

Unsupervised learning in second-order neural networks for motion analysis

Tomaś Maul^{a,*}, Sapiyan Baba^b

^a School of Computer Science, The University of Nottingham Malaysia Campus, Malaysia

^b Faculty of Computer Science & IT, University Malaya, Malaysia

ARTICLE INFO

Article history:

Received 15 September 2009

Received in revised form

14 July 2010

Accepted 9 September 2010

Communicated by I. Bojak

Available online 21 December 2010

Keywords:

Second-order neural networks

Motion analysis

Unsupervised learning

Dendritic computation

Feature correspondences

ABSTRACT

This paper demonstrates how unsupervised learning based on Hebb-like mechanisms is sufficient for training second-order neural networks to perform different types of motion analysis. The paper studies the convergence properties of the network in several conditions, including different levels of noise and motion coherence and different network configurations. We demonstrate the effectiveness of a novel variability dependent learning mechanism, which allows the network to learn under conditions of large feature similarity thresholds, which is crucial for noise robustness. The paper demonstrates the particular relevance of second-order neural networks and therefore correlation based approaches as contributing mechanisms for directional selectivity in the retina.

© 2010 Elsevier B.V. All rights reserved.

1. Introduction

The general motivation underlying this paper consists of revealing how biological neural systems implement motion analysis and the specific aim consists of proposing a model of the computational mechanisms that may underlie directional selectivity (DS) in the vertebrate retina. The general problem of motion analysis is here defined as the estimation of the global transformation taking place between two patterns. These two patterns are assumed to be manifestations of the same physical entity, albeit under different configurations due to the application of some transformation. Because we are talking about motion, which unfolds through time, the two patterns should consist of retinal images from two consecutive time instants. Typically, early (or low-level) motion analysis is only concerned with speed and direction parameters, which essentially comprise the group of translation transformations. In this paper, apart from translation, we will also briefly consider rotation.

The cellular and thus computational mechanisms underlying motion analysis are arguably better understood at the level of the retina [10]. This is due mainly to the retina's ease of access and its long history of research. Some of the earliest recordings of the DS properties of retinal cells [2] revealed two main types of cells: (1) ON-OFF direction-selective retinal ganglion cells (DSGC) and (2) ON DSGC. Because of the low encounter rate of the latter type,

more is known about ON-OFF DSGC [1,38,9]. These cells can detect the motion of bright or dark objects, are robust to a broad range of speeds and are divided into four subtypes according to which direction they are sensitive to [2,37].

In spite of the 45 years that have passed since these early discoveries and the numerous experiments that have followed, the mechanisms underlying the response properties of these cells are still not entirely understood. Several early models were built on a correlation based approach [12] wherein motion is computed from non-linear interactions between two input channels with slightly different delays. In [2] the non-linearity consists of a summation followed by a threshold operation. In other cases [41,42,4] the non-linearity consists of a multiplicative function.

Another simple mechanism [40] relies on the passive and delayed conductance of signals along dendritic cables. When stimuli move towards the soma, signals accumulate strongly, whereas when they move away from the soma, the outer (later) signals will not be added on time to the inner (earlier) signals because of propagation delays. Although this mechanism can implement direction selectivity, the difference of accumulated signals (towards/away from the soma) is not strong enough for this to represent a robust DS mechanism [30].

Recent models are the result of a long string of experimental and modeling studies of which we will highlight a few key examples. In [24] it was proposed that directional selectivity results from an asymmetric pattern of excitatory and inhibitory inputs in the dendritic trees of ganglion cells, whereby movement in the preferred (PREF) direction results in excitation before inhibition, leading to cell firing, and movement in the opposite (NULL) direction results in inhibition before excitation and therefore the

* Corresponding author.

E-mail addresses: Tomas.Maul@nottingham.edu.my (T. Maul), pian@um.edu.my (S. Baba).

URL: <http://baggins.nottingham.edu.my/~kcztm/> (T. Maul).

former vetoing the latter. The model in [3] proposed the same mechanism as [24], but changed its locus of operation from retinal ganglion cells (RGCs) to Starburst Amacrine Cells (SACs). The existence of directional selectivity in signals presynaptic to RGCs, particularly in the dendrites of SACs, was confirmed in [15], via Ca^{2+} imaging techniques.

Starburst Amacrine Cells (SACs) play a fundamental role in the retina's ability to discriminate motion in different directions [15] (see [17] for a short review). Each cell is divided into 6–10 pie slices, each one electrotonically semi-independent from the other, and each one of which can discriminate between centripetal (inward) or centrifugal (outward) motion. The cell responds more strongly to centrifugal motion. How do SACs achieve this sensitivity? The general idea is based on overlapping SAC dendritic arbors and the fact that neighboring SACs inhibit each other [27]. In an inward moving stimulus relative to (A), a neighbor is excited first (B), producing inhibition (from B to A), which then does not allow depolarization when the stimulus reaches the center of A. In the reverse outward motion case, excitation happens first, which counteracts the subsequent inhibition.

The different pie-slices of a SAC connect to different retinal ganglion cells (RGCs), thus allowing ganglion cells to represent different directions of motion [18]. RGCs represent motion in different directions, with a relative amount of robustness to object contrast (e.g. black on white or white on black) or shape. Although some controversy still remains as to the exact mechanisms through which this directional selectivity is obtained, it is generally agreed that the specific connectivity between SACs and RGCs is central to this functionality.

Examples of open question include: (1) since inhibitory mechanisms are essential to DS in the retina why is directional selectivity not entirely eliminated by GABA_A blockers [15], (2) how do SAC and bipolar cell (BC) inputs interact in DS ganglion cells, (3) what mechanisms underlie DS in ON DSGC and (4) what is the role of dendritic spikes [35] in retinal directional selectivity.

These open questions, and in particular the partial effect of GABA_A blockers, suggest that other mechanisms might be required to fully explain retinal DS. Thus a correlation based approach might still be relevant. In [25,26], it is suggested that because of developmental constraints and environmental variability, the retina may implement DS through more than one mechanism. One possible hypothesis states that both excitatory and inhibitory correlational mechanisms are used by the retina, the former mostly via SACs and their connections with RGCs, the latter mostly via connections from BCs to RGCs [18]. In [43] it was shown that directional selectivity in rabbit ON-OFF DSGC is implemented via three mechanisms: (1) a presynaptic signal producing more excitation in favour of the PREF direction, (2) a presynaptic signal producing more inhibition in favour of the NULL direction and (3) post-synaptic non-linear interactions between excitatory and inhibitory signals.

Interestingly, the basic idea that underlies our correlation-based model is also a common approach in Computer Vision (i.e.: feature correspondences) and has strong parallels with the "Reichardt detector" [41]. A correspondence is generally defined as a vector between two local features which represent the same physical element [6]. Features span the 2D space of an image (e.g. retinal image) and may be represented by intensity, colour, or other more complex characteristics. Feature matching may be implemented by simple similarity measures such as the Euclidean distance. In motion analysis, one feature of a correspondence belongs to an image (or frame) at time t , while the other belongs to an image (or frame) at time $t-1$. The basic idea is that each correspondence is consistent with one or more transformations, which may represent the motion being applied to the pattern in the image. If we look at the whole distribution of correspondences and allow each one to vote for one or more consistent transformations, then the

transformation with the most votes, is likely to correspond to the global motion observed. This vote based approach is highly reminiscent of the Hough transform [22]. From this point onwards, our proposed approach will be referred to as correspondence distribution analysis (CDA).

One of the elegant aspects of CDA is that it can be very easily represented by a second-order neural network (SONN), which is closely related to the Sigma-Pi network (SPN). Briefly put, SONNs are a specialization of higher-order neural network (HONN), essentially characterized by a first layer of second-order connections and a second layer of summation units (refer to [11] for an early reference on SPN). Recall that the two main aspects of CDA are: (1) the detection of feature correspondences and (2) the voting for transformations. The relevance of SONNs lies in the fact that the first aspect can be easily implemented by the layer of higher-order units, whereas the second aspect can be easily implemented by the summation layer. Because we are interested only in correspondences between two matching points, second-order nodes suffice for our purposes. Refer to Fig. 1 for an illustration of a SONN that can solve a very simple one-dimensional translation problem. The "source map" represents an image at time $t-1$, while the "target map" refers to an image at time t . Each map unit is binary and represents the presence or absence of a feature at a particular location. The second-order product nodes (middle layer) represent feature correspondences and the vote nodes represent transformations, e.g.: the " -1 " node represents a single pixel shift to the left. As an example of the logic carried out by the network notice how the second pixel of the left image is connected to the first pixel of the right image at the third product node (counting from the left), which in turn is connected to the " -1 " vote node.

Other work that involves using SPNs (or other HONNs) for solving visual problems related to the estimation of transformations includes [44,36]. SONNs remain a fertile area of research, partly because of their usefulness in classification and regression and partly due to several unresolved questions such as how to reduce their combinatorial cost and how to increase their convergence speed. Although the current paper does not offer solutions to these open problems, it demonstrates how SONNs can be applied to the problem of motion analysis through basic correlative learning, elucidates the biological relevance of the model and proposes a new variability dependent learning mechanism which makes the algorithm more robust to similarity thresholds and noise.

