

Retina Color-Opponency Based Pursuit Implemented Through Spiking Neural Networks in the Neurorobotics Platform

Alessandro Ambrosano¹(✉), Lorenzo Vannucci¹, Ugo Albanese¹,
Murat Kirtay¹, Egidio Falotico¹, Pablo Martínez-Cañada⁶, Georg Hinkel⁴,
Jacques Kaiser², Stefan Ulbrich², Paul Levi², Christian Morillas⁶, Alois Knoll⁵,
Marc-Oliver Gewaltig³, and Cecilia Laschi¹

¹ The BioRobotics Institute, Scuola Superiore Sant’Anna,
Viale R. Piaggio 34, 56025 Pontedera, Italy
alessandro.ambrosano@sssup.it

² Department of Intelligent Systems and Production Engineering (ISPE IDS/TKS),
FZI Research Center for Information Technology, Haidund-Neu-Str. 10-14,
76131 Karlsruhe, Germany

³ Blue Brain Project (BBP), École polytechnique fédérale de Lausanne (EPFL),
Campus Biotech, Bâtiment B1, Ch. des Mines 9, 1202 Geneva, Switzerland

⁴ Department of Software Engineering (SE), FZI Research Center for Information
Technology, Haid-und-Neu-Str. 10-14, 76131 Karlsruhe, Germany

⁵ Department of Informatics, Technical University of Munich,
Boltzmannstrae 3, 85748 Garching, Germany

⁶ Department of Computer Architecture and Technology,
CITIC, University of Granada, Granada, Spain

Abstract. The ‘red-green’ pathway of the retina is classically recognized as one of the retinal mechanisms allowing humans to gather color information from light, by combining information from L-cones and M-cones in an opponent way. The precise retinal circuitry that allows the opponency process to occur is still uncertain, but it is known that signals from L-cones and M-cones, having a widely overlapping spectral response, contribute with opposite signs. In this paper, we simulate the red-green opponency process using a retina model based on linear-nonlinear analysis to characterize context adaptation and exploiting an image-processing approach to simulate the neural responses in order to track a moving target. Moreover, we integrate this model within a visual pursuit controller implemented as a spiking neural network to guide eye movements in a humanoid robot. Tests conducted in the Neurorobotics Platform confirm the effectiveness of the whole model. This work is the first step towards a bio-inspired smooth pursuit model embedding a retina model using spiking neural networks.

1 Introduction

One of the most important characteristics of the primate visual system is represented by the space-variant resolution retina with a high-resolution fovea that

offers considerable advantages for a detailed analysis of visual objects. When light hits the retina, *photoreceptor cells* transform it in electrochemical signals, which get furtherly processed first by flowing through *bipolar cells* and then through *ganglion cells*, whose axons form the optic nerve. Two more types of cells, *horizontal* and *amacrine*, participate in this process by connecting multiple photoreceptors and bipolar cells respectively, acting as feedback channels.

All these cells cooperate in a complex “retina circuit”, that can be split in many different “microcircuits” where around 80 types of identified neural cell types take part. These microcircuits process simultaneously the light and forward different information to the brain through dedicated pathways. Color information is carried by some of these pathways, combining signals from the cone photoreceptors, which are sensitive to coloured light [1].

There are three types of cone cells in the human retina, each one sensitive to a different wavelength range: S-cones are sensitive to short wavelengths, M-cones to middle wavelengths and L-cones to long wavelengths, corresponding roughly to blue, green and red light respectively. Both the density of photons on the cone and the color of the light determine the probability that an individual cone will capture a photon [2]. For this reason, signals from a single cone type can’t carry any color clue, but at least two types of cone must be compared in order to get actual color information.

Two processes of this sort are known to happen inside the retina: the *red-green opponency*, where M-cones and L-cones responses are taken into account in order to get color information in the red-green axis, and *yellow-blue opponency*, where all three kinds of cone are considered to get color information in the yellow-blue axis [3]. They are called opponency mechanisms because in both cases some cone type contribute with a positive weight and some other with a negative weight. In the red-green case we have M-cones opposed to L-cones whereas in the yellow-blue case the joint effect of M-cones and L-cones is opposed to S-cones response, though the exact circuitry of both processes is still under investigation.

