2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids)
November 3 - 5, 2015, Seoul, Korea

A visual tracking model implemented on the iCub robot as a use case for a novel neurorobotic toolkit integrating brain and physics simulation

Lorenzo Vannucci[1], Alessandro Ambrosano[1], Nino Cauli[1], Ugo Albanese[1], Egidio Falotico[1], Stefan Ulbrich[2], Lars Pfotzer[2], Georg Hinkel[5], Oliver Denninger[5], Daniel Peppicelli[3], Luc Guyot[3], Axel Von Arnim[4], Stefan Deser[6], Patrick Maier[6], Rüdiger Dillman[2], Gundrun Klinker[6], Paul Levi[2], Alois Knoll[6], Marc-Oliver Gewaltig[3], Cecilia Laschi[1]

***Abstract*— Developing neuro-inspired computing paradigms that mimic nervous system function is an emerging field of research that fosters our model understanding of the biological system and targets technical applications in artificial systems. The computational power of simulated brain circuits makes them a very promising tool for the development for brain-controlled robots. Early phases of robotic controllers development make extensive use of simulators as they are easy, fast and cheap tools. In order to develop robotics controllers that encompass brain models, a tool that include both neural simulation and physics simulation is missing. Such a tool would require the capability of orchestrating and synchronizing both simulations as well as managing the exchange of data between them. The Neurorobotics Platform (NRP) aims at filling this gap through an integrated software toolkit enabling an experimenter to design and execute a virtual experiment with a simulated robot using customized brain models. As a use case for the NRP, the iCub robot has been integrated into the platform and connected to a spiking neural network. In particular, experiments of visual tracking have been conducted in order to demonstrate the potentiality of such a platform.**

I. INTRODUCTION

Simulation is typically used to easily evaluate and verify technical approaches and methods without the usage of real hardware like robotic systems. The main reason to use simulations is that they can provide a large number of robotics platforms without the need for buying and maintaining them. In order for these simulation to be useful to develop controller that are also suitable for the real robots, the physics simulation must be as realistic as possible.

At the neuro-scientific side, also sophisticated simulations of brain circuits are being developed. In order to have a realistic brain simulation that takes into account the dynamics of neurons, one has to use Spiking Neural Networks (SNN). These kinds of network differ from classic artificial neural networks as they are computationally more powerful than the latter, considering the number of neuron that are needed to solve the same task in both cases[1]. On the other hand, the level of detail required to simulate SNN (simulation of a membrane voltage potential through differential equations) makes it so that more computational power and an optimized implementation is required.

In order to integrate both worlds, neural simulation and physics simulation, a proper tool is missing. Such a tool would require the capability of orchestrating and synchronizing both simulations as well as managing the exchange of data between them. In a work by Tolu and colleagues[2], the EDLUT brain simulator was coupled with a robotic simulation in order to perform closed loop experiments, but the solution proposed was created ad-hoc for the work, thus it did not provide flexibility. The Neurorobotics Platform (NRP), currently developed within the Human Brain Project (HBP)[1], aims at filling these gaps.

In order to conduct a first proof-of-concept of the NRP, this work shows the simulation of a visual tracking control algorithm implemented on a humanoid robot as simple spiking neural network. In the last years several models of robotic visual tracking have been developed. Shibata and colleagues suggested a control circuit for the integration of the most basic oculomotor behaviours[3] including the smooth pursuit eye movement. A similar model of smooth pursuit and catch-up saccades[4], [5] or occlusions[6] was implemented on the iCub robot. Also models based on artificial neural networks were developed for visual tracking tasks[7][8][9]. The objective of this work is not to provide a visual tracking model that outperforms existing robotic controllers, but to present a spiking model which represents a use case for the Neurorobotics Platform.