So where does the biological plausibility of the model lie? In other words, can biological neural systems embody CDA as implemented by SONNs? In [39] it was shown how logical AND operations (via sigmoidal integration) can be computed at a local dendritic level. In brief, two active inputs close enough in space and time produce a signal that is

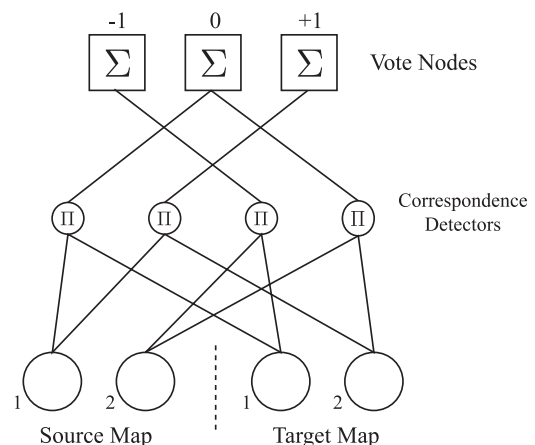


Fig. 1. An artificial neural architecture for CDA.

greater than the sum of the inputs. If only a single input is active then the dendritic signal is not amplified in the same manner. Although our correspondence detection function (see Eq. (3)) is not a simple AND operation, we can envision decomposing our feature space into several binary detectors. In other words, instead of each coordinate being represented by one node capable of expressing a range of values, each coordinate can be associated with several nodes, each one capable of representing the presence or absence of a feature. In this scenario, the dendritic conjunctions of [39] become very useful. A related scenario would involve graded input from BCs, whereby a correspondence would be detected for pairs of high-contrast regions.

The ability to compute synaptic products at the level of dendrites, together with the traditional summation function of neurons [31], means that an individual retinal ganglion cell can represent both SONN layers with respect to a single sigma node. Thus, each RGC can represent a different transformation, by receiving inputs from two different images (from consecutive time instants), implementing correspondence detection, and summing the results. Fig. 2 depicts the biological counterpart of Fig. 1. Pairs of synapses represent the layer of second-order products the results of which are summed via passive dendritic integration. The image at time t_i could originate from ON BCs, whereas the image from time t_{i-1} could originate from OFF BC rebound signals. Alternatively the image from time t_{i-1} could originate from a delayed inhibitory signal. Since BC responses are contrast based and have large receptive fields, they contribute to the noise robustness of the solution and attenuate the combinatorial cost resulting from second order relationships.

As was argued in [34], one of the drawbacks of solutions that require local dendritic computations, is that they raise the difficult question of how synapses are specifically grouped in local regions of the dendritic tree. The specificity of these groupings seems to call for prohibitively complex developmental mechanisms. Thus, in order to strengthen the biological plausibility of our model, we demonstrate how simple Hebbian mechanisms [21] are sufficient for training SONNs, such that their connection weights reflect the proper association between correspondences and consistent transformations. We assume that all possible pairings are present in the dendritic tree with a subsequent refinement of their strengths. Pairs whose strengths drop below a certain level can be pruned out of the final dendritic architecture. In [44] it was shown how learning rules derived from self-organizing maps (SOMs) were appropriate for training SPNs for learning coordinate transformations. In previous work [33] we implemented a similar correlative learning approach to a SONN where each sigma node was allowed to connect to only a subset of all possible correspondences. In the current work, each sigma node connects to all possible correspondences.

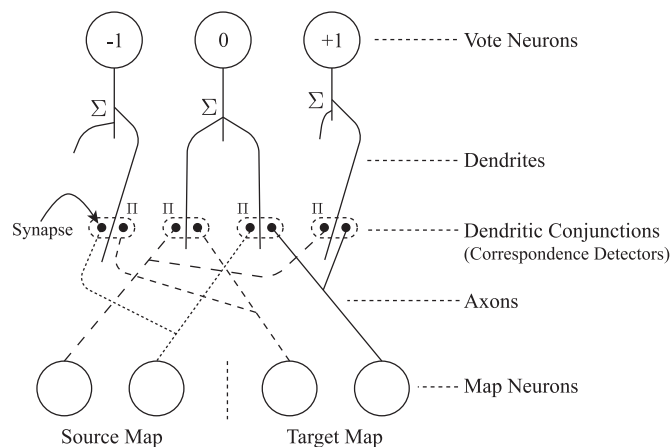


Fig. 2. A biological neural architecture for CDA.

The long term hypothesis of this work is that RGCs implement DS through a correspondence-based mechanism (most probably in association with inhibition-mediated SAC mechanisms). From a modelling perspective, this hypothesis cannot be confirmed without a large-scale detailed model of the retina, which falls outside the scope of the current paper. For now, we focus on the plausibility of the solution and on useful learning/developmental mechanisms with some of their properties.

The main contributions of this paper can be summarized into five points. First, to the best of our knowledge, this is the first time unsupervised learning techniques have been applied to SONN for the purpose of motion analysis. Second, arguments are proposed for the biological realization of the solution. Third, several experiments are discussed which reveal how the solution behaves under different environmental and network conditions (e.g. different noise levels). Fourth, the paper introduces *variability dependent learning* in order to cope with large feature similarity thresholds. Finally, the paper provides simulation modelling evidence for the fact that direction selectivity in the retina may at least be partially implemented via correlation based computations.

The next section deals with the simulation environment, SONN architecture, learning algorithm and testing procedures in more detail. The third section presents the results of the experiments and the fourth section discusses those results, draws several conclusions and suggests various directions for further work.

2. Methods

2.1. Environment

For training and testing purposes, two types of data (or environment) were used: one semi-naturalistic and another entirely synthetic. In the first case, the *environment* consisted of two elements: (1) a static greyscale natural image with resolution 100×100 and (2) a dynamic window. The dynamic window, which can be of different sizes, was allowed to move around the image (refer to Fig. 3), continuously conveying the underlying information to the developing SONN. By controlling the motion of the window, we control the type of motion analysis problem that the SONN is confronted with. In the second case, the *environment* consisted of an artificial stimulus consisting of a random array of pixels, each

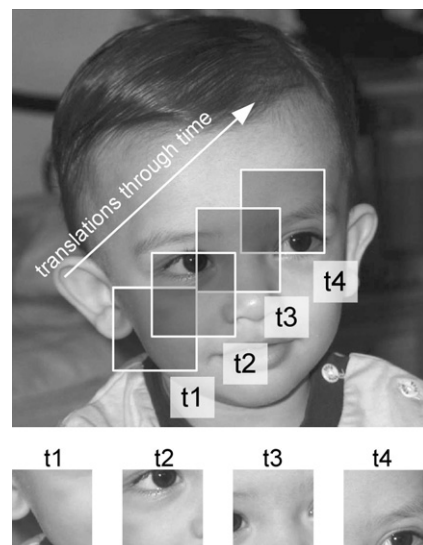


Fig. 3. Window translations on a natural image.

one of which was characterized by either coherent or incoherent motion. At a high-level, both environments function in a similar way: refer to Algorithm 1. Essentially, at each iteration of the learning process there are two images, one from time t and the other from time $t - 1$, to which are added noise, and which are then observed by the SONN. From this point onwards, we will refer to the two environments as *window* and *coherence* environments, respectively.

Algorithm 1. General learning environment.

```

1:   procedure ENVIRONMENT ( $p$ )   ▷  $p$ : param.
2:    $S \leftarrow \text{genSONN}(p)$    ▷ Generate a SONN
3:    $I_{old} \leftarrow \text{getNewImg}(p)$    ▷ Get new image
4:    $I_{old} \leftarrow \text{addNoise}(I_{old}, p)$    ▷ Add noise
5:    $\text{doLearn} \leftarrow \text{true}$    ▷ Stop condition
6:   while  $\text{doLearn}$  do   ▷ Learning loop
7:      $I_{new} \leftarrow \text{getNewImg}(p)$    ▷ New img.
8:      $I_{new} \leftarrow \text{addNoise}(I_{new}, p)$    ▷ Add noise
9:      $S \leftarrow \text{update}(S, I_{old}, I_{new}, p)$    ▷ Learn
10:     $I_{old} \leftarrow I_{new}$    ▷ New img. becomes old
11:     $\text{doLearn} \leftarrow \text{checkStopCond}(p)$ 
12:  end while
13:  end procedure

```

The *addNoise* function is controlled by two main parameters: (1) proportion of affected pixels and (2) maximum noise magnitude. First, a set of random pixels from the input image is selected, where the size of the set is determined by the first parameter. Then a random intensity (positive or negative) within the greyscale range (i.e. 256 values), where the largest magnitude is controlled by the second parameter, is added to each pixel from this affected set.

Probably the main aspect of the *window environment* pertains to how window motion is controlled. Currently, we allow motion to be dictated by three free parameters, i.e. horizontal (T_x) and vertical (T_y) translation and rotation (θ). For example, if the motion (or transformation) vector (i.e. $[T_y, T_x, \theta]$) is defined by $[0, -1, \pi]$ this means that the window is traveling from right to left whilst rotating π radians at every iteration. Motion is not constant, therefore the motion vector changes every z iterations, also a free parameter. The environment allows for two different types of change. In the first one, random motion vectors can be selected from the whole space of transformations, while in the second one, new motion vectors, which can be described as *nudges*, can be chosen from a subset of transformation space. Consider a T_x space ranging from -3 to $+3$. If the current T_x is 2, then the new T_x can be nudged to either 1 or 3, but not the other values. Thus, in this mode, each motion vector can only be nudged to adjacent positions in transformation space. If we assume an image of resolution 100×100 , a window of size 3×3 and single-pixel shifts, then we have a total of 85,240 possible pairs of windows. Since both training and testing are based on stochastic transformations (i.e. stochastically chosen subsets of this large set), we effectively have different training and testing sets each time we run our simulation.