The space-variant resolution of the retina requires efficient eye movements for correct vision. Two forms of eye movements — saccades and smooth pursuit — enable us to fixate the object on the fovea. Saccades are high-velocity gaze shifts that bring the image of an object of interest onto the fovea. The purpose of smooth pursuit eye movements is to minimise the retinal slip, i.e. the target velocity projected onto the retina, stabilizing the image of the moving object on the fovea. Retinal slip disappears once eye velocity catches up to target velocity in smooth pursuit eye movements. In primates, with a constant velocity or a sinusoidal target motion, the smooth pursuit gain, i.e. the ratio of tracking velocity to target velocity, is almost 1.0. In the last years several models of robotic visual pursuit have been developed. Shibata and colleagues suggested a control circuits for the integration of the most basic oculomotor behaviours [4] including the smooth pursuit eye movement. A similar model of smooth pursuit and catch-up saccade [5] was implemented on the iCub robot. Also models based on artificial neural networks were developed for visual tracking tasks [6–9]. In this paper we present a first attempt of embedding a retina model inside a visual pursuit controller suitable for a robotic

implementation. The controller, taking as input the output of the retina can generate an appropriate motor command to follow a moving target without needing for computer vision algorithm in extracting the target position from the incoming visual image. Moreover, the controller uses biologically inspired Spiking Neural Networks in order to implement parts of the controller, thus a proper framework that combines robotics and neural simulations has been used: the Neurorobotics Platform.

2 The Retina Simulation Platform

A great number of models have been proposed to reproduce different processing mechanisms of the retina [10–13]. However, these are often ad hoc models focused on fitting some specific retina functions rather than providing a general retina simulation platform. We chose the retina simulator COREM [14, 15], which includes a set of computational retinal microcircuits that can be used as basic building blocks for the modeling of different retina functions. In addition, COREM implements an interface with Spiking Neural Networks (SNNs) that allows its integration with models of higher visual areas and the Neurorobotics Platform.

The computational retinal microcircuits that can be configured within COREM consist of one spatial processing module (a space-variant Gaussian filter), two temporal modules (a low-pass temporal filter and a single-compartment model), a configurable time-independent nonlinearity and a Short-Term Plasticity (STP) function. The simulation engine allows the user to create custom retina scripts and to easily embed the retina model in the neural simulator.

3 The Neurorobotics Platform

The Neurorobotics Platform (NRP) is developed as part of the Human Brain Project¹ to run coupled neural and robotics simulations in an interactive platform. Whereas there are multiple neural simulators (e.g. Neuron [16], NEST [17]), robotics and physics simulators (e.g. Gazebo [18]), the NRP aims at offering a platform that combines the neuroscientific and robotic fields, by providing coupled physics and neural simulations. A core part of the NRP is the Closed-Loop-Engine (CLE) that allows to specify the data exchange between the brain simulation and the robot in a programmatic manner and orchestrates the simulations.

The key concept of the NRP is offering scientists an easy access to a simulation platform using a state-of-the-art web interface. Scientists are relieved from the burdensome installation process of scientific simulation software and are able to leverage large-scale computing resources. Furthermore, support for monitoring and visualizing the spiking activity of the neurons or joint states of the robot is offered as well as the camera image perceived by the robot.

To give an impression on how the platform looks like, a screenshot of a visual pursuit experiment with the iCub [19] humanoid robot is depicted in Fig. 1.

¹ <https://www.humanbrainproject.eu/>.