II. THE NEUROROBOTICS PLATFORM

The Neurorobotics Platform is an integrated software toolkit enabling an experimenter to design and execute a virtual experiment with a simulated robot using customized brain models. The aim of this platform is twofold: from one side the platform can be used to test neuroscientific models of brain areas, or even reconstruction of these areas based on neurophysiological data; on the other side roboticists can take

[1]The BioRobotics Institute, Scuola Superiore Sant'Anna, Viale R. Piaggio 34, 56025 Pontedera, Italy
[2]Department of Intelligent Systems and Production Engineering (ISPE – IDS/TKS) at the FZI Research Center for Information Technology, Haid-und-Neu-Str. 10-14, 76131 Karlsruhe, Germany
[3]Blue Brain Project (BBP) at École polytechnique fédérale de Lausanne (EPFL), Campus Biotech, Bâtiment B1, Ch. des Mines 9, CH-1202 Genève, Switzerland
[4]fortiss GmbH, Guerickestrasse 25, 80805 Munich, Germany
[5]Department of Software Engineering (SE) at the FZI Research Center for Information Technology, Haid-und-Neu-Str. 10-14, 76131 Karlsruhe, Germany
[6]Department of Informatics at Technical University of Munich, Boltzmannstraße 3, 85748 Garching, Germany

[1]www.humanbrainproject.eu

978-1-4799-6885-5/15/$31.00 ©2015 IEEE

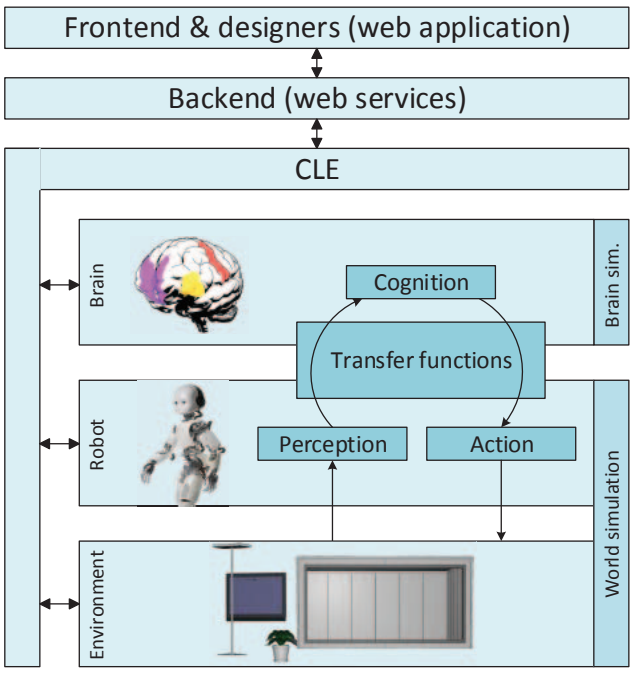

Fig. 1. Functional architecture of the Neurorobotics Platform.

advantage of such a platform to develop more biologically inspired control architectures.

The physical and neural simulations are properly synchronized and they exchange data through *transfer functions* that translate sensory information originating from the robot (camera image, encoders, etc.) into input for the brain (current and spikes) from one side and the network output into motor commands from the other,

Figure 1 illustrates the functional architecture of the NRP. It can be seen that the platform is not only composed of the robot and brain simulations, but it provides a complete framework in which experiments can be constructed from scratch, by using the designers. The platform also provides a web interface, so that it can be easily accessed and used by a broader user base. One of the pillars of the NRP development is the reuse and extension of existing software, thus many components were implemented using suitably chosen existing software.

In the rest of this section, the components that constitute the platform will be described in detail.

A. Brain Simulator

The goal of the Brain Simulator is to simulate a brain circuit, implemented with a spiking neural network.

Several simulators for SNNs exist, with different kind of detail, ranging from more abstract point neuron simulations, that consider neural networks as direct graphs to the morphologically accurate ones, where also the properties of axons and dendrites are taken into account.

Inside the NRP, the simulator currently supported is NEST[10], a point neuron simulator with the capability of running on high-performance computing platforms. NEST is supported through the use of the PyNN abstraction layer[11] that provides the same interface for different simulators and also for neuromorphic processing units, i.e. dedicated hardware for the simulation of SNN, such as SpiNNaker[12].

B. World Simulator

In order to have realistic experiments, the accurate brain simulation must be coupled with a detailed physics simulation. The *World Simulator* component aims at delivering a realistic simulation for both the robot and the environment in which the robot interacts.

