SP10 Neurorobotics Platform: Results for SGA2 Year 2 (D10.5.2 – SGA2)

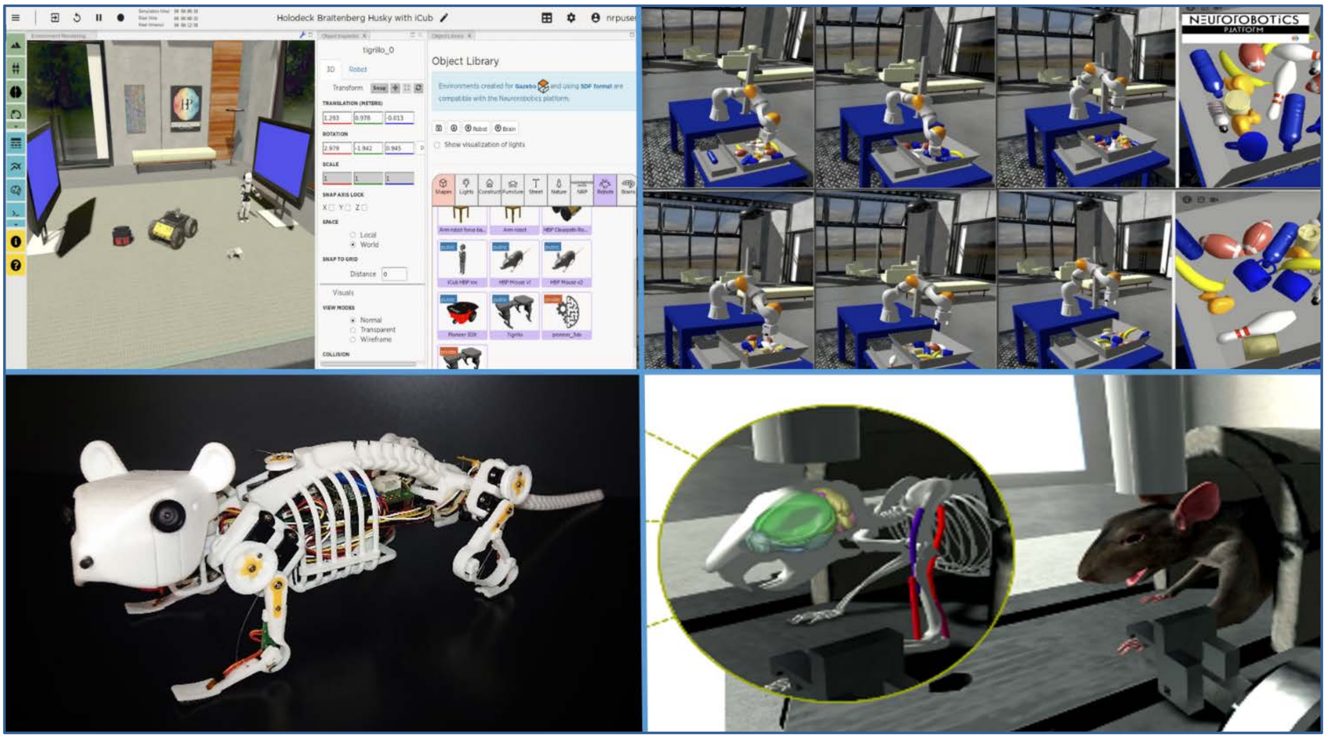

Figure 1: The Neurorobotics Platform as enabling technology for neuroscience, AI and robotics

The Neurorobotics Platform supports *in silico* exploration of multiple scientific questions in neuroscience, robotics and embodied AI. It enables users to investigate the synergies between musculoskeletal systems and motor control, leverage HPC infrastructure for parallelised / distributed learning, and explore sim-to-real transfer learning to robotic platforms such as the HBP robot rodent NeRmo.

