always positive, we also assume that the coefficients a_0,a_1,a_2,c_0,c_1,c_2 are positive. Therefore,

$$v^i(0)=\frac{a_0}{c_0}, \tag{40}$$

determines the response level at the lowest possible light intensity level, and

$$v^i(\infty)=\begin{cases}\frac{a_2}{c_2}, & \text{if } c_2\neq 0,\\ \infty, & \text{if } c_2=0, a_2\neq 0,\\ \frac{a_1}{c_1}, & \text{if } c_2=a_2=0, c_1\neq 0,\\ \infty, & \text{if } c_2=a_2=c_1=0, a_1\neq 0,\\ \frac{a_0}{c_0}, & \text{if } c_2=a_2=c_1=a_1=0.\end{cases} \tag{41}$$

Clearly, it is not desirable to have $v^i(\infty)=\infty$. We are interested in the case where $v^i(\infty)>v^i(0)$, *i.e.*, $\frac{a_2}{c_2}>\frac{a_0}{c_0}$ if $c_2\neq 0$ or $\frac{a_1}{c_1}>\frac{a_0}{c_0}$ if $c_2=0$ and $c_1\neq 0$.

Here we assume that $0\leq a_0\ll c_0$, and thus $v^i(0)\approx 0$. We also consider normalizing $v^i(\infty)$ to 1. Therefore, either

$$\frac{a_2}{c_2}=1, \text{ when } a_2\neq 0 \text{ and } c_2\neq 0, \tag{42}$$

or

$$\frac{a_1}{c_1} = 1, \text{ when } c_2 = 0. \tag{43}$$

Let $I' = \log_{10}(I)$, we have

$$v^i(I') = \frac{a_0 + a_1 10^{I'} + a_2(10^{2I'})}{c_0 + c_1 10^{I'} + c_2(10^{2I'})}. \tag{44}$$

If $a_2 = c_2 = 0$, then

$$\begin{aligned} v^i(I') &= \frac{a_0 + a_1 10^{I'}}{c_0 + c_1 10^{I'}} \\ &= \frac{a_0}{c_0 + c_1 10^{I'}} + \frac{1}{\frac{c_0}{a_1} 10^{-I'} + \frac{c_1}{a_1}} \\ &\approx \frac{1}{\frac{c_0}{a_1} 10^{-I'} + \frac{c_1}{a_1}} \end{aligned} \tag{45}$$

is a sigmoidal function.

Otherwise, we have

$$\frac{\mathrm{d}v^i(I')}{\mathrm{d}I'} = \frac{(a_1c_0 - a_0c_1)e^{I'} + 2(a_2c_0 - a_0c_2)e^{2I'} + (a_2c_1 - a_1c_2)e^{3I'}}{(c_0 + c_1e^{I'} + c_2(e^{2I'}))^2}. \tag{46}$$

Substituting $a_2 = c_2$ and $a_0 \ll c_0$,

$$\frac{\mathrm{d}v^i(I')}{\mathrm{d}I'} = \frac{a_1c_0 10^{I'} + 2c_2c_0 10^{2I'} + c_2(c_1 - a_1)10^{3I'}}{(c_0 + c_1 10^{I'} + c_2(10^{2I'}))^2}. \tag{47}$$

Figure 15 shows two qualitatively different steady-state response resulted from two sets of parameters. To guarantee $\frac{\mathrm{d}v^i(I')}{\mathrm{d}I'} > 0, \forall I'$, *i.e.*, ensuring $v^i(I')$ is monotonically increasing, we must have $c_1 \geq a_1$. In Fig. 15A, the "overshoot" in the mid-range light intensity levels is due to $a_1 > c_1$. In contrast, when $a_1 \leq c_1$, the steady-state response is sigmoidal.