Gazebo[13] was chosen as the physics simulator. It offers a multi-robot environment with an accurate simulation of the dynamics, in particular of gravity, contact forces and friction. This dynamic simulation can be computed with different supported software libraries like ODE[14] and Bullet[15].

Any communication with the simulated robot and Gazebo itself is done through the Robot Operating System (ROS)[16]. This will in principle allow an easy exchange of the simulated robot with its physical counterpart.

C. Transfer Functions

As mentioned earlier, the Transfer Functions (TF) are special functions that connect the physics and the neural simulations. Two kinds of functions can be distinguished: the robot to neuron ones (R2N) translate sensory information coming from the robot to spikes and current that can be sent as an input to the brain simulation; the neuron to robot ones (N2R) take measurements on the neural network such as spike rate or membrane potential and transform them into motor commands that control the robot.

From the robot side, these transfer functions communicate through ROS topic, while on the brain simulation side data is exchanged via devices, which are wrappers around PyNN nodes such as spike sources or neuron models.

Transfer Functions can be specified using the TF framework, which effectively offers an internal Domain-Specific Language hosted in Python[17]. Currently, the TF body can only be defined by implementing it in Python or through an XML file which is translated into Python code. The ROS topics and PyNN devices on which each transfer function must be wired, are specified through the mentioned DSL. In order to support different levels of users, from the roboticist to the neuroscientist, a graphical user interface that facilitate the usage of the framework will be developed.

D. Closed Loop Engine

The Closed Loop Engine (CLE) is responsible for the control of the synchronization as well as for the data exchange among the simulations and the TFs. The purpose of the CLE is to guarantee that both the simulations start and run for the same timestep, and also to run the TFs that will use the data collected at the end of the simulations. Figure 2 shows a sequence diagram of a typical execution of a timestep: after Gazebo and NEST have completed their execution in parallel, the TFs receive and process data from the simulations and produce an output which is the input for the next execution.

From the point of view of the implementation, the timestep of the physic simulation is sent to Gazebo through a ROS service call, while the brain simulation is directly run for the desired timestep with a PyNN call.

The idea behind the proposed synchronization mechanism is to let both the simulations run for a fixed timestep, receiving and processing the output of the previous steps and yielding data that will be processed in the future steps by the concurrent simulation. In other words it is not possible to receive data yielded in the current timestep by the concurrent simulation. This can be read as the TFs introducing a delay of sensory perception and motion actuation greater than the simulation timestep.

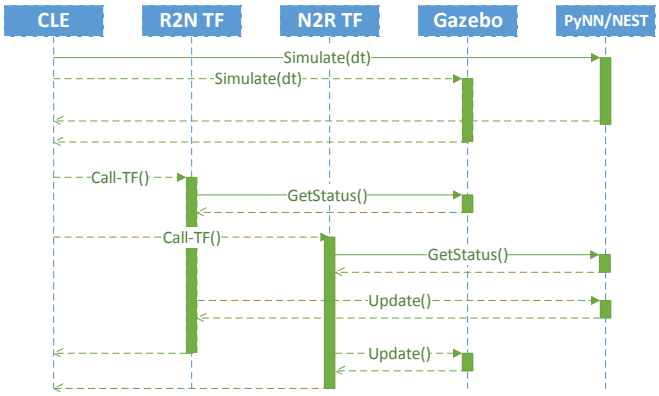

Fig. 2. Synchronization between the components of a simulation, as orchestrated by the CLE.

E. Designers

In order for the user to design his own experiments, three different designers are being developed inside the NRP.

Through the use of the Robot Designer, the user is able to create his own robot either by designing it from scratch or by editing an existing one. Aside from the appearance of the robot, kinematic parameters can be edited and actuators and sensor can be added.

The aim of the Environment Designer is to create an environment in which the experiment will run. The environment can be modelled from scratch or built from objects present in a predefined library.

Using the Experiment Designer the user is able to design an experiment. In this context an experiment is composed of: robot, environment, brain model, transfer functions and a collection of events (i.e. time triggered actions that affect the environment).