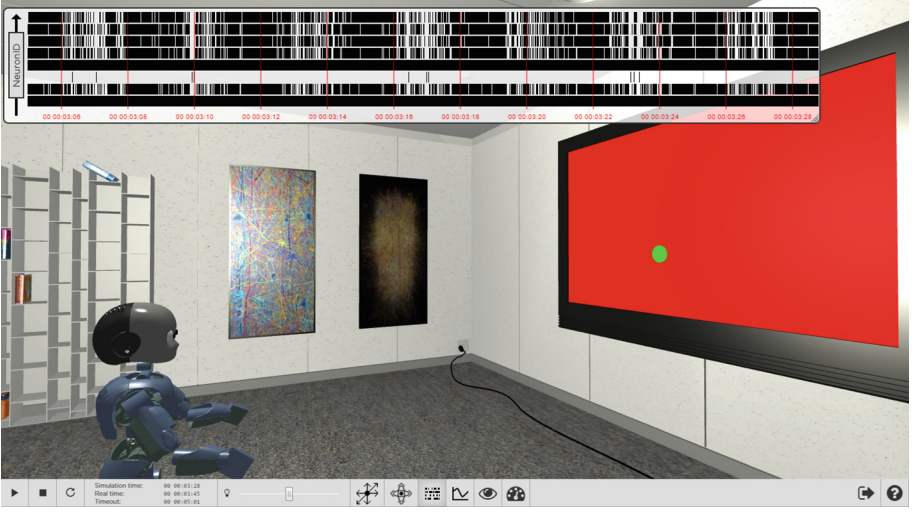


Fig. 1. Screenshot showing a pursuit experiment with the iCub robot in the Neurobotics Platform (NRP). A plot at the top shows spiking activity of neurons.

The Closed-Loop-Engine (CLE), depicted in Fig. 2, implements the core of the NRP. The CLE controls the neural and the world simulation, and executes a lightweight simulation data interchange mechanism. Neural and robotics simulation are iteratively run in parallel for the same amount of time, after which the transfer functions (special functions that translate output from one simulation into input for the other) are executed periodically. Communication with the brain is realized through recording and injecting spike data. Interfacing with the robot simulation is done using the Robot Operating System middleware (ROS [20]). In order to ensure reproducibility, data exchange is conducted in a deterministic fashion.

3.1 Integrating the Retina Simulation Platform in the NRP

The original retina codebase [15] was implemented as a stand-alone program. So the first step towards the embedding in the NRP has been to isolate all the core functionalities of the model in a separate module, that could be used as a library. The original executable has been kept as a “frontend” application depending on such library.

In order to have a retina model involved in the robot control inside a NRP simulation, it is necessary to forward the camera image or the robot to the retina, get the retina output information so it can be processed in a SNN, and finally decode the SNN output to control the robot. All these steps must take place inside transfer functions.

The whole retina model is written in C++, whereas all the NRP back-end, including transfer function related business logic, is written in Python.

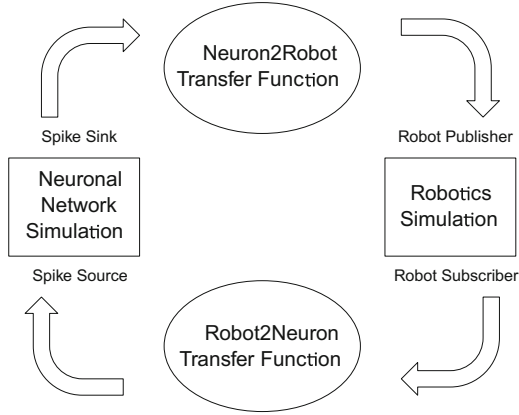


Fig. 2. A closed loop between a robotics simulation and a neural network

In particular, in order to ease the development of these transfer function modules, the part of the transfer functions that specify the connection to and from both simulations are implemented using a subset of python that defines a Domain Specific Language [21]. Thus, to allow the NRP to send input to the retina and get output from it, the functions in the retina library involved in these processes have been wrapped with Python bindings, making them accessible from Python code.

4 Retina Based Controller

We tested the retina-NRP interaction by creating a custom experiment on the platform performing a visual tracking task with a humanoid robot. Since our focus is on the red-green opponency, we created an environment in which a simulated iCub [19] humanoid robot will have to track a green ball moving horizontally on a red background.