C.2 Steady-state I/O of temporal DNPs with feedforward and feedback normalization

We now characterize the steady-state response of temporal DNPs with feedforward and feedback normalization. With both feedforward and local-feedback normalization, the output of the DNP can be expressed as

$$v^i(t) = \frac{\mathcal{T}^1 u^i}{\mathcal{T}^2 u^i + \mathcal{T}^3 v^i}, \tag{48}$$

and the steady-state response to a constant intensity value I is given by

$$v^i(I) = \frac{a_0 + a_1 I + a_2 I^2}{c_0 + c_1 I + c_2 I^2 + d_1 v^i + d_2 (v^i)^2}. \tag{49}$$

Therefore, $v^i(I)$ is the real, positive root of the cubic equation

$$\begin{aligned} &d_2(v^i)^3 + d_1(v^i)^2 + (c_0 + c_1 I + c_2 I^2)v^i \\ &\quad -(a_0 + a_1 I + a_2 I^2) = 0. \end{aligned} \tag{50}$$

By denoting $A = -(a_0 + a_1 I + a_2 I^2)$ and $C = (c_0 + c_1 I + c_2 I^2)$, we have

$$d_2(v^i)^3 + d_1(v^i)^2 + Cv^i + A = 0. \tag{51}$$

If $d_2 = 0$, then

$$v^i = \frac{-C \pm \sqrt{C^2 - 4d_1A}}{2d_1}. \tag{52}$$

For v^i to be real and non-negative, the coefficients must satisfy $C^2 \geq 4d_1A$, and

$$v^i = \frac{\sqrt{C^2 - 4d_1A} - C}{2d_1}. \tag{53}$$

If $d_2 \neq 0$, then the discriminant

$$\Delta = \frac{4\Delta_0^3 - \Delta_1^2}{27d_2^2}, \tag{54}$$

where

$$\Delta_0 = d_1^2 - 3d_2C, \tag{55}$$

$$\Delta_1 = 2d_1^3 - 9d_2d_1C + 27d_2^2A. \tag{56}$$

In addition, we define

$$D = \sqrt[3]{\frac{\Delta_1 + \sqrt{-\Delta}}{2}}. \tag{57}$$

If $\Delta > 0$ then the cubic equation has 3 real roots. In this case, since $A < 0$ by definition, and there must exists a positive root with value between 0 and 1, we must have the other 2 roots negative. Therefore, the steady-state solution must be the positive root that amounts to

$$v^i = -\frac{1}{3d_2}\left(d_1 + D\frac{-1 + \sqrt{-3}}{2} + \frac{\Delta_0}{D\frac{-1+\sqrt{-3}}{2}}\right). \tag{58}$$

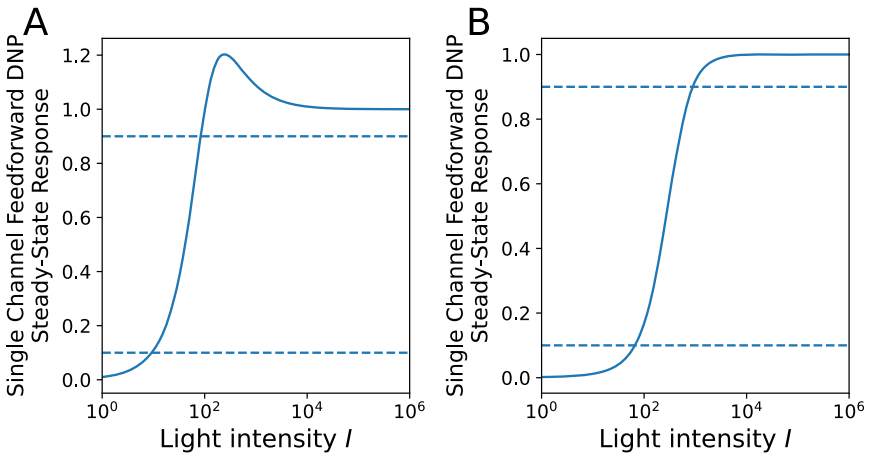

Fig. 15 Steady-state responses of feedforward DNPs exhibit two different characteristics. **A** The steady-state response of a feedforward DNP with "overshoot" response. The following parameters were used: $a_0 = 1, a_1 = 1,000, a_2 = 10, c_0 = 100,000, c_1 = 10, c_2 = 10$. **B** Monotonically increasing steady-state response of a feedforward DNP. The following parameters were used: $a_0 = 10, a_1 = 10, a_2 = 0.1, c_0 = 100,000, c_1 = 10, c_2 = 0.1$