F. Frontend

The Frontend is the user interface to the NRP. Using it, the user is able to access the Designers, select and run the experiments and visualize the execution and results. Through this interface, it is also possible to select the hardware platform on which the simulation will run, with the option of high-performance computers or neuromorphic hardware. During the simulation the user is able to control the execution (play, pause, reset, stop) and see realtime data gathered from the simulation (brain activity, sensor data, camera images) as overlays over the rendered 3D scene.

In order to have a coherent interface and user experience for the Frontend and other informatics tools developed as part of the Human Brain Project, the Frontend is implemented as a web application that communicates with the CLE via a Backend that offers REST APIs.

III. VISUAL TRACKING - A SIMPLE USE CASE

As an example to demonstrate the applicability of spiking neural networks for movement generation, we implemented a tracking experiment that shows how a humanoid robot can be controlled by an "artificial brain". In particular, we focused on a visual tracking task. A humanoid robot stands in front of a blue screen with a moving green circle which is the target of the tracking. The robot reaches and follows such a target using only eye movements. In this task it uses only horizontal eye movements, but the same controller can be easily extended to the vertical axis. The humanoid robot we used for this task is the iCub robot.

The iCub robot in the Neurorobotics Platform

We decided to test our controller on a Gazebo model of the iCub robot imported in the Neurorobotics Platform. The original robot model[2] had 32 active joints: 3 for the head (neck roll, pitch and yaw), 3 for the torso (roll, pitch and yaw), 7 for each arm (shoulder roll, pitch and yaw, elbow and wrist roll, pitch and yaw) and 6 for each leg (hip roll, pitch and yaw, knee and ankle roll and pitch). In addition to these joints we added 1 joint for the eye tilt, 2 for the pan of left and right eyes and 2 virtual joints for the vergence and the version, respectively computed as the difference and the mean between the pans.

The available sensors in the model are two 320X240@60Hz cameras for the eyes and encoders for each joint. In order to control each joint a gazebo plugin was written using the Gazebo API. Each joint is controlled by two PID, one for positions and one for velocities, both implemented using the Gazebo API library functions. The plugin publish the encoders values to a ROS topic and subscribe to other ROS topics to receive position and velocity commands. The cameras were implemented using the standard Gazebo camera plugin.

During the experiments we controlled the eye version in position. This virtual joint is directly connected with the eye pan real joints. The gains used for the velocity PID controller of each eye pan joint were the following:

$$P = 2.0, \quad I = 0.1, \quad D = 0.003$$

IV. VISUAL TRACKING MODEL

In order to test the Neurorobotics Platform, a simple visual tracking task experiment was chosen. In such a task, information about the target position is extracted from the

[2]The original iCub model can be found at https://github.com/robotology-playground/icub-gazebo

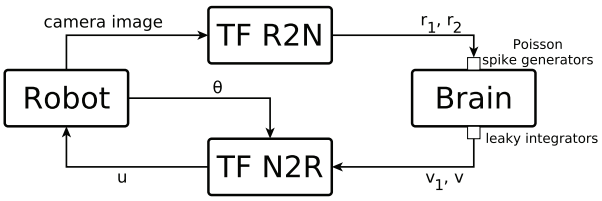

Fig. 3. Tracking architecture as implemented inside the NRP.

camera image and then sent to a controller which generates appropriate motor commands to move the gaze towards the target.

To implement such a model in the Neurorobotics Platform, the target extraction must be implemented as Robot to Neuron transfer function, while the controller must be implemented as a brain model plus an appropriate Neuron to Robot transfer function (Figure 3). One possible implementation is described in the rest of the section.

A. Robot to Neuron transfer function

In this transfer function the sensory data coming from the robot is translated into an input for the brain model: in this case the input is the camera image. From this image, the target position is extracted via a simple colour filtering. Then, only information about the fact that the object is in the left or right side of the image is sent to the brain. In order to do this, the angle of the object in the camera reference frame is flattened into this kind of information with a sigmoidal logistic function:

$$r = \frac{1}{1 + e^{\alpha}} \tag{1}$$

where α is the angle of the object in the camera reference frame and r is closer to 0 the more the object is on the left and closer to 1 the more the object is on the right.