In the experiment, image from one of the two cameras mounted on the robot eyes is sent to the retina simulator. Retinal output is then gathered and one horizontal stripe of the output image (320 pixels from a 320×240 image), intersecting the target, is forwarded to a neural network, where the spikes are filtered in order to avoid any undesired false detection. Finally, spike trains coming from the SNN are translated in a motion command for the robot eyes, making them following the target. An overview of the controller is shown in Fig. 3.

4.1 Retina Circuit

The retina circuit implemented with the framework described in Sect. 2 is composed of two symmetric subcircuits processing independently the input image. In the first layer we have simulated L-cones and M-cones, receiving the initial

image. Output from the photoreceptors is then forwarded to the two different subcircuits, namely the L^+M^- circuit and the L^-M^+ circuit.

The L^+M^- circuit (L^-M^+ description can be derived by symmetry) sums the output signals from the two cones, giving a positive weight to L-cones and a negative weight to M-cones. Combined cones signal is processed by simulated horizontal, bipolar, amacrine and ganglion cells. Every simulated ganglion cell is then sensible to a red center on green background within its receptive field. The high level result is a sensitivity to borders on a static image, which is slightly accentuated around borders between red and green objects for both circuits. In case of moving images instead, ganglion cells in the L^+M^- are particularly sensitive, due to their temporal characteristics, to green objects appearing in receptive fields that were earlier impressed by red objects. We will exploit this peculiarity to infer the position of an object by combining responses from the two circuits.

4.2 Brain Model

The spiking neural network used in our experiment comprises 1280 integrate and fire neurons [22], organized in two layers. The first layer acts as a current to spike converter, it has been designed for taking current intensities value coming from the retina library to neural spikes. In the second layer, every neuron, except the “side” ones, gather information from 7 subsequent neurons on the first layer, acting as a local spike integrator.

Every layer embeds two independent populations of 320 neurons, processing separately output from the two different ganglion cell layers of the retina circuit, so there is a one to one correspondence between considered pixels and neurons in a single first layer population. Thus, neurons from 1 to 320 will get input from the first ganglion circuit and forward spikes to neurons from 641 to 960, and neurons from 321 to 640 will get input from the second ganglion circuit and forward spikes to neurons from 961 to 1280.

The summation occurring in the second layer, together with linearly decreasing synapse weights from the center of the image to the periphery, serves as a filter for avoiding undesired spikes in peripheral regions of sight.

4.3 Transfer Functions

The robot controller is implemented by means of two transfer functions (TFs), one robot to neuron and one neuron to robot.

In the robot to neuron TF, the image on the eye camera of the robot is collected and forwarded to the retina library, updating its status. Outcome from the ganglion layers of the retina circuit are processed, then current values for a strip of pixel containing the target are transmitted to the two populations in the first layer of the brain.

Output spikes from the second brain layer are then processed by the second TF with the following steps:

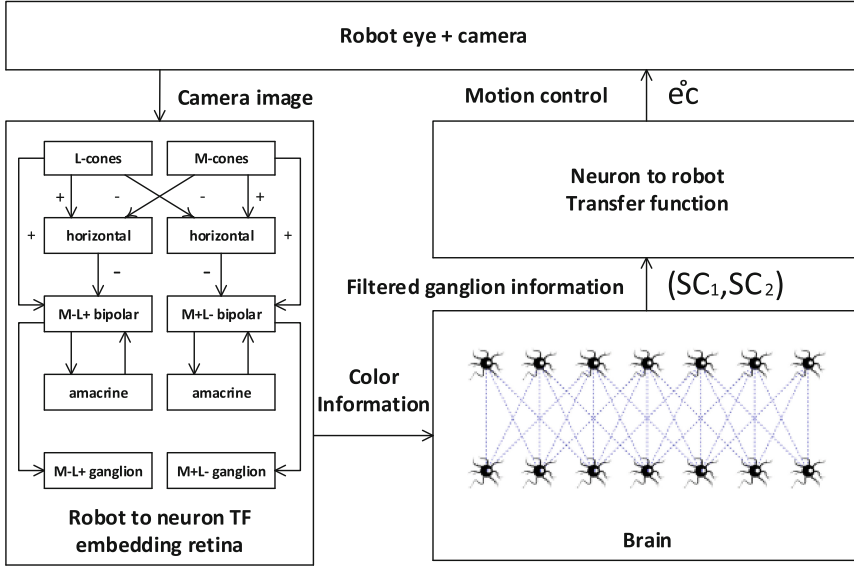


Fig. 3. Block diagram of the implemented robot controller. (Neuron image from Amelia Wattenberger, released under Creative Commons License 3.0 <https://thenounproject.com/term/neuron/79860/>)