Project Number:	785907	Project Title:	Human Brain Project SGA2

Document Title:	SP10 Neurorobotics Platform: Results for SGA2 Year 2
Document Filename:	D10.5.2 (D67.2 D48) SGA2 M24 ACCEPTED 200731.docx
Deliverable Number:	SGA2 D10.5.2 (D67.2, D48)
Deliverable Type:	Report
Work Packages:	WP10.1, WP10.2, WP10.3, WP10.4, WP10.5
Key Result(s):	KR10.1, KR10.2, KR10.3, KR10.4, KR10.5
Dissemination Level:	PU = Public
Planned Delivery Date:	SGA2 M24 / 31 Mar 2020
Actual Delivery Date:	SGA2 M24 / 03 Apr 2020, Accepted 31 Jul 2020
Author(s):	Alois KNOLL, TUM (P56), Egidio FALOTICO, SSSA (P49)
Compiled by:	Fabrice MORIN, TUM (P56), Alma SERICA, SSSA (P49)
Contributor(s):	Egidio FALOTICO, SSSA (P49), KR10.1 Axel VON ARNIM, FORTISS (P21), KR10.4 Fabrice MORIN, TUM (P56), Introduction, KR10.2, KR10.3 Jörg CONRADT, KTH (P39), KR10.5
SciTechCoord Review:	Mehdi SNENE, EPFL (P1)
Editorial Review:	Guy WILLIS, EPFL (P1)
Description in GA:	Summary of SP Results for M12-24, broken down by Task: • Actual vs. planned results (due & done/due & not done) • Results passed to another Task, WP or SP (due & done/due & not done) • Milestones due and achieved, how validated, by whom. • Milestones due but not achieved, why, impact, corrective action • Publications • SP Roadmap for SGA3
Abstract:	This Deliverable is the annual compound of HBP deliveries and results (outputs and outcomes) from Subproject SP10 – Neurorobotics Platform (NRP). The main technical and scientific deliveries from April 2019 to March 2020 for SP10 were: • *In silico* experiments with realistic spinal circuits and the musculoskeletal rodent model as co-design drivers (KR10.1). • The latest release of the robot rodent with extended sensing capabilities and a new mechanical design for improved performance (KR10.2).

	• The Integrated Behavioural Architecture: and integrative software framework to compose cognitive architectures from heterogeneous functional components on the NRP(KR10.3). • A series of new functional features and improvements to the NRP delivered in Releases 2.3 and 3.0 (KR10.4). • The continued development of compliant robotics and demonstration of what the NRP can offer in terms of knowledge transfer to such physical robots (KR10.5)
Keywords:	Neurorobotics, virtual robotics, *in silico* experiments, neural control of movement
Target Users/Readers:	Computational neuroscience community, Robotics community, consortium members, funders, Neuroscience community, neuroscientists working on neural control of movement, platform users, researchers, scientific community

NOTE:

Components: a full list of Components will be included in the SGA2 Periodic Report.

Dissemination: dissemination actions to promote specific Key Results and the Outputs that contribute to them will be documented in the SGA2 Periodic Report.

Table of Contents

Table of Tables

Table of Figures

1. Overview

Neurorobotics is an emerging research field where key concepts and technologies from both brain science and robotics are fused in order to: 1) provide new experimental paradigms in brain simulation; and 2) produce new technological solutions in artificial intelligence and robotics. The first objective relies on the concept of embodiment (i.e. placing the brain inside a body and simulating them both) for implementing closed-loop experiments, where brain activity is driven by streams of sensory stimuli and interactions with the environment. The second objective relies on identifying features of the brain as an information-processing system that would provide digital systems (including robots) with functional capabilities that are currently beyond the state of the art (situational awareness, decision-making capability, etc.).

HBP Subproject 10 (SP10) aims to establish neurorobotics and closed-loop embodied simulations as a new paradigm in neuroscience research. To that end, SP10 is building and operating the Neurorobotics Platform (NRP), a research infrastructure that provides a set of tools and workflows for designing and simulating complex models of cognitive architectures, robots and physically realistic environments. Through the NRP, researchers from within and outside the HBP can define, run and share experiments and embodied simulations.

The NRP is thus a unique tool that serves as a common ground on which neuroscientists and roboticists can collaborate. The former can evaluate models, ranging from simple sensorimotor models to large-scale behavioural architectures, in the context of behavioural tasks. The latter can control complex robot bodies with many degrees of freedom with brain-inspired controllers, and leverage lessons from neuroscience to endow robots with abilities that are currently beyond the state of the art (e.g. adaptability to unforeseen changes in task parameters, etc.). Through co-design activities linking research in neuroscience and software development, SP10 also strives to provide the research community with tools that support continuous integration of new data and models in a standardised and collaborative manner.

The present document provides a high-level summary of the Key Results (as defined in the Grant Agreement) from SP10.

2. Introduction

The present document provides a high-level summary of the scientific and technical activities carried out by the SP10 Partners in the second year of SGA2. It is structured around the Key Results defined in the Grant Agreement, and frames the developments in terms of their contribution to the overarching objectives of both the HBP and SP10.

The **first Key Result** is the continued development and expanded use of the virtual rodent model for *in silico* behavioural experiments (KR10.1). This detailed body model is available on the NRP. It is based upon a realistic musculoskeletal model and enables simulation of behavioural tasks involving rodents. It was leveraged inside SP10 in two different types of experiments and demonstrators. The first demonstrator focused on motor control in rodents, in the context of a pulling task. This experiment was originally conceived to study motor learning before and after a topical stroke in the motor cortex. The second type of experiments focused on modelling locomotion in rodents (as a prelude to modelling in humans) in the context of spinal cord injury treatment. Several high-profile papers were published during SGA2 (e.g. Formento *et al*. (2018), Nature Neuroscience, 21, 1728–1741 – P1623; Wagner *et al*. (2018), Nature, Vol. 563, No. 7729 – P1622) and multiple follow-up projects are now running that directly result from this approach, thereby illustrating its success and relevance.