This information is sent to the brain by using two Poisson spike generators which rates are computed as follows:

$$r_1 = 1000 \cdot r \tag{2}$$

$$r_2 = 1000 \cdot (1 - r) \tag{3}$$

B. Brain Model

The brain model was implemented as a network of 8 adaptive exponential integrate and fire neurons [18], connected as depicted in Figure 4.

Neurons 0 to 3 receive input from the first spike generator coming from the robot to neuron transfer function, while only neuron 4 receives input from the second spike generator.

The neurons labelled as 6 and 7 are the output neurons that will send the output to the neuron to robot transfer function. Given the network configuration, these neurons codify the direction towards which the eye should move. The parameters for the neurons are given in Table I. If not specified, we use the standard values set by the PyNN abstraction layer.

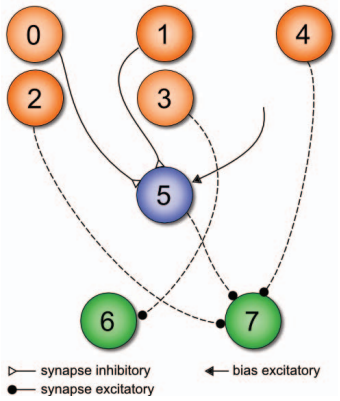

Fig. 4. Brain network. Neurons 0 to 4 are input neurons, while 6 and 7 are output neurons.

Name	Value	Unit	Description
Cm	0.025	nF	Capacity of the membrane
tau_refrac	10.0	ms	Duration of refractory period
v_spike	0.0	mV	Spike detection threshold
v_reset	-60.5	mV	Reset value for membrane potential after a spike
v_rest	-60.5	mV	Resting membrane potential
tau_m	10.0	ms	Membrane time constant
i_offset	0.0	nA	Offset current
a	0.0	nS	Subthreshold adaptation conductance
b	0.0	nA	Spike-triggered adaptation
$delta\_T$	0.0	mV	Slope factor
tau_w	10.0	ms	Adaptation time constant
v_thresh	-60.0	mV	Spike initiation threshold
e_rev_E	0.0	mV	Excitatory reversal potential
tau_syn_E	2.5	ms	Rise time of excitatory synaptic conductance
e_rev_I	-75.0	mV	Inhibitory reversal potential
tau_syn_I	2.5	ms	Rise time of the inhibitory synaptic conductance

TABLE I

PARAMETERS FOR THE NEURONS IN THE BRAIN MODEL.

The synaptic weights of the different connections were hand-tuned to produce the best experiment results.

C. Neuron to Robot transfer function

The output neurons are connected to two leaky integrators, implemented with inhibited integrate and fire neurons. Thus, the input of this transfer function are the two membrane potential of said neurons, v_1 and v_2.

The motor command given to the robot is then computed as a sum between the current eye position and a displacement given by a function of v_1 and v_2:

$$u = \theta + f(v_1, v_2) \tag{4}$$

where θ is the eye joint angle and f is defined as

$$f(v_1, v_2) = k - 2 \cdot \frac{v_2 - v_1 + 0.03}{0.09} \cdot k \tag{5}$$

where k is a parameter defining the absolute maximum displacement.

During simulations, a value of $k = 0.5$ was used. The value of u is then sent to the robot as a position motor command for eye vergence.

V. RESULTS

A. Step response

In this experiments, the controller had to reach a static target from a starting position (eye version equal to 0). Figure 5 shows the eye motion needed to reach a static target positioned on the left side (Figure 5 A,B,C) and on the right side (Figure 5 D,E,F) of the camera image, at a rotation distance of 9,14 and 25 degrees (eye rotation needed in order to centre the target in the camera image). It can be observed that the time needed to reach the target is dependent on the motion amplitude (less than 1 second for 9 degrees and almost 2 seconds for 25 degrees motion). In addition it should be noticed that the performances of the proposed controller do not depend on the motion side (the time needed to reach the target on the left and on the right is not statistically different). The mean square error (comparing eye position and target position) in the last second of the trial ranges from 0.06 for the left motion with an amplitude of 14 degrees to 0.14 of the left motion with an amplitude of 9 degrees. There is no relationship between the mean squared error and the motion direction or the motion amplitude.