- Spike counts for every neuron on the two output population are gathered in two collections SC_1 and SC_2 , that we represent as two functions

$$SC_1, SC_2 : [1, 320] \rightarrow \mathbb{N}$$

where $[1, 320]$ is a discrete set, representing a neural population.

- As higher stimulation of ganglions correspond to higher spike frequency in the brain, we expect one clearly distinguished frequency maximum per population, the first one corresponding to the pixel where the ball enters the receptive field in the first ganglion layer and the other to the pixel where the ball leaves the receptive field in the second ganglion layer. Since we are using one neuron per pixel on an horizontal stripe of pixels, the two maxima will correspond to two column indexes of the original image, which we will call p_1 and p_2 . For each population we obtain the said column index by computing first the maximum neural spike rate, then taking the first neuron index with that spike rate.

In formulas, p_1 and p_2 are defined as

$$p_1 = \min\{\arg \max_x \{SC_1(x)\}\}$$

$$p_2 = \min\{\arg \max_x \{SC_2(x)\}\}$$

- We take as an estimate of the target center the value

$$\bar{p} = \frac{p_1 + p_2}{2}$$

- The estimated position of the ball \bar{p} is then converted to an estimated angle \dot{p} with respect to the eye center with the equation

$$\dot{p} = -\arctan\left(\frac{\bar{p} - 160}{160}\right)$$

- The eye position angle command $\dot{e}c_{t+1}$, depending on the current eye position \dot{e}_t is computed as follows

$$\dot{e}c_{t+1} = \dot{e}_t + 0.2\dot{p}$$

where the constant 0.2 is determined empirically and prevents quick eye movements that could result in the target loss from the robot sight.

5 Results

5.1 Target Detection

In this experiment the controller has been validated by testing only its target detection capabilities without any actuation on the robot. The experiment was

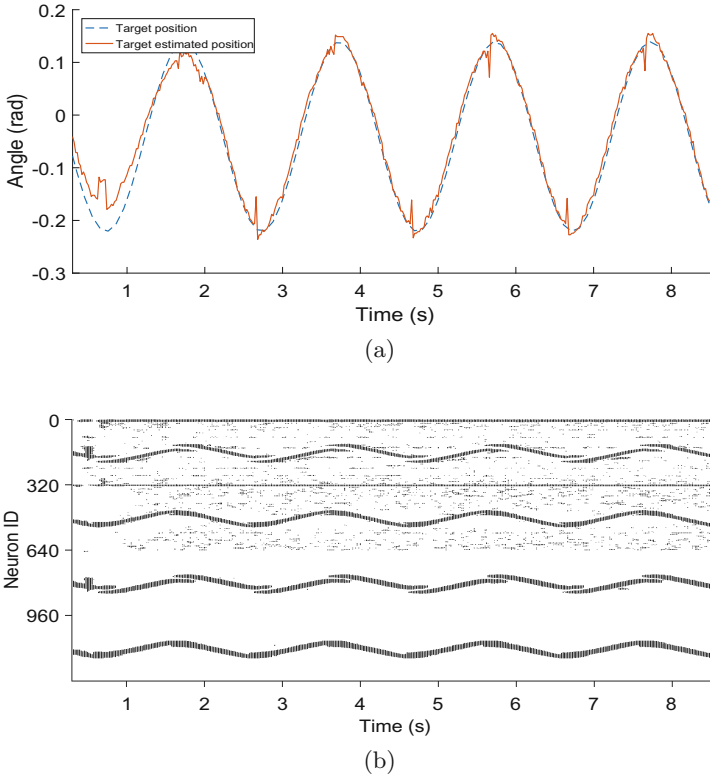


Fig. 4. The computed ball position (a) and the brain response (b) for a ball with sinusoidal motion moving at 0.5 Hz, robot eyes still. (Color figure online)

started with a slightly different neuron to robot transfer function which didn't send any control message back to the robot and with the target ball moving with a sinusoidal trajectory with a cycle frequency of 0.5 Hz, covering almost half the field of vision of the camera. During the experiment, data about both target perceived location and the neural network response was collected and it is reported in Fig. 4.