The *coherence environment* generates stimuli where one set of points moves across the image with consistent (i.e. coherent) motion and another set moves randomly (i.e. incoherently). The proportion of moving points (relative to the total number of pixels comprising the stimulus) and the proportions of coherent vs. incoherent points are free-parameters. The transformation space of both coherent and incoherent sets are also flexible. As with the *window environment*, it is possible for the experimenter to control how frequently coherent motion changes. The range of greyscale intensities adopted by points is also controllable.

2.2. Second-order neural networks

The general mathematical definition of the output of a HONN unit [20,19] is encapsulated in the following expression:

$$y = \gamma \left(w_0 + \sum_j w_j x_j + \sum_{j,k(j \leq k)} w_{jk} x_j x_k + \sum_{j,k,l(j \leq k \leq l)} w_{jkl} x_j x_k x_l + \dots \right) \quad (1)$$

where y represents the output of a node from the last layer, j, k and l represent image coordinates, w_{jk} represents the weight of the connection originating from the jk second-order node, x_j represents the value of x at pixel coordinate j and γ usually represents some squashing function (e.g. sigmoid).

The output of our SONN, which is a specific variation of Eq. (1), is defined by the following expression:

$$y = \sum_{j,k} w_{jk} \sigma(x_j, z_k) \quad (2)$$

where y, j, k, w_{jk} are defined as before, x and z represent two different images and the function σ is defined as follows:

$$\sigma(x_j, z_k) = \begin{cases} 1 & \text{if } |x_j - z_k| \leq T \\ 0 & \text{if } |x_j - z_k| > T \end{cases} \quad (3)$$

where T represents a feature similarity threshold. Since the image features used in our experiments consisted of greyscale values, thresholds were limited by the range $[0, 255]$.

As can be seen from Eqs. (2) and (3), and because our model deals with correspondences between two greyscale images, it adopts the following main specializations in relation to general HONNs:

1. Higher-order nodes are exclusively second-order.
2. The second-order node function is a thresholded difference.
3. Inputs into second-order nodes originate from different images.

Fig. 4 illustrates the general architecture of our SONN. Each summation node (i.e. y) is connected to all possible correspondences (i.e. σ units) which in turn compute thresholded differences between features from different images as in Eq. (3) (this is slightly different from the network in Fig. 1 where correspondences represent products between binary values). From this point onwards, summation nodes will also be referred to as vote nodes, seeing that it is at these units that transformations are voted for.

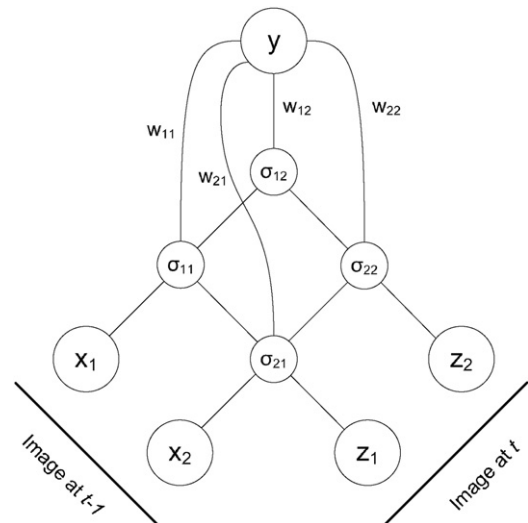


Fig. 4. Diagrammatic depiction of Eq. (2).

The model is closely related to SPNs (as implemented, for example, in [44]), the main difference being the function used at the higher-order units.

2.3. Learning

Our simple Hebbian-based learning algorithm is outlined in Algorithm 2. Note that p stands for *parameters*. At each iteration, the SONN is updated based on the previous and current images. The algorithm scans all the vote nodes and all possible correspondences and weights. For each vote node, the algorithm checks whether it is a winning node. The winning vote node is defined to be the one with the most activation based on the current patterns of correspondences (i.e. feature matches between images) and weights. How each weight of a vote node is updated depends primarily on whether the vote node is a winner or not, and whether the correspondence (to which the weight refers to) is active or not. These two conditions create four possibilities, each one with a different update rate. The general rate α is applied in all four conditions, whereas β_1 is for a winning vote node and an active correspondence, β_2 is for a winning vote node and an inactive correspondence, β_3 is for a non-winning vote node and an active correspondence and finally β_4 is for a non-winning vote node and an inactive correspondence. When not explicitly mentioned, the learning parameters can be assumed to be $\alpha = 0.1$, $\beta_1 = 1$, $\beta_2 = 1$, $\beta_3 = 0.05$ and $\beta_4 = 0$.

Algorithm 2. Learning algorithm.

```

1:   procedure UPDATESONN( $V, C, p$ )
2:     for all  $v \in V$  do   ▷  $V$ : set of vote nodes
3:       for all  $c \in C$  do   ▷  $C$ : correspondences
4:          $w \leftarrow \text{getWeight}(v, c)$    ▷ Cor. weight
5:         if ( $v \equiv \text{winner}$ ) & ( $c \equiv 1$ ) then
6:            $w \leftarrow w + \alpha \beta_1$    ▷ Increment
7:         els if ( $v \equiv \text{winner}$ ) & ( $c \equiv 0$ ) then
8:            $w \leftarrow w - \alpha \beta_2$    ▷ Decrement
9:         els if ( $v \neq \text{winner}$ ) & ( $c \equiv 1$ ) then
10:           $w \leftarrow w - \alpha \beta_3$    ▷ Decrement
11:         els if ( $v \neq \text{winner}$ ) & ( $c \equiv 0$ ) then
12:           $w \leftarrow w - \alpha \beta_4$    ▷ Decrement
13:         end if
14:       end for
15:     end for
16:   end procedure

```

2.4. Performance measures

We have tested the robustness of our model (i.e. architecture and learning) in different conditions as defined by various parameter settings. Examples of conditions we controlled include: window motion properties, noise, similarity thresholds, size of transformation space, coherence and others. Because of the effect of random variations, mostly due to the stochastic nature of weight initialization and window motion, for each condition we have run 10 separate simulations, the results of which were averaged.

Because we are working within an unsupervised learning framework, the network's outputs (i.e. vote nodes) have no fixed representational format. For example, on some occasions the first output node may represent a right shift, whilst on others it might represent a left shift, or some other transformation. This makes it impossible to compute network accuracy in the classical sense, even if we know input labels. This impossibility led us to create a performance indicator consisting of two measures denoted by

consistency and *completeness*. In broad terms, completeness measures how much of the transformation space is covered by a particular SONN, while consistency measures how consistently each vote node represents its transformation. To exemplify completeness, if the transformation space consists of four transformations but the SONN only responds to three of them, then the completeness of the network is 75%. To exemplify consistency, consider a vote node v_1 that is known to represent transformation T_1 (because the majority of its activations are for T_1). If v_1 is a winning node 10% of the times for transformations other than T_1 , then the consistency of v_1 is 90%. The global consistency measure for the network is calculated by taking the average of all individual vote node consistencies. Wherever the single measure *performance* is mentioned, this should be interpreted as the average between consistency and completeness.

3. Results

Fig. 5 explains one of the correspondence distribution representations adopted in Figs. 6 and 7. This representation unfolds each 2D image into a 1D representation, and represents the whole space of correspondences as a 2D matrix. A low-resolution "object" and its background scene are represented in (a) and (b). In (c) the object is shifted to the right by one pixel. The representation (0,+1) denotes a vertical shift of 0 and a horizontal shift of +1. The specific feature correspondences that can be detected between frames at times t_1 and t_2 are depicted in (d). All possible correspondences consistent with a (0,+1) shift are depicted in (e). Subfigure (f) assigns a specific index to each pixel whereas (g) lists down all correspondences from (e) using the indices of (f). In (h) both frames (from times t_1 and t_2) are unfolded (2D to 1D) and placed at right angles from each other, defining a matrix that represents all possible correspondences between the images. The correspondences from (e) and (g) are represented in the correspondence matrix by darkened squares.

Fig. 6 gives us a general impression of the appearance of vote nodes in terms of their correspondence distributions. For this example the network was trained in the *window environment* where it was exposed to four different translations: static, right, up and diagonal (45°). The figure depicts two different representations of correspondence distributions. The top representation is as explained in Fig. 5. The "x-axis" of the matrix represents indexes of the window image at time $t-1$, whereas the "y-axis" represents indexes at time t . The greyscale intensity of each correspondence (or matrix square) is directly proportional to the magnitude of its weight. In the bottom representation each graph depicts the correspondence geometry (i.e. length and angle) of each vote node. Note that, for clarity purposes, the graphs were manually redrawn from actual simulation results and that only correspondences whose weights were larger than 0.64 were depicted. From left to right, both representations depict static, rightward, upward and diagonal shifts. Using the same representational devices just described, Fig. 7 depicts a set of vote nodes that have been trained on rotational motion.

In order to get a better idea of the convergence properties of the SONN, Fig. 8 depicts the effect of the algorithm's general learning rate α on performance. The network was trained in an environment consisting of nine possible translations, i.e.: $T_x \in \{-1, 0, +1\}$ and $T_y \in \{-1, 0, +1\}$. Note that this transformation space although small from a Computer Vision perspective, conforms well to the tuning spaces of RGCs, where for instance ON-OFF DS ganglion cells are divided into four subtypes based on their sensitivities to four different directions [10]. The window motion was randomly changed every three iterations. As one can see, the network converges relatively rapidly, around iteration 1500, and seems to work best with $\alpha = 0.1$.