B. Visual pursuit

In this experiments, the robot had to follow a target displayed on a screen moving with a sinusoidal motion (see Figure 6). The performances of the model were tested at various frequencies (from 0.1 to 0.3 Hz), with an amplitude of 18 degrees. It can be observed that the model is able to pursue the target. Figures 7 and 8 show the pursuit of the target, for oscillation frequencies of 0.2Hz and 0.3Hz, by comparing the target and current eye horizontal positions in time. In addition, the robot is able to follow the target at 0.2Hz with just a delay but without any reduction in amplitude (the eye motion amplitude compared to the target one), while at 0.3Hz the performance deteriorates. The mean square error increases from 0.36 at 0.1Hz to 0.46 at 0.2Hz and 0.72 at 0.3Hz.

VI. CONCLUSIONS

This paper presented the Neuro-Robotics Platform (NRP) which for the first time enables, as a generic framework, a coupled simulation of brain-based controllers and robots. Two experiments have been performed in the NRP, using spiking neural networks to realize a simple brain-based control behaviour and physical simulation approach for the iCub robot. The presented implementation provides an initial validation for the NRP and at the same time is a starting point from which more complex brain models may be implemented and tested as a new kind of robotic controllers.

This proof-of-concept also shows the applicability of the NRP and that the coupling of both simulations is able to foster neurorobotics and neuroscientific research.

In future work, we will conduct accurate and exact evaluation of the physic and neural simulations. Computation on high-performance computing clusters and neuromorphic hardware will be integrated for simulation of large SNN.

ACKNOWLEDGMENTS

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 Italian Ministry of Foreign Affairs and International Cooperation DGSP-UST (Direzione Generale per la promozione del Sistema Paese - Unitá per la cooperazione Scientifica e Tecnologica bilaterale e multilaterale) for the support through Joint Laboratory on Biorobotics Engineering project.

REFERENCES

[1] W. Maass, "Networks of spiking neurons: the third generation of neural network models," *Neural networks*, vol. 10, no. 9, pp. 1659–1671, 1997.

[2] S. Tolu, M. Vanegas, J. A. Garrido, N. R. Luque, and E. Ros, "Adaptive and predictive control of a simulated robot arm," *International journal of neural systems*, vol. 23, no. 03, p. 1350010, 2013.

[3] T. Shibata, S. Vijayakumar, J. Conradt, and S. Schaal, "Biomimetic oculomotor control," *Adaptive Behavior*, vol. 9, no. 3-4, pp. 189–207, 2001.

[4] E. Falotico, D. Zambrano, G. G. Muscolo, L. Marazzato, P. Dario, and C. Laschi, "Implementation of a bio-inspired visual tracking model on the icub robot," in *Proc. 19th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN'10)*. IEEE, 2010, pp. 564–569.

[5] D. Zambrano, E. Falotico, L. Manfredi, and C. Laschi, "A model of the smooth pursuit eye movement with prediction and learning," *Applied Bionics and Biomechanics*, vol. 7, no. 2, pp. 109–118, 2010.

[6] E. Falotico, M. Taiana, D. Zambrano, A. Bernardino, J. Santos-Victor, P. Dario, and C. Laschi, "Predictive tracking across occlusions in the icub robot," in *Proceedings of the 9th IEEE-RAS International Conference on Humanoid Robots (Humanoids 2009)*. IEEE, 2009, pp. 486–491.

[7] L. Vannucci, N. Cauli, E. Falotico, A. Bernardino, and C. Laschi, "Adaptive visual pursuit involving eye-head coordination and prediction of the target motion," in *Proceedings of the 14th IEEE-RAS International Conference on Humanoid Robots (Humanoids 2014)*. IEEE, 2014, pp. 541–546.

[8] E. Falotico, L. Vannucci, N. Di Lecce, P. Dario, and C. Laschi, "A bio-inspired model of visual pursuit combining feedback and predictive control for a humanoid robot," in *Advanced Robotics (ICAR), 2015 17th International Conference on*. IEEE, 2015.