At the beginning of the simulation, the robot suddenly moves to a default position, while the controller responsible for the upright position starts. For this reason, the camera image during the first 2 seconds of simulation may be affected by this movement and thus its elaboration may provide wrong results. In Fig. 4a we can observe how the estimated target position, after the robot stabilization, follows a sinusoidal trend. The target position is plotted with a dashed line as a reference. Sensitivity of different ganglion inputs can be noticed in the first half of the spike plot of Fig. 4b, where the first population spikes in correspondence to the target entering the receptive field and the second spikes when the target

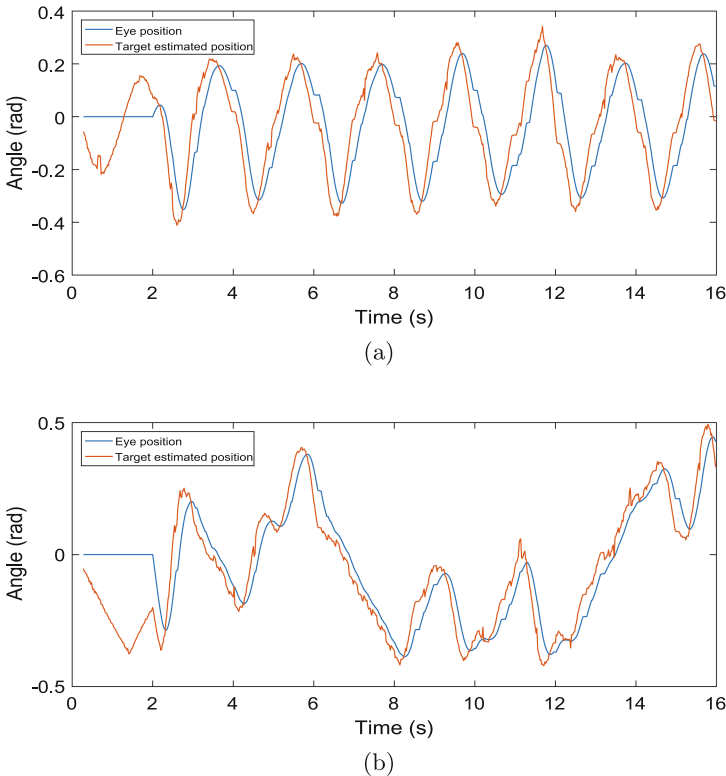


Fig. 5. Tracking results with sinusoidal (a) and random linear (b) trajectories (Color figure online)

leaves the receptive field. The filtering action of the second neural layer can be observed in the second half of the spike plot.

5.2 Target Pursuit

In these experiments we used the controller enabling motion commands with the same setup of the previous experiment. We tested the controller with a target moving with a sinusoidal trajectory (Fig. 5a) and with a random linear trajectory (Fig. 5b), namely a linear trajectory changing direction (left or right) randomly every second. For the same reasons explained in the previous test, we wait for the robot stabilization before starting actuating the eye. It can be observed how the target estimation is still effective even with a moving eye, which validates our choice of a retina simulator as a data source for the controller.

Comparing our approach with a purely image-processing based target detection [23], we can easily observe a greater estimate error on our controller, that can be noticed easily in Fig. 5b at seconds 5, 10 and 12. A one to one comparison, though, is imprecise, as in the image-processing approach all the pixels of the camera image are processed by the controller, whereas in our implementation we consider just one stripe of 320 pixels.