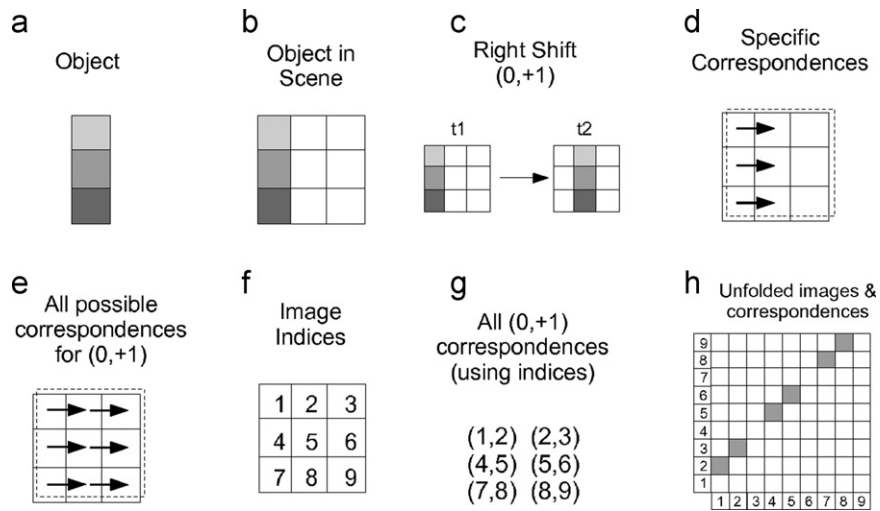


Fig. 5. Representation of correspondences using unfolded images.

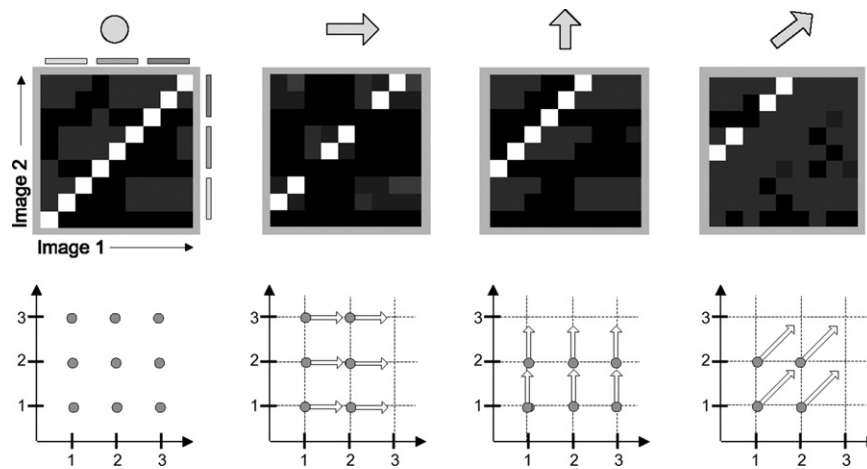


Fig. 6. Correspondence distributions for translational motion.

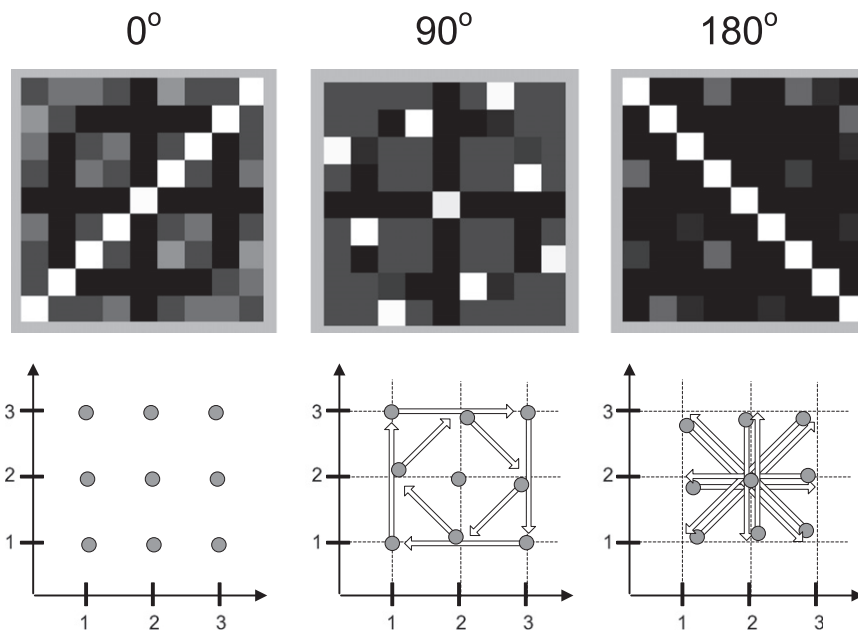


Fig. 7. Correspondence distributions for rotational motion.

Fig. 9 illustrates how our SONN seems to be relatively scalable. All experiments were conducted in the *window environment*. In Fig. 9(a), the effect of window size on performance is depicted. Motion consisted of $T_x \in \{-1, +1\}$ and window radius consisted of $rad \in \{1, 2, 3\}$. In Fig. 9(b), the effect of the size of transformation space on performance is shown. For this set of experiments 3×3 windows were used, and the number of transformations consisted of $n(T) \in \{2, 4, 9\}$. In both cases some deterioration is evident in the early stages of learning, where it can be seen that convergence speed is inversely proportional to the size of transformation space and the size of the scanning window. Having said this, the graphs

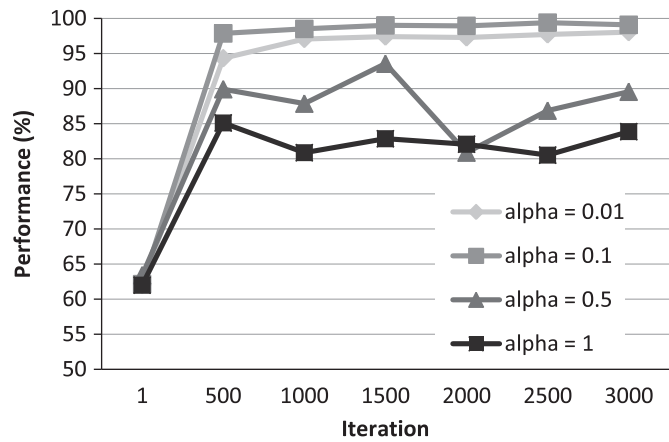


Fig. 8. The effect of a general learning rate on convergence.

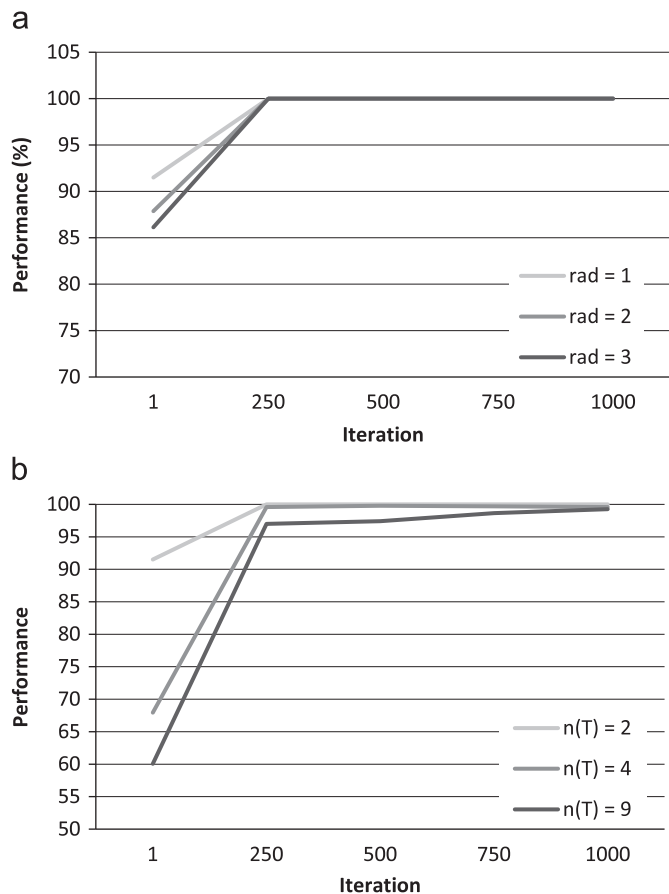


Fig. 9. The effect of window size and number of transformations on performance: (a) window size and (b) number of transformations.

also show evidence for the scalability of the solution. Those conditions which exhibit early lag, subsequently accelerate their learning, leading to shared convergence at iteration 1000 for all cases.

Fig. 10 provides us with an initial idea of how robust the solution is to noise. Again, simulations were run in the *window environment*, where motion was characterized by $T_x \in \{-1, 0, +1\}$ and $T_y \in \{-1, 0, +1\}$, which changed randomly every three iterations, with a window size of 5×5 . Each curve depicts performance across several iterations for different proportions of noisy pixels. As can be seen, for low to moderate levels of noise, SONN convergence is minimally affected. However, when the proportion of noisy pixels is 0.6 or more, convergence deteriorates significantly.

Recall Eq. (3), where it is shown how the detection of a correspondence depends on a similarity threshold (ST). One obvious way to deal with noise is to increase this threshold. Fig. 11 depicts the effect of different maximum noise intensities (MNI) and similarity thresholds at iteration 1000. Notice how for MNI=0, performance is inversely proportional to ST. When MNI is increased to 1, learning is improved for ST=1 but not ST = 2. The same observation is true for MNI=2. Except for ST = 1, performance is inversely proportional to MNI.