[9] L. Vannucci, E. Falotico, N. Di Lecce, P. Dario, and C. Laschi, "Integrating feedback and predictive control in a bio-inspired model of visual pursuit implemented on a humanoid robot," in *Biomimetic and Biohybrid Systems*. Springer, 2015.

[10] M.-O. Gewaltig and M. Diesmann, "Nest (neural simulation tool)," *Scholarpedia*, vol. 2, no. 4, p. 1430, 2007.

[11] A. P. Davison, D. Brderle, J. M. Eppler, J. Kremkow, E. Muller, D. A. Pecevski, L. Perrinet, and P. Yger, "PyNN: a common interface for neuronal network simulators," *Front. Neuroinform.*, 2008.

[12] M. M. Khan, D. R. Lester, L. A. Plana, A. Rast, X. Jin, E. Painkras, and S. B. Furber, "SpiNNaker: mapping neural networks onto a massively-parallel chip multiprocessor," in *Neural Networks, 2008. IJCNN 2008.(IEEE World Congress on Computational Intelligence). IEEE International Joint Conference on*. IEEE, 2008, pp. 2849–2856.

[13] N. Koenig and A. Howard, "Design and use paradigms for gazebo, an open-source multi-robot simulator," in *Intelligent Robots and Systems, 2004.(IROS 2004). Proceedings. 2004 IEEE/RSJ International Conference on*, vol. 3. IEEE, 2004, pp. 2149–2154.

[14] E. Drumwright and al., "Extending open dynamics engine for robotics simulation," in *Simulation, Modeling, and Programming for Autonomous Robots*, ser. Lecture Notes in Computer Science. Springer, 2010, vol. 6472, pp. 38–50.

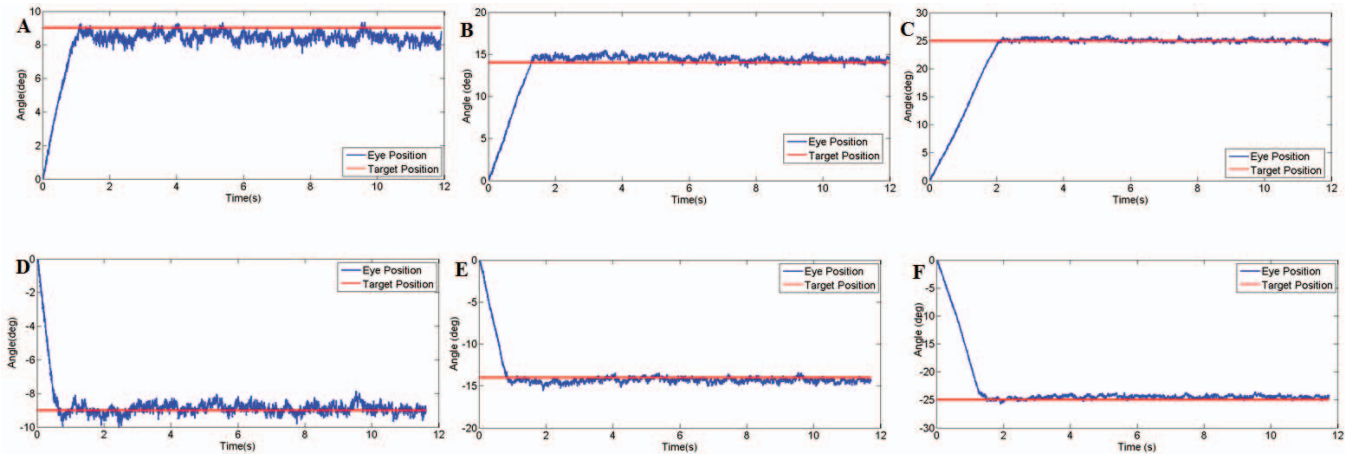

Fig. 5. Controller performance in the step response (target on the left (A,B,C) and target on the right (D,E,F). The considered target positions are: 9 degrees (A,D), 14 degrees (B,E), 25 degrees (C,F).