6 Discussion and Future Works

In this paper we propose a retinal red-green opponency based robot controller, processing image from the outside world through a retina model instead of using classic image-processing methods. We set up a framework by integrating an existing retina framework in the NRP, implementing a custom retina circuit and a custom neural network and finally setting up an experiment in the NRP with suitable transfer functions. Two experiments allowed us to validate the effectiveness of this setup for both detection and pursuit of a moving green target on a red background, although with the limitations of a single retina pathway and a single image stripe processed by the brain model. This work represents a first attempt towards a bio-inspired visual pursuit controller embedding a retina model. In future works, we plan to extend the “field of view” of the controller to the whole image, after a careful design of the pursuit controller in order to simulate a brain function areas involved in the smooth pursuit eye movement through spiking neural networks. We will improve our retina circuit to simulate also space-variant resolution and try to model and combine more retinal pathways concurring in color detection or motion detection. In order to interface this controller model with a real robotic platform, the retina model could be interfaced with real-time neural simulations such as the ones provided by neuromorphic hardware like SpiNNaker [24].

Acknowledgements. The research leading to these results has received funding from the European Union Seventh Framework Programme (FP7/2007-2013) under grant agreement no. 604102 (Human Brain Project). The authors would like to thank the

Italian Ministry of Foreign Affairs, General Directorate for the Promotion of the “Country System”, Bilateral and Multilateral Scientific and Technological Cooperation Unit, for the support through the Joint Laboratory on Biorobotics Engineering project.

References

1. Dacey, D.M.: Primate retina: cell types, circuits and color opponency. *Prog. Retinal Eye Res.* **18**(6), 737–763 (1999)
2. Baylor, D., Nunn, B., Schnapf, J.: Spectral sensitivity of cones of the monkey *Macaca fascicularis*. *J. Physiol.* **390**, 145 (1987)
3. Dacey, D.M., Packer, O.S.: Colour coding in the primate retina: diverse cell types and cone-specific circuitry. *Curr. Opin. Neurobiol.* **13**(4), 421–427 (2003)
4. Shibata, T., Vijayakumar, S., Conradt, J., Schaal, S.: Biomimetic oculomotor control. *Adapt. Behav.* **9**(3–4), 189–207 (2001)
5. Falotico, E., Zambrano, D., Muscolo, G., Marazzato, L., Dario, P., Laschi, C.: Implementation of a bio-inspired visual tracking model on the icub robot. In: *Proceedings of IEEE International Workshop on Robot and Human Interactive Communication*, pp. 564–569 (2010)
6. Vannucci, L., Cauli, N., Falotico, E., Bernardino, A., Laschi, C.: Adaptive visual pursuit involving eye-head coordination and prediction of the target motion. In: *IEEE-RAS International Conference on Humanoid Robots*, pp. 541–546 (2014)
7. Vannucci, L., Falotico, E., Di Lecce, N., Dario, P., Laschi, C.: Integrating feedback and predictive control in a bio-inspired model of visual pursuit implemented on a humanoid robot. In: Wilson, S.P., Verschure, P.F.M.J., Mura, A., Prescott, T.J. (eds.) *Living Machines 2015. LNCS (LNAI)*, vol. 9222, pp. 256–267. Springer, Heidelberg (2015)
8. Zambrano, D., Falotico, E., Manfredi, L., Laschi, C.: A model of the smooth pursuit eye movement with prediction and learning. *Appl. Bionics Biomech.* **7**(2), 109–118 (2010)
9. Falotico, E., Taiana, M., Zambrano, D., Bernardino, A., Santos-Victor, J., Dario, P., Laschi, C.: Predictive tracking across occlusions in the icub robot. In: *9th IEEE-RAS International Conference on Humanoid Robots, HUMANOIDS 2009*, pp. 486–491 (2009)
10. Benoit, A., Caplier, A., Durette, B., Héroult, J.: Using human visual system modeling for bio-inspired low level image processing. *Comput. Vis. Image Underst.* **114**(7), 758–773 (2010)
11. Wöhrer, A., Kornprobst, P.: Virtual retina: a biological retina model and simulator, with contrast gain control. *J. Comput. Neurosci.* **26**(2), 219–249 (2009)
12. Héroult, J., Durette, B.: Modeling visual perception for image processing. In: Sandoval, F., Prieto, A.G., Cabestany, J., Graña, M. (eds.) *IWANN 2007. LNCS*, vol. 4507, pp. 662–675. Springer, Heidelberg (2007)
13. Morillas, C.A., Romero, S.F., Martínez, A., Pelayo, F.J., Ros, E., Fernández, E.: A design framework to model retinas. *Biosystems* **87**(2), 156–163 (2007)
14. Martínez-Cañada, P., Morillas, C., Pino, B., Ros, E., Pelayo, F.: A computational framework for realistic retina modeling. *Int. J. Neural Syst.* (Accepted for publication)
15. Martínez-Cañada, P., Morillas, C., Nieves, J.L., Pino, B., Pelayo, F.: First stage of a human visual system simulator: the retina. In: Trémeau, A., Schettini, R., Tominaga, S. (eds.) *CCIW 2015. LNCS*, vol. 9016, pp. 118–127. Springer, Heidelberg (2015)