Fig. 12 provides an illustrative example of how image statistics can influence convergence in a dramatic way. Three randomized images were synthetically generated corresponding to three different levels of mean local variability (i.e. standard deviation, $\sigma = \sqrt{E[(X-\mu)^2]}$). For each image, the average of the standard deviations of all local image patches was calculated. The three images were classified into low (1.22), medium (5.78) and high (14.34) variability conditions. The simulation was run in a *window environment*, where motion consisted of $T_x \in \{-1, 0, +1\}$, and window size was 5×5 . The proportion of noisy pixels was set to

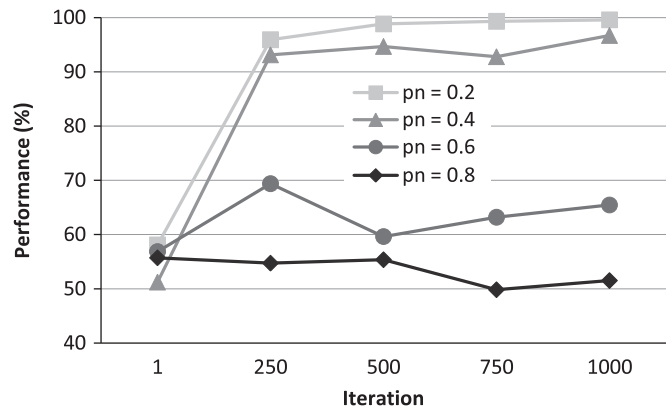


Fig. 10. How the proportion of noisy pixels affects performance.

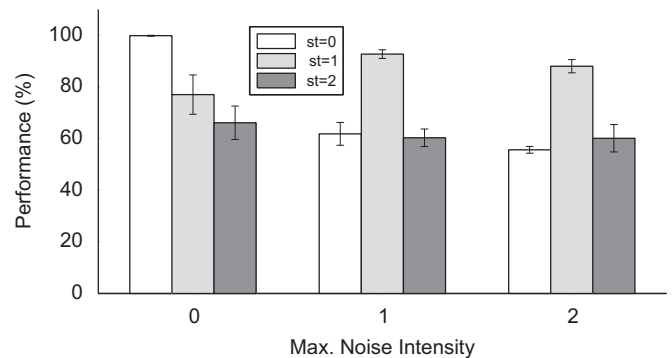


Fig. 11. The effect of different noise intensities and similarity thresholds.

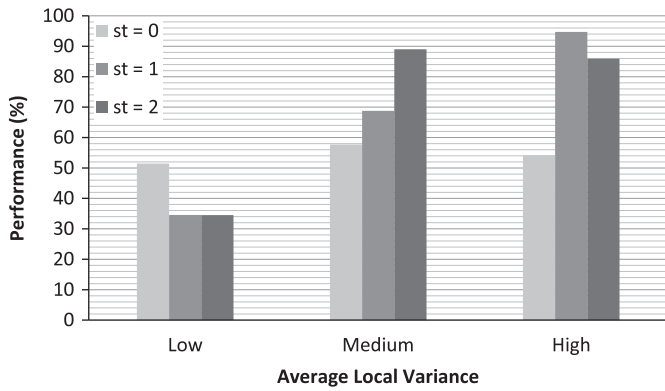


Fig. 12. The effect of average local variability and similarity thresholds on performance.

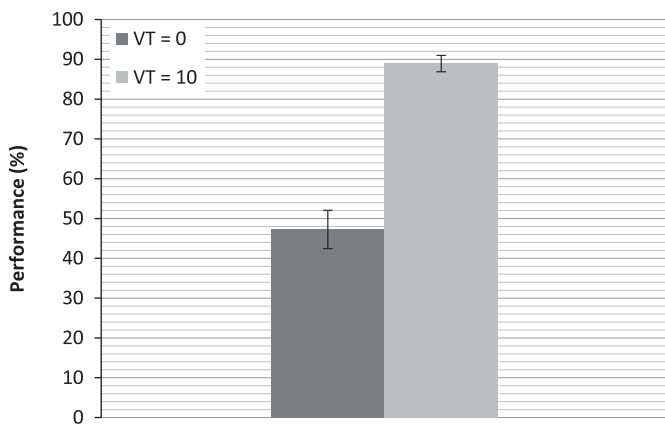


Fig. 13. The positive effect of variability dependent learning.

100% and the maximum magnitude of noise was set at 2. Performance levels were measured at iteration 1000. As one can see from the graph in Fig. 12, as the average local variability increases so does general performance, and so does the positive effect of larger similarity thresholds (in counteracting noise). Note that the difference between $st=1$ and $st=2$ for the high variability condition is statistically significant only at the 0.05 level (Wilcoxon Sum Rank test; $p = 0.014 > 0.01$).

The above findings prompted an experiment where learning was made to depend on the variability of the input images. In the simple version of *variability dependent learning* reported here, modification of weights was done only if the variability of the current input image was larger than some threshold (i.e. vt). For this experiment a *window* environment was used, with motion defined by $T_x \in \{-1, 0, +1\}$, window radius 5×5 , noise proportion 100%, maximum noise intensity 1 and similarity threshold 2. Fig. 13 shows the average performance level at iteration 2000, for the conditions $vt=0$ (equivalent to all previous experiments) and $vt=10$. The figure clearly shows that when learning takes variability into account (i.e. when $vt=10$), the potentially negative effects of increasing the similarity threshold can be counteracted. The difference between the two conditions is statistically significant (Wilcoxon Sum Rank test; $p \ll 0.01$).

Fig. 14 depicts how the type of window motion influences convergence. In these experiments, the range of motion was defined by $T_x \in \{-1, 0, +1\}$ and $T_y \in \{-1, 0, +1\}$ and the size of the window consisted of 3×3 . The random motion condition is shown in Fig. 14(a) whereas the nudge motion condition is shown in Fig. 14(b). In both cases, several curves are shown which

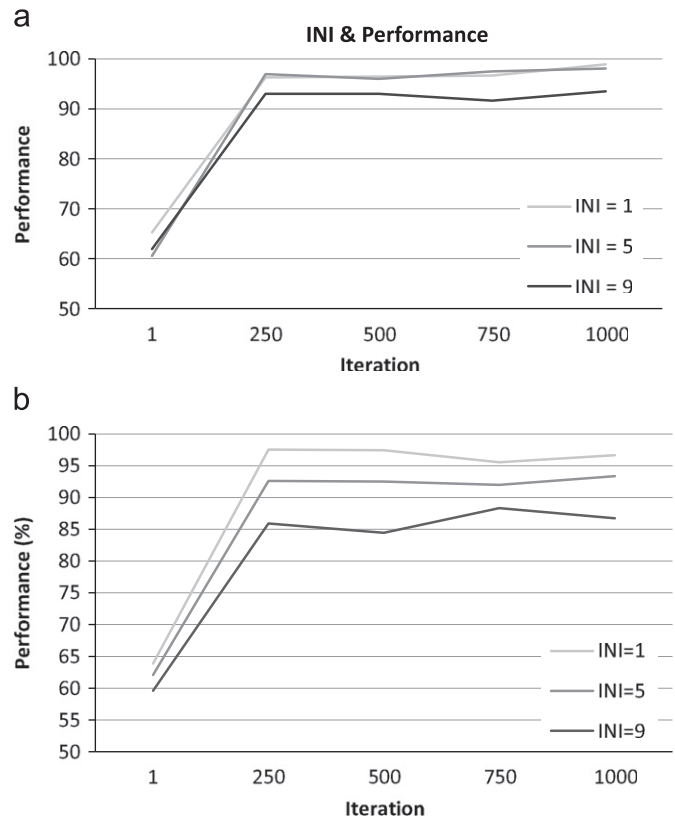


Fig. 14. The effect of type of window motion on performance: (a) random motion and (b) nudge motion.

correspond to different intervals between motion changes or inter-nudge intervals (INI). There appears to be a general decline in learning performance as this interval is increased for both cases. For example, there is a statistically significant performance difference in the *nudge* condition, for iteration 1000, between $INI=1$ and $INI=9$ (Wilcoxon Sum Rank test; $p=0.002$). Moreover, convergence seems to be generally worse in the *nudge* condition as compared to the *random* condition. For iteration 1000 and $INI=9$ there is a statistically significant performance difference between the motion conditions (Wilcoxon Sum Rank test; $p=0.021$).

Up to now we have dealt with cases where the number of vote nodes (i.e. $n(V)$) is equal to the number of transformations (i.e. $n(T)$). What happens when $n(V) \neq n(T)$? This inequality can be justified by referring to fluctuations of transformation statistics or coding considerations. If we assume that the number of neurons (and therefore the number of vote nodes) involved in a particular motion analysis circuit is fixed, but that the statistics of transformations change (i.e. the nature and size of the transformation space changes with time), then we have the possibility of $n(T)$ sometimes being smaller, or larger or equal to $n(V)$. This being the case, it would make sense for the circuit to adapt in order to optimize its representation relative to the environment. The other justification pertains to coding, and the fact that certain representations resulting from $n(V) \neq n(T)$ lead naturally to broad tuning and population coding strategies, which are advantageous from the point of view of accuracy and robustness [28].