The **second Key Result** is the physical rodent robot, a lightweight technology platform which uses mechanically compliant structural elements and is designed to approximate rodent locomotion patterns. It is intended to be cheap enough to be shared between HBP Partners as a means to transfer neuronal models into a common physical embodiment controlled with neuromorphic hardware. Several versions of this rodent robot were released in SGA2. The latest version not only provides additional degrees of freedom and additional sensing capabilities, it was also vastly redesigned so as to be more robust and easier to produce.

The **third Key Result** regroups the results of several activities carried out to enable users to run complex multi-component cognitive architectures on the NRP. The central component in this endeavour is the Integrated Behavioural Architecture (IBA), a modular software framework enabling compositionality of heterogeneous models of brain functions into a single cognitive architecture running on the NRP. The IBA provides a way for individual scientists to easily plug-in their own code into NRP simulations, in order to perform experiments on complex cognitive tasks. To make their task even easier, several functional modules were developed and made available (e.g. visual module, motor primitive module, cerebellum module and motor exploration module); demonstrators were then created to showcase how these can be combined within the IBA.

The **fourth Key Result** is the improved NRP itself. In the second year of SGA2, the software development activities inside SP10 focused essentially on improving the connection of the NRP to other tools and infrastructures in the HBP. The various new features introduced are described, as well as the rationale for their implementation.

Finally, the **fifth and last Key Result** is modular control for physical robots under real-time constraints. While simulation can indeed guide robotic development, especially when combined with learning processes (inspired by either biology or AI), the intrinsic limitations of the physics engines available in the NRP create an unavoidable reality gap that needs to be characterised. We thus address manipulation of objects with complex inertial properties (e.g. a half-full water bottle) with a compliant / soft robot as a test case for motor learning and adaptation, comparing simulations on the NRP to “real-world” experimental results.

3. Key Result KR10.1: Virtual rodent model for *in silico* behaviour experiments

3.1 Outputs

3.1.1 Overview of Outputs

3.1.1.1 List of Outputs contributing to this KR

- Output 1: Virtual mouse and motor control (C2596, C2603, C2612, C2603, C2609, C2610)
- Output 2: Virtual stroke rehabilitation experiment (C2614)
- Output 3: Deconstruction of spinal circuits engaged by epidural electrical stimulation that restore locomotion after paralysis (C2607)

3.1.1.2 How Outputs relate to each other and the Key Result

Output 1 provides all the necessary building blocks for the creation of different virtual behavioural experiments relative to motor tasks including biological neural networks. Such components were used to perform a simulated stroke rehabilitation experiment (Output 2). Output 3, relying on rat experiments, provides a better understanding of locomotion mechanisms.

3.1.2 Output 1: Virtual mouse and motor control

This output demonstrates the virtual mouse model and the motor control applied in the locomotion and the pulling tasks. It involves five different sub-components.

1) Skeletal model. The mouse skeletal model was obtained from a CT-scan, and is fully rigged with the necessary degrees of freedom between any two links. The joint centres are anatomically relevant. In the current version, all rotations are limited to simple revolute joints. If necessary, the user may define more complex joint rotations using the OpenSim Application Programming Interface (API), provided in the NRP. The physical properties of the links are computed based on simple bounding objects which enclose any given link with a minimum area. The masses and inertias of each link (Figure 2) are then computed with the assumption that each of the bounding boxes has a uniform distribution of water density. The total computed mass of the full model with this assumption (~25g) is in accordance with the average mass of actual mice. The user is still free to update the mass of any link with more accurate data from actual measurements.

Figure 2: Fully articulated skeletal model of the mice in its neutral pose.

The orange boxes represent computed inertia of the individual segments.

2) Muscle models. Two different types of muscle models are supported as part of the locomotion template: users can choose between the Hill-type model and a simpler spring-damper muscle model. Integration of the OpenSim with the NRP allows for the direct use of standard Hill-muscle implementations. The muscle attachment points had to be recomputed for the NRP model from the literature data. Figure 3 shows the transferred muscle attachments of the hind limb visualised in Blender.