16. Hines, M.L., Carnevale, N.T.: The NEURON simulation environment. *Neural Comput.* **9**(6), 1179–1209 (1997)
17. Gewaltig, M.O., Diesmann, M.: NEST (NEural Simulation Tool). *Scholarpedia* **2**(4), 1430 (2007)
18. Koenig, N., Howard, A.: Design and use paradigms for gazebo, an open-source multi-robot simulator. In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2004)*, vol. 3, pp. 2149–2154. IEEE (2004)
19. Metta, G., Natale, L., Nori, F., Sandini, G., Vernon, D., Fadiga, L., Von Hofsten, C., Rosander, K., Lopes, M., Santos-Victor, J., et al.: The iCub humanoid robot: an open-systems platform for research in cognitive development. *Neural Netw.* **23**(8), 1125–1134 (2010)
20. Quigley, M., Conley, K., Gerkey, B., Faust, J., Foote, T., Leibs, J., Wheeler, R., Ng, A.Y.: ROS: an open-source robot operating system. In: *ICRA Workshop on Open Source Software*, vol. 3, p. 5 (2009)
21. Hinkel, G., Groenda, H., Vannucci, L., Denninger, O., Cauli, N., Ulbrich, S.: A domain-specific language (DSL) for integrating neuronal networks in robot control. In: *ACM International Conference Proceeding Series*, pp. 9–15, 21 July 2015
22. Brette, R., Gerstner, W.: Adaptive exponential integrate-and-fire model as an effective description of neuronal activity. *J. Neurophysiol.* **94**(5), 3637–3642 (2005)
23. Vannucci, L., Ambrosano, A., Cauli, N., Albanese, U., Falotico, E., Ulbrich, S., Pfozter, L., Hinkel, G., Denninger, O., Peppicelli, D., Guyot, L., Von Arnim, A., Deser, S., Maier, P., Dillman, R., Klinker, G., Levi, P., Knoll, A., Gewaltig, M.O., Laschi, C.: A visual tracking model implemented on the iCub robot as a use case for a novel neurorobotic toolkit integrating brain and physics simulation. In: *IEEE-RAS International Conference on Humanoid Robots*, pp. 1179–1184 (2015)
24. Painkras, E., Plana, L.A., Garside, J., Temple, S., Galluppi, F., Patterson, C., Lester, D.R., Brown, A.D., Furber, S.B.: SpiNNaker: A 1-w 18-core system-on-chip for massively-parallel neural network simulation. *IEEE J. Solid-State Circuits* **48**(8), 1943–1953 (2013)

Biomimetic and Biohybrid Systems

5th International Conference, Living Machines 2016,

Edinburgh, UK, July 19-22, 2016. Proceedings

Lepora, N.F.; Mura, A.; Mangan, M.; Verschure, P.F.M.J.;

Desmulliez, M.; Prescott, T.J. (Eds.)

2016, XVI, 555 p. 302 illus., Softcover

ISBN: 978-3-319-42416-3