Learning parameters for the set of experiments characterized by $n(V) \neq n(T)$ consisted of $\alpha = 0.1$, $\beta_1 = 1$, $\beta_2 = 0.1$, $\beta_3 = 0$ and $\beta_4 = 0$. Fig. 15 illustrates one particular experimental outcome for $n(V) < n(T)$. More specifically $n(V) = 3$ and $n(T) = 9$ (i.e. $T_x \in \{-1, 0, +1\}$ and $T_y \in \{-1, 0, +1\}$). Each matrix of squares represents a different vote node. Each square represents a correspondence type. For example, the square indexed by $y=0$ and $x=1$ represents all correspondences of

type $T_y = 0$ & $T_x=1$. The lighter a particular square is, the more correspondences with strong weights exist (of the type represented by the square). The resulting (broadly tuned) network in Fig. 15 shows that it is possible to learn in conditions where $n(V) < n(T)$, by making each vote node cover a larger area of the transformation space.

Fig. 16 attempts to give us some clues as to the effectiveness of these broadly tuned nodes. We conducted experiments based on the *window environment*, through which we collected data on window transformations and the corresponding activations of a network consisting of three vote nodes (trained as in Fig. 15). Window motion consisted of translations $T_x \in \{-1, +1\}$, $T_y \in \{-1, +1\}$ and no rotations. Fig. 16 depicts the average activations of each one of the three vote nodes for different transformations. From this figure we can observe three broadly tuned vote nodes, whose tuning patterns are sufficiently different from each other to suggest that the network can provide

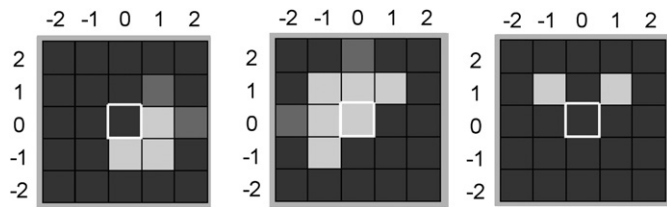


Fig. 15. Broadly tuned vote nodes when $n(V) < n(T)$.

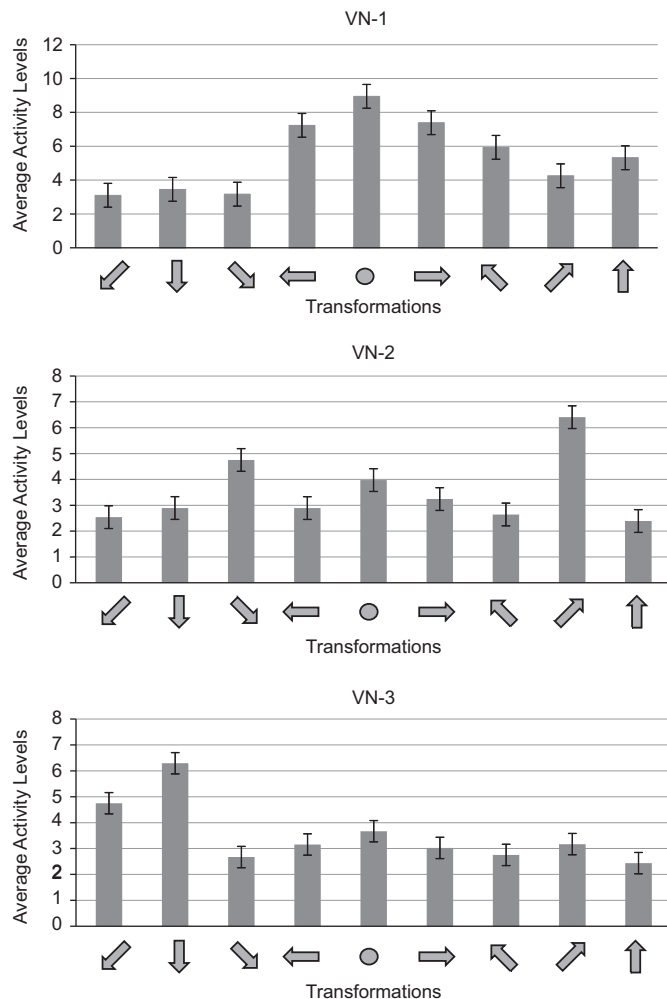


Fig. 16. Vote node tuning for a case of $n(V) < n(T)$.

useful information about a higher-dimensional transformation space. In order to confirm this, we tested the effectiveness of a multilayered perceptron (MLP) in associating patterns of vote node activations with transformations. An MLP with a single hidden layer with three hidden units, with *tansig* transfer functions in both layers, was trained for 1000 epochs, and reached a mean squared error (MSE) in validation tests of 0.0594. Although this is a rough indication of the usefulness of the low dimensional representation, future work is required to determine optimal broad tuning patterns and unsupervised methods capable of learning these optimal patterns.

Example correspondence distributions resulting from the reverse case (i.e. $n(V) > n(T)$) are illustrated in Fig. 17, where nine vote nodes were used and window motion was characterized by $T_x \in \{-1, 0, +1\}$. Window dimensions were 3×3 . As can be seen, vote nodes 1, 2 and 4 can be used for estimating the environmental motion. The remaining nodes exhibit a significant amount of symmetry and reflect the statistical nature of the correspondence distributions encountered (whether through motion or the self-similarity of patterns). This suggests that the resulting network configuration not only exhibits broad tuning, population coding and the ability to estimate motion, but also has basic shape representation capabilities. In other words, the pattern of vote-node activities contains elements of the transformation being observed and the nature of the pattern being transformed.

The problem of estimating translations and rotations simultaneously (where the unknowns are: T_x , T_y and θ) is unconstrained because single correspondences only have the power to constrain two unknowns. Fig. 18 depicts ten performance curves for networks that have learnt in a *window environment* where motion was characterized by $T_x \in \{-1, +1\}$, $T_y \in \{-1, +1\}$ and $\theta \in \{0, \pi\}$ and where the window size was 5×5 . The average performance at iteration 500 was 99.4%.

The effect of motion coherence on learning is depicted in Fig. 19. The *coherence environment* was used, with an image size of 5×5 , coherent motion characterized by $T_x \in \{-1, +1\}$ and incoherent motion characterized by $T_x \in \{-1, 0, +1\}$ and $T_y \in \{-1, 0, +1\}$. In order to cater for this new environment two basic modifications were adopted. Firstly, pixels with an intensity value of zero were defined as background, and therefore did not partake in correspondence detection. Without this exception, matches between background pixels would lead to a large number of false correspondences. Secondly, the learning rate β_2 was set to 0.5. This too

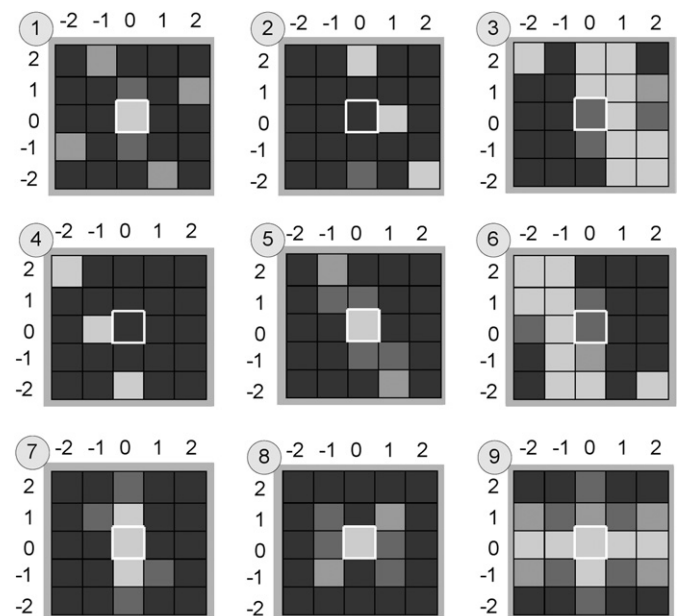


Fig. 17. Broadly tuned vote nodes when $n(V) > n(T)$.

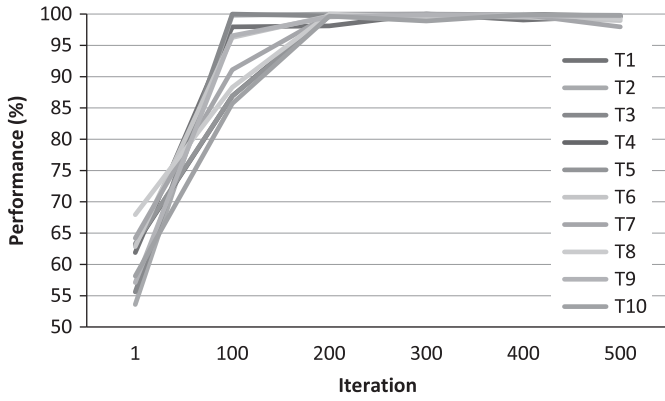


Fig. 18. Ten performance tests on the $T_x T_y \theta$ motion estimation problem.

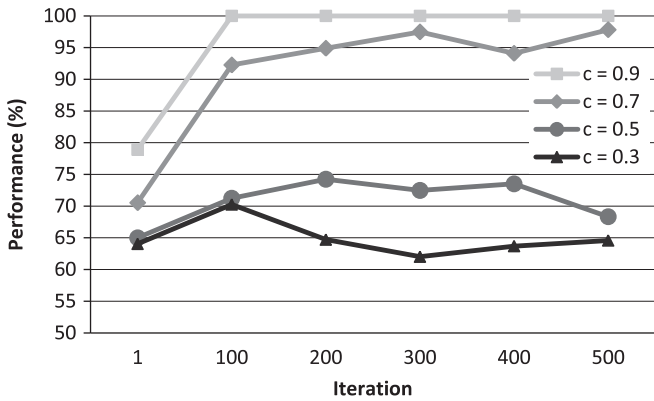


Fig. 19. The effect of motion coherence on convergence.