If $\Delta \leq 0$ then the cubic equation has only one real root, and the steady-state response amounts to

$$v^i = -\frac{1}{3d_2}\left(d_1 + D + \frac{\Delta_0}{D}\right). \tag{59}$$

To sum up, the steady-state response of a temporal DNP with feedforward and feedback normalization is given by

$$v^i(I) = \begin{cases} \frac{-C+\sqrt{C^2-4d_1A}}{2d_1} & \text{if } d_2 = 0, \\ -\frac{1}{3d_2}\left(d_1 + D\frac{-1+\sqrt{3}j}{2} + \frac{\Delta_0}{D\frac{-1+\sqrt{3}j}{2}}\right) & \text{if } d_2 \neq 0, \Delta > 0, \\ -\frac{1}{3d_2}\left(d_1 + D + \frac{\Delta_0}{D}\right) & \text{if } d_2 \neq 0, \Delta \leq 0 \end{cases} \tag{60}$$

where

$$A = -(a_0 + a_1 I + a_2 I^2), \tag{61}$$

$$C = (c_0 + c_1 I + c_2 I^2), \tag{62}$$

$$D = \sqrt[3]{\frac{\Delta_1 + \sqrt{-\Delta}}{2}}, \tag{63}$$

$$\Delta_0 = d_1^2 - 3d_2 C, \tag{64}$$

$$\Delta_1 = 2d_1^3 - 9d_2 d_1 C + 27 d_2^2 A. \tag{65}$$

In Fig. 16, we compare the steady-state responses of the single-channel self-feedback DNP with different d_1 and d_2 values.

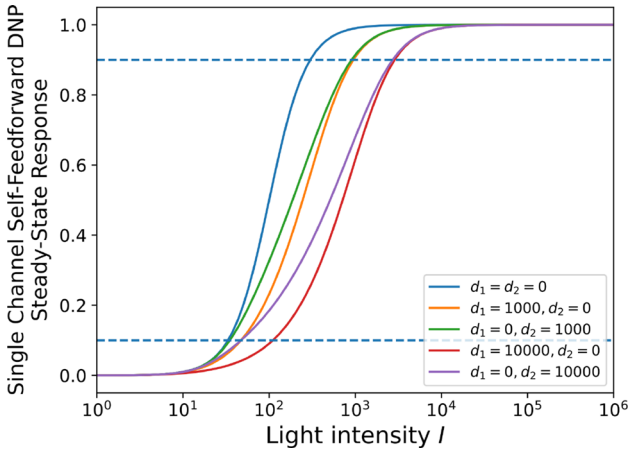

Fig. 16 Effect of d_1 and d_2 on the steady-state response of single-channel feedback DNP. The DNP is constructed with the following parameters: $a_0 = 0, a_1 = 0, a_2 = 0.01, c_0 = 100, c_1 = 0, c_2 = 0.01$

C.3 Steady-state I/O of spatio-temporal DNPs with global feedback

Finally, we characterize the steady-state response of spatio-temporal DNPs with global feedback normalization. The response of a spatio-temporal DNP can be expressed as

$$v^i(t) = \frac{\mathcal{T}^1 u^i}{\mathcal{T}^2 u^i + \mathcal{T}^3 v^i + \mathcal{L}^4 \mathbf{v}}, \tag{66}$$

for $i = 1, 2, \ldots, N$. The steady-state response is given by the system of equations

$$v^i(\mathbf{I}) = \frac{a_0 + a_1 I^i + a_2 (I^i)^2}{c_0 + c_1 I^i + c_2 (I^i)^2 + d_1 v^i + d_2 (v^i)^2 + \sum_{k=1}^N g_1^k v^k + \sum_{k=1}^N \sum_{l=1}^N g_2^{kl} v^k v^l}, \tag{67}$$

for $i = 1, 2, \ldots, N$, where $\mathbf{I} = \left[I^1, I^2, \ldots, I^N\right]$ is a vector of constant intensity values that are presented to each photoreceptor i, and

$$g_1^k = \int_{\mathbb{R}} h^{k4}(t)dt, k = 1, 2, \ldots, N, \tag{68}$$

$$g_2^{kl} = \int_{\mathbb{R}^2} h^{kl4}(t_1, t_2)dt_1 dt_2, k, l = 1, 2, \ldots, N. \tag{69}$$

After rearranging terms, the steady-state response of the spatio-temporal DNP should satisfy the following equations

$$\begin{aligned} &d_2(v^i)^3 + \sum_{k,l} g_2^{kl} v^k v^l v^i + d_1(v^i)^2 + \sum_{k=1}^{N} g_1^k v^k v^i \\ &+ \left(c_0 + c_1 I^i + c_2(I^i)^2\right) v^i - \left(a_0 + a_1 I^i + a_2(I^i)^2\right) = 0, \end{aligned} \tag{70}$$

for $i = 1, 2, \ldots, N$. The zeros can be found by using Powell's hybrid method (Powell 1970) efficiently.