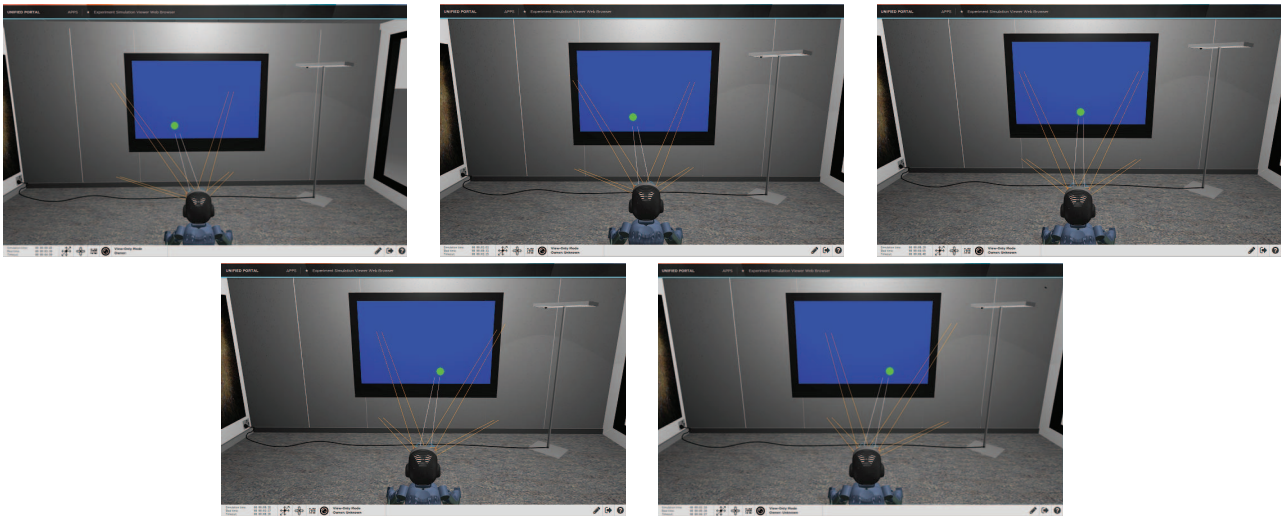

Fig. 6. Snapshots of the iCub performing visual pursuit inside the Neurorobotics Platform.

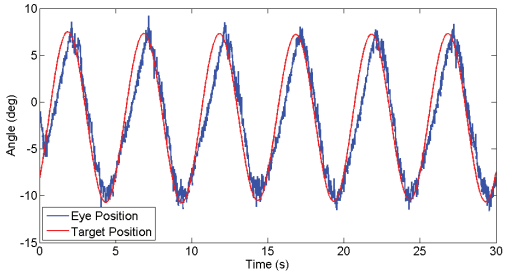

Fig. 7. Target and eye alongside the x axis during an oscillation frequency of 0.2Hz.

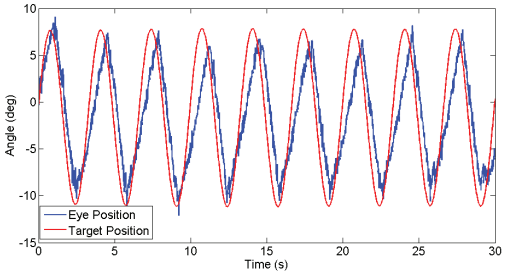

Fig. 8. Target and eye alongside the x axis during an oscillation frequency of 0.3Hz.

[15] E. Coumans *et al.*, "Bullet physics library," *Open source: bullet-physics.org*, 2013.
[16] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, "ROS: an open-source robot operating system," in *ICRA workshop on open source software*, vol. 3, no. 3.2, 2009, p. 5.
[17] G. Hinkel, H. Groenda, L. Vannucci, D. O., N. Cauli, and S. Ulbrich, "A domain-specific language (DSL) for integrating neuronal networks in robot control," in *International Workshop on Model-driven Engineering for Robotics (MORSE 2015)*, 2015.
[18] R. Brette and W. Gerstner, "Adaptive exponential integrate-and-fire model as an effective description of neuronal activity," *Journal of Neurophysiology*, vol. 94, no. 5, pp. 3637–3642, 2005.