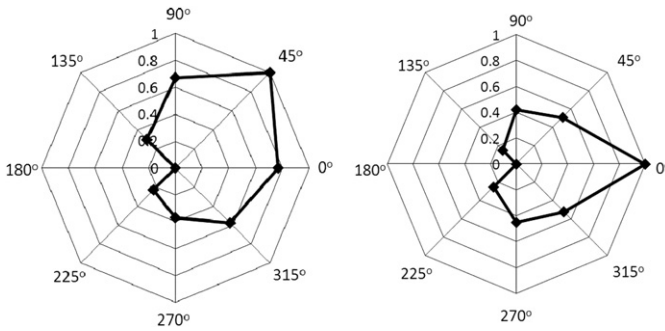


Fig. 20. Tuning properties of two vote nodes.

was due to background pixels, or “holes”. Without it, there is excessive suppression of inactive detectors in winning nodes leading to the eventual and total silencing of a node. Fig. 19 demonstrates how learning success is directly proportional to the amount of motion coherence manifested by a stimulus. Although learning is still somewhat robust at coherence 0.7, nevertheless there is a statistically significant performance difference between coherence levels 0.9 and 0.7, at iteration 500 (Wilcoxon Sum Rank test; $p < 0.0001$). Note that, the problem of *estimating* motion under different coherence conditions is distinct from, and most probably easier than, the problem of *learning* about motion under the same conditions. Most of the literature on motion coherence deals with the former problem.

Fig. 20 illustrates how the broad response properties of vote nodes can be strikingly consistent with biological data. Each spider chart represents the response properties of a different vote node

from a 9-node network that was trained on the set of transformations $T_x \in \{-1, 0, +1\}$ and $T_y \in \{-1, 0, +1\}$. Numbers on the vertical axis represent mean node activations. Activations were obtained by recording the responses of vote nodes after feeding the network with 3000 random transformations. The activations were subsequently normalized to fit in the interval $[0, 1]$. Axis angles represent motion direction. These tuning properties are very similar to those found for example in rabbit ON-OFF DSGCs [43], which points towards the validity of the proposed mechanism as a contributor to directional selectivity in the retina.

4. Discussion

From the above results, we can see that SONNs, as defined by Eqs. (2) and (3) and depicted in Fig. 4, can indeed learn (without supervision) how to perform motion analysis and do behave in a manner reflective of biological DS circuits (e.g. Fig. 20). The networks were shown to converge for different conditions of scalability (see Fig. 9), noise (see Fig. 10) and motion coherence (see Fig. 19). SONNs were even shown to be capable of learning unconstrained problems (see Fig. 18): the weight patterns of connections between second-order and vote nodes can be shown to reflect rigid transformation equations:

$$\begin{cases} T_x = x_2 - (x_1 \cos \theta - y_1 \sin \theta) \\ T_y = y_2 - (x_1 \sin \theta + y_1 \cos \theta) \end{cases} \quad (4)$$

where T_x and T_y refer to horizontal and vertical translation, respectively, θ refers to rotation, and (x_1, y_1) and (x_2, y_2) correspond to the 2D coordinates of correspondence features from the previous and current images, respectively. Based on the results reported in this paper, we propose that CDA as implemented by the SONN of Fig. 4 is an effective and biologically plausible model of motion analysis.

One potential weakness that we did identify was the algorithm’s sensitivity to different similarity thresholds. Thresholds larger than zero are required when images are affected by noise, which is a common occurrence in natural conditions. The problem with this is that a larger threshold leads to a larger proportion of false correspondences, which in turn tends to disrupt learning. Refer to Fig. 11 for an example of this deleterious effect, e.g.: condition $MNI=1$ and performances for $st=1$ and $st=2$. Intuitively, one can say that the number of false correspondences detected at a particular iteration is inversely proportional to the variability of the image patches being analyzed. A smaller variability, means that the intensities of different pixels tend to be more similar to each other, which means that they are more likely to match when the similarity threshold is larger than zero. To make matters worse, because there is significant correlation between neighboring pixels, the smaller the window of analysis, the smaller the variability.

Fig. 12 provided us with some experimental support for the above arguments, and suggested a possible solution. If we make learning somehow dependent on the variability of the input patterns, we can make the network ignore images that lead to a large number of false correspondences. Indeed, when we allow learning to take place only when the variability of the current image is larger than some threshold, the algorithm becomes significantly more robust to larger similarity thresholds, thus allowing it to cope with larger levels of noise (see Fig. 13).

One question one might ask pertains to the biological plausibility of *variability dependence*. Interestingly, there is evidence of subtraction of mean luminance signals at the outer retina (one of the putative functions of horizontal cells [32]). The results of these computations, if averaged again over some local region, can be seen to contribute to a measure of variability not far from the traditional standard deviation. Finally, this signal can then be used to influence learning through the adoption of neuromodulators [23] or chemical

synapses (most probably excitatory, seeing that the learning rate should be directly proportional to variability).

The uncovering of the effectiveness of *variability dependent learning* opens up many interesting avenues for further research. One question pertains to how to determine (or even modulate) the variability threshold. If it is too small, then learning becomes sensitive to larger similarity thresholds and cannot cope with much noise. If it is too large, then it can deal with larger similarity thresholds and noise levels, but will tend to face the problem of sparsity of learning. It should be worthwhile to do a thorough investigation into the relationships between image and noise statistics and similarity and variability thresholds. What are the limits of learning in the context of these factors? It will also be important to look into continuous non-linear relationships between variability and learning rate (as opposed to functions consisting of hard thresholds).

Having argued for the relevance and plausibility of *variability dependent learning*, note that if we consider BC outputs as the inputs into the SONN then the issue of noise might be negligible. The fact that many BCs pool inputs from several cones (e.g. primate diffuse BCs such as DB3 connect to 8 to 10 cone pedicles [5]), are connected to each other via gap junctions (GJs), and receive input from horizontal cells (HCs) which themselves pool inputs from many photoreceptors, means that their response properties have averaged (and thus attenuated) photoreceptor noise to a large extent. Future work should model the contrast based features encoded by BCs (including their noise properties), in order to study their effect on accuracy and learning.

One aspect of the solution that needs further improvement refers to the combinatorial problem manifested at the level of second-order nodes. If image resolution is defined by $n \times n$ then the network requires n^4 second-order nodes. In the same manner that Sigma-Pi-Sigma networks can address the combinatorial problems of Sigma-Pi networks, we also foresee a possible solution in the introduction of a Sigma layer between the input and second-order node layer. If again we consider diffuse BC inputs, whose receptive fields tend to be much larger than their dendritic fields (e.g. in the macaque monkey retina the typical dendritic field diameter is about 30–50 μm and yet the average receptive field center is about 90 μm [8], while in non-mammalian retinae this discrepancy is even larger due to more extensive GJ coupling), we can get an effective down-sampling of the input space which alleviates our combinatorial problem.

Since a core objective of this paper is to demonstrate how visual stimuli can be used to guide the development of a correlation-based motion analysis circuit, we need to consider the issue of how DS responses develop in the biological retina (refer to [14] for a recent review). According to [29], DS neurons in cortical area V1 require visual experience to develop correctly. In contrast, DS neurons of the retina seem to develop correctly irrespective of whether they receive natural visual stimulation or not [7]. This seems to imply that, at least in the context of correlation-based DS mechanisms in the retina, Hebbian mechanisms are irrelevant. However, this may not be the case, if we consider the phenomenon of retinal waves [45,16]. These waves have been shown to be important for circuit refinement and may play a role similar to coherent visual stimulation. Unfortunately this developmental role does not seem to be required in retinal DS. It was shown in [13,14] that cholinergic retinal waves (and visual stimulation) were not critical for the development of DS in mouse RGCs. This, however, does not exclude the possibility of active roles for other types of retinal waves [7] or the possibility that DS development might be guided by retinal waves in other species. In future work we intend to incorporate realistic retinal waves in order to study how the circuit develops and how it performs in terms of directional selectivity.

Future work will also explore different aspects of learning, representation and estimation, for conditions characterized by $n(V) \neq n(T)$. The hypotheses that nodes in Fig. 15 are implementing effective broad tuning strategies and that nodes in Fig. 17 are implementing both motion and shape analysis need further investigation.

One of our long-term goals is to develop large-scale and detailed models of the retina. Large-scale models will allow for a more thorough investigation of the cells and circuits involved in directional selectivity. Detailed compartmental models will allow us to address questions concerning the types of dendritic computations involved in DS. We hypothesize that functions such as those exemplified by Eq. (3) might be implemented at the dendritic level, possibly involving modified shunting inhibition mechanisms, because of their simplicity, usefulness and generality (e.g. directional selectivity and pattern matching). Apart from looking into the plausible mechanisms for implementing Eq. (3) future work should also look at alternative SONN implementations that will work with currently known dendritic mechanisms. One possibility would involve an amacrine cell computing squared feature differences at its dendrites and then inhibiting a DSGC.

To conclude, we have presented an approach to motion analysis, based on unsupervised learning in second-order neural networks, and shown its effectiveness in different internal (e.g. network size) and external (e.g. noise) conditions, and have argued for its biological relevance. Early results (Fig. 20) show that the model reflects known response properties of directionally selective ganglion cells thus supporting correlation-based mechanisms for retinal directional selectivity and suggesting that this line of enquiry is still valid, particularly when done in conjunction with investigations of other co-contributing mechanisms.