References

Adelson EH, Bergen JR (1985) Spatiotemporal energy models for the perception of motion. J Opt Soc Am 2(2):284

Baker S, Scharstein D, Lewis JP et al (2011) A database and evaluation methodology for optical flow. Int J Comput Vis 92:1–31

Barlow HB, Levick WR (1965) The mechanism of directionally selective units in rabbit's retina. J Physiol 178(3):477–504

Beaudoin D, Gorghuis BG, Demb JB (2007) Cellular basis for contrast gain control over the receptive field center of mammalian retinal ganglion cells. J Neurosci 27(10):2636–2645

Borst A (2014) Fly visual course control: behaviour, algorithms and circuits. Nat Rev Neurosci 15(9):590–599. https://doi.org/10.1038/nrn3799

Borst A, Haag J, Mauss AS (2020) How fly neurons compute the direction of visual motion. J Comp Physiol A 206(2):109–124. https://doi.org/10.1007/s00359-019-01375-9

Burge J, Geisler WS (2015) Optimal speed estimation in natural image movies predicts human performance. Nat Commun 6(1):7900. https://doi.org/10.1038/ncomms8900

Butler DJ, Wulff J, Stanley GB, et al (2012) A naturalistic open source movie for optical flow evaluation. In: A. Fitzgibbon et al. (Eds) European Conf. on Computer Vision (ECCV). Springer-Verlag, Part IV, LNCS 7577, pp 611–625

Carandini M, Heeger DJ (1994) Summation and division by neurons in primary visual cortex. Science 264:1333–1336

Carandini M, Heeger D (2012) Normalization as a canonical neural computation. Nat Rev Neurosci 13:51–62

Cronin TW, Johnsen S, Marshall NJ et al (2014) Visual ecology. Princeton University Press, Princeton

Dosovitskiy A, Fischer P, Ilg E, et al (2015) Flownet: learning optical flow with convolutional networks. In: IEEE International Conference on Computer Vision (ICCV), pp 2758–2766. http://lmb.informatik.uni-freiburg.de/Publications/2015/DFIB15

Eichner H, Joesch M, Schnell B et al (2011) Internal structure of the fly elementary motion detector. Neuron 70(6):1155–1164. https://doi.org/10.1016/j.neuron.2011.03.028

Geiger A, Lenz P, Urtasun R (2012) Are we ready for autonomous driving? the kitti vision benchmark suite. In: 2012 IEEE Conference on Computer Vision and Pattern Recognition, pp 3354–3361, https://doi.org/10.1109/CVPR.2012.6248074

Goodfellow I, Bengio Y, Courville A (2016) Deep Learning. MIT Press, Cambridge

Goyette N, Jodoin PM, Porikli F, et al (2012) Changedetection.net: A new change detection benchmark dataset. In: 2012 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, pp 1–8, https://doi.org/10.1109/CVPRW.2012.6238919

Grzywacz NM, Yuille AL, Barlow HB (1990) A model for the estimate of local image velocity by cells in the visual cortex. Proc R Soc London B Biol Sci 239(1295):129–161. https://doi.org/10.1098/rspb.1990.0012

Hassenstein B, Reichardt W (1956) Systemtheoretische analyse der zeit-, reihenfolgen- und vorzeichenauswertung bei der bewegungsperzeption des rüsselkäfers chlorophanus. Z Naturforsch B 11(9):513–524

Ioffe S, Szegedy C (2015) Batch normalization: accelerating deep network training by reducing internal covariate shift. In: International conference on machine learning. PMLR, pp 448–456

Lazar AA, Ukani NH, Zhou Y (2016) A motion detection algorithm using local phase information. Comput Intell Neurosci 7915:245

Lazar AA, Ukani NH, Zhou Y (2020) Sparse identification of contrast gain control in the fruit fly photoreceptor and amacrine cell layer. J Math Neurosci 10(1):1–35. https://doi.org/10.1186/s13408-020-0080-5

Lazar AA, Liu T, Zhou Y (2022) Divisive normalization circuits faithfully represent auditory and visual sensory stimuli. bioRxiv. https://doi.org/10.1101/2022.09.17.506431

Lazar AA, Turkcan MK, Zhou Y (2022) A programmable ontology encompassing the functional logic of the drosophila brain. Front Neuroinf. https://doi.org/10.3389/fninf.2022.853098

Lazar AA, Liu T, Yeh CH (2023) The functional logic of odor information processing in the drosophila antennal lobe. PLoS Comput Biol 19(4):1–33. https://doi.org/10.1371/journal.pcbi.1011043

Li R, Tan RT, Cheong LF (2018) Robust optical flow in rainy scenes. ECCV pp 288–304

Lyu S, Simoncelli EP (2008) Nonlinear image representationi using divisive normalization. In: IEEE Conference on Computer Vision and Pattern Recognition, pp 1–8

Mathis A, Nothwang W, Donavanik D et al (2016) Making optic flow robust to dynamic lighting conditions for real-time operation. Tech. rep, US Army Research Laboratory

Mayer N, Ilg E, Häusser P, et al (2016) A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In: IEEE International Conference on Computer Vision and Pattern Recognition (CVPR), pp 4040–4048. http://lmb.informatik.uni-freiburg.de/Publications/2016/MIFDB16, arXiv:1512.02134

Menze M, Heipke C, Geiger A (2018) Object scene flow. ISPRS J Photogramm Remote Sens 140:60

Michelson A (1927) Studies in optics. University of Chicago Press, Chicago

Ohshiro T, Angelaki DE, DeAngelis GC (2017) A neural signature of divisive normalization at the level of multisensory integration in primate cortex. Neuron 95(2):399–411

Olsen SR, Bhandawat V, Wilson RI (2010) Divisive normalization in olfactory population codes. Neuron 66(2):287–299

Orchard G, Etienne-Cummings R (2014) Bioinspired visual motion estimation. Proc IEEE 102(10):1520–1536. https://doi.org/10.1109/JPROC.2014.2346763

Powell M (1970) A hybrid method for nonlinear equations. In: Rabinowitz P (ed) Numerical methods for nonlinear algebraic equations. Gordon and Breach, London, pp 87–114

Rabinowitz NC, Willmore BDB, Schnupp JWH et al (2011) Contrast gain control in auditory cortex. Neuron 70:1178–1191

Sanes JR, Zipursky SL (2010) Design principles of insect and vertebrate visual systems. Neuron 66:15–36

Shah STH, Xuezhi X (2021) Traditional and modern strategies for optical flow: an investigation. SN Appl Sci 3(3):289. https://doi.org/10.1007/s42452-021-04227-x

Shi C, Luo G (2018) A compact VLSI system for bio-inspired visual motion estimation. IEEE Trans Circuits Syst Video Technol 28(4):1021–1036. https://doi.org/10.1109/TCSVT.2016.2630848

Simoncelli EP, Heeger DJ (1998) A model of neuronal responses in visual area MT. Vis Res 38(5):743–761. https://doi.org/10.1016/S0042-6989(97)00183-1

Song Z, Postma M, Billings S et al (2012) Stochastic, adaptive sampling of information by microvilli in fly photoreceptors. Curr Biol 22:1371–1380

Sterling P, Laughlin S (2015) Principles of neural design. MIT Press, Cambridge

Stocker AA (2006) Analog integrated 2-d optical flow sensor. Analog Integr Circ Sig Process 46(2):121–138. https://doi.org/10.1007/s10470-005-0439-2

Sy Takemura, Bharioke A, Lu Z et al (2013) Avisual motion detection circuit suggested by Drosophila connectomics. Nature 500:175–181

Teed Z, Deng J (2020) Raft: recurrent all-pairs field transforms for optical flow. In: Vedaldi A, Bischof H, Brox T et al (eds) Computer Vision - ECCV 2020. Springer, Cham, pp 402–419

van Hateren J (1997) Processing of natural time series of intensities by the visual system of the blowfly. Vis Res 37(23):3407–3416. https://doi.org/10.1016/S0042-6989(97)00105-3

van Santen JPH, Sperling G (1985) Elaborated Reichardt detectors. J Opt Soc Am A Opt Image Sci 2(2):300-321

Wulff J, Sevilla-Lara L, Black MJ (2017) Optical flow in mostly rigid scenes. In: IEEE Conf. on Computer Vision and Pattern Recognition (CVPR), pp 6911–6920. https://doi.org/10.1109/CVPR.2017.731

Yang HH, Clandinin TR (2018) Elementary motion detection in drosophila: algorithms and mechanisms. Annu Rev Vis Sci 4:143–63