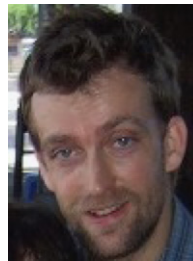
Acknowledgements

The authors would like to thank the anonymous reviewers for their very useful comments.

References

- [1] F.R. Amthor, C.W. Oyster, E.S. Takahashi, Morphology of on-off directionally selective ganglion cells in the rabbit retina, *Brain Research* 298 (1) (1984) 187–190.
- [2] H.B. Barlow, W.R. Levick, The mechanism of directionally selective units in rabbit's retina, *The Journal of Physiology* 178 (3) (1965) 477–504.
- [3] L.J. Borg-Graham, N.M. Grzywacz, A model of the directional selectivity circuit in retina: transformation by neuron singly and in concert, in: T. McKenna, J. Davis, S.F. Zornetzer (Eds.), *Single Neuron Computation*, San Diego Academic, 1992, pp. 347–376.
- [4] A. Borst, Correlation versus gradient type motion detectors: the pros and cons, *Philosophical Transactions of the Royal Society B: Biological Sciences* 362 (1479) (2007) 369–374.
- [5] B.B. Boycott, H. Wässle, Morphological classification of bipolar cells of the primate retina, *European Journal of Neuroscience* 3 (11) (1991) 1069–1088.
- [6] M.Z. Brown, D. Burschka, G.D. Hager, Advances in computational stereo, *IEEE Transactions on Pattern Analysis and Machine Intelligence* 25 (8) (2003) 993–1008.
- [7] M. Chen, S. Weng, Q. Deng, Z. Xu, S. He, Physiological properties of directionally selective ganglion cells in early postnatal and adult mouse retina, *The Journal of Physiology* 587 (2008) 819–828.
- [8] D. Dacey, O.S. Packer, L. Diller, D. Brainard, B. Peterson, B. Lee, Center surround receptive field structure of cone bipolar cells in primate retina, *Vision Research* 40 (14) (2000) 1801–1811.
- [9] R.F. Dacheux, M.F. Chimento, F.R. Amthor, Synaptic input to the on-off directionally selective ganglion cell in the rabbit retina, *The Journal of Comparative Neurology* 456 (3) (2003) 267–278.
- [10] J.B. Demb, Cellular mechanisms for direction selectivity in the retina, *Neuron* 55 (2) (2007) 179–186.
- [11] R. Durbin, D.E. Rumelhart, Product units: a computationally powerful and biologically plausible extension to backpropagation networks, *Neural Computation* 1 (1) (1989) 133–142.
- [12] M. Egelhaaf, A.A. Borst, B. Pilz, The role of GABA in detecting visual motion, *Brain Research* 509 (1) (1990) 156–160.
- [13] J. Elstrott, A. Anishchenko, M. Greschner, A. Sher, A.M. Litke, E.J. Chichilnisky, M.B. Feller, Direction selectivity in the retina is established independent of visual experience and cholinergic retinal waves, *Neuron* 58 (4) (2008) 499–506.
- [14] J. Elstrott, M.B. Feller, Vision and the establishment of direction-selectivity: a tale of two circuits, *Current Opinion in Neurobiology* 19 (3) (2009) 293–297.
- [15] T. Euler, P.B. Detwiler, W. Denk, Directionally selective calcium signals in dendrites of starburst amacrine cells, *Nature* 418 (6900) (2002) 845–852.
- [16] S.I. Firth, C.T. Wang, M.B. Feller, Retinal waves: mechanisms and function in visual system development, *Cell Calcium* 37 (5) (2005) 425–432.

- [17] S.I. Fried, R.H. Masland, Image processing: how the retina detects the direction of image motion, *Current Biology* 17 (2) (2007) 63–66.
- [18] S.I. Fried, T.A. Münch, F.S. Werblin, Mechanisms and circuitry underlying directional selectivity in the retina, *Nature* 420 (6914) (2002) 411–414.
- [19] J. Ghosh, Y. Shin, Efficient higher-order neural networks for classification and function approximation, *International Journal of Neural Systems* 3 (1993) 323.
- [20] C.L. Giles, T. Maxwell, Learning, invariance, and generalization in high-order neural networks, *Applied Optics* 26 (23) (1987) 4972–4978.
- [21] D.O. Hebb, *The Organization of Behavior: A Neuropsychological Theory*, Lawrence Erlbaum Associates, 2002.
- [22] J. Illingworth, J. Kittler, A survey of the Hough transform, *Computer Vision, Graphics, and Image Processing* 44 (1) (1988) 87–116.
- [23] A. Kirkwood, Neuromodulation of Cortical Synaptic Plasticity, Monoaminergic Modulation of Cortical Excitability, 2007, p. 209.
- [24] C. Koch, T.T. Poggio, V. Torre, Retinal ganglion cells: a functional interpretation of dendritic morphology, *Philosophical Transactions of the Royal Society of London, Series B, Biological Sciences* 298 (1090) (1982) 227–263.
- [25] C. Koch, *Biophysics of Computation: Information Processing in Single Neurons*, Oxford University Press, New York, 1999.
- [26] B. Krekelberg, Motion detection mechanisms, in: A. Basbaum et al. (Ed.), *The Senses: A Comprehensive Reference*, Elsevier Inc., Oxford, , 2008.
- [27] S. Lee, Z.J. Zhou, The synaptic mechanism of direction selectivity in distal processes of starburst amacrine cells, *Neuron* 51 (6) (2006) 787–799.
- [28] J.E. Lewis, Sensory processing and the network mechanisms for reading neuronal population codes, *Journal of Comparative Physiology A: Neuroethology, Sensory, Neural, and Behavioral Physiology* 185 (4) (1999) 373–378.
- [29] Y. Li, S. Van Hooser, M. Mazurek, L.E. White, D. Fitzpatrick, Experience with moving visual stimuli drives the early development of cortical direction selectivity, *Nature* 456 (2008) 952–956.
- [30] M. London, M. Häuser, Dendritic Computation, *Annual Review of Neuroscience* 28 (2005) 503–532.
- [31] J.C. Magee, Dendritic integration of excitatory synaptic input, *Nature Reviews Neuroscience* 1 (3) (2000) 181–190.
- [32] R.H. Masland, Neuronal diversity in the retina, *Current Opinion in Neurobiology* 11 (4) (2001) 431–436.
- [33] T.H. Maul, S. Baba, Neural clustering of correspondences for visual Pose estimation, in: *Proceedings of the 23rd European Conference on Modelling and Simulation*, 2009.
- [34] B.W. Mel, Information processing in dendritic trees, *Neural Computation* 6 (1994) 1031–1085.
- [35] N. Oesch, T. Euler, W.R. Taylor, Direction-selective dendritic action potentials in rabbit retina, *Neuron* 47 (5) (2005) 739–750.
- [36] B.A. Olshausen, C.H. Anderson, D.C. Van Essen, A neurobiological model of visual attention and invariant pattern recognition based on dynamic routing of information, *Journal of Neuroscience* 13 (11) (1993) 4700–4719.
- [37] C.W. Oyster, The analysis of image motion by the rabbit retina, *The Journal of Physiology* 199 (3) (1968) 613–635.
- [38] C.W. Oyster, F.R. Amthor, E.S. Takahashi, Dendritic architecture of ON–OFF direction-selective ganglion cells in the rabbit retina, *Vision Research* 33 (5–6) (1993) 579–608.
- [39] A. Polsky, B.W. Mel, J. Schiller, Computational subunits in thin dendrites of pyramidal cells, *Nature Neuroscience* 7 (6) (2004) 621–627.
- [40] W. Rall, Theoretical significance of dendritic trees for neuronal input–output relations, in: R. Reiss (Ed.), *In Neural Theory and Modeling*, Stanford University Press, 1964, pp. 73–97.
- [41] W. Reichardt, Autocorrelation, a principle for the evaluation of sensory information by the central nervous system, in: W.A. Rosenblith (Ed.), *Sensory Communication*, MIT/Wiley, New York, 1961, pp. 303–317.
- [42] W. Reichardt, Evaluation of optical motion information by movement detectors, *Journal of Comparative Physiology A: Neuroethology, Sensory, Neural, and Behavioral Physiology* 161 (4) (1987) 533–547.
- [43] W.R. Taylor, D.I. Vaney, Diverse synaptic mechanisms generate directional selectivity in the rabbit retina, *The Journal of Neuroscience* 22 (17) (2002) 7712–7720.
- [44] C. Weber, S. Wermter, A self-organizing map of sigma–pi units, *Neurocomputing* 70 (13–15) (2007) 2552–2560.
- [45] R.O.L. Wong, Retinal waves and visual system development, *Annual Review of Neuroscience* 22 (1) (1999) 29–47.



Tomaš H. Maul was born in Madeira, Portugal and did a B.Sc. in Biological Psychology at the University of St. Andrews, an M.Sc. in Computer Science at Imperial College and a Ph.D. in Computational Neuroscience at the University of Malaya. He worked for two years at MIMOS Bhd. as a Senior Researcher in the fields of Pattern Recognition and Computer Vision. He is currently an Assistant Professor at the University of Nottingham Malaysia Campus, where he conducts research in the areas of Neural Computation and Computer Vision.



Sapiyan Baba was born in Malaysia and did a B.Sc. at Essex University, an M.Sc. at Dundee University and a Ph.D. at Keele University. Professor Sapiyan Baba was recently serving as Dean of the Faculty of Computer Science and Information Technology at the University of Malaya. His areas of expertise include Artificial Intelligence, Neural Networks, Cognitive Science, Cognitive Robotics, Intelligent Tutoring Systems and Bioinformatics.