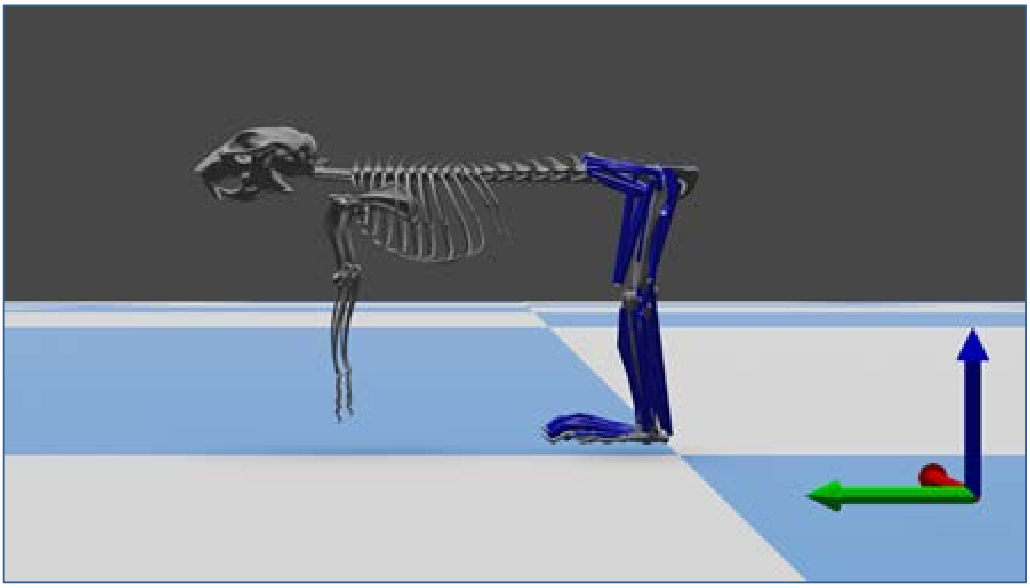

Figure 3: Reduced model for locomotion with hind-limb Hill-type muscles

3) Neural networks for locomotion. A neural network library has been developed. The library has a high-level Python interface and the neuronal models themselves are developed in Cython for efficiency. Figure 4 shows an example of a network setup in the locomotion template. The user has several options to extend the network complexity or even completely rewrite the network from scratch.

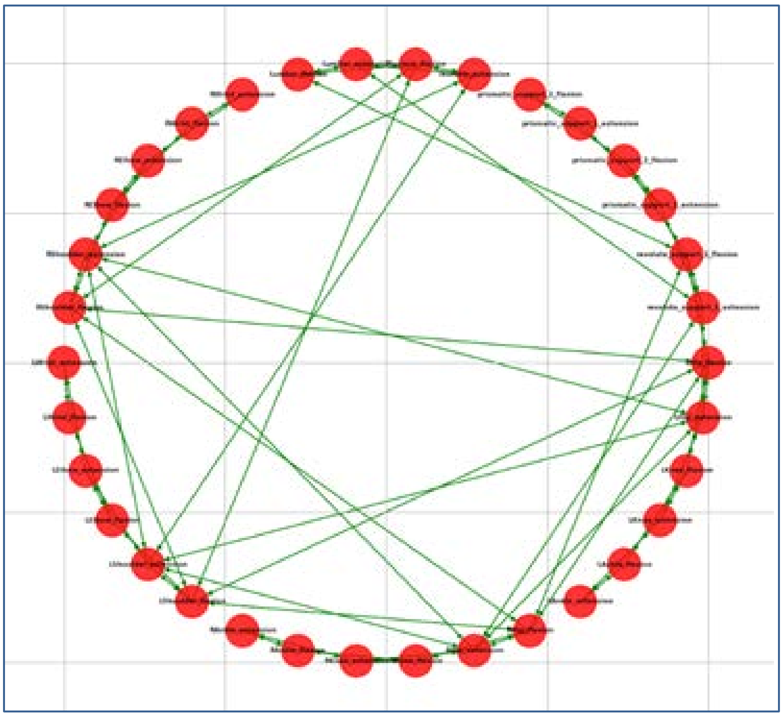

Figure 4: Central Pattern Generator network used to generate locomotion patterns

4) Neural networks for pulling task. A spinal cord model capable of actuating the simulated muscles was developed and implemented as a spiking neural network in NEST. The spinal cord comprises a circuit for a single muscle, inhibitory connections between antagonistic pair of muscles and interneurons to modulate descending stimuli (see Figure 2 in Deliverable D10.5.1 SGA2 M12).

A brain model capable of generating realistic pulling motions was developed by implementing a spiking functional model of relevant cortical areas. The two main motor areas modelled are the rostral forelimb area (RFA) and the caudal forelimb area (CFA), which act as premotor and motor cortices, respectively. The RFA is directly modelled as a population reproducing neurophysiological recordings from *in-vivo* experiments, which is connected to the CFA network (Figure 5). More details on this experiment are available in Deliverable D10.1.1 SGA2 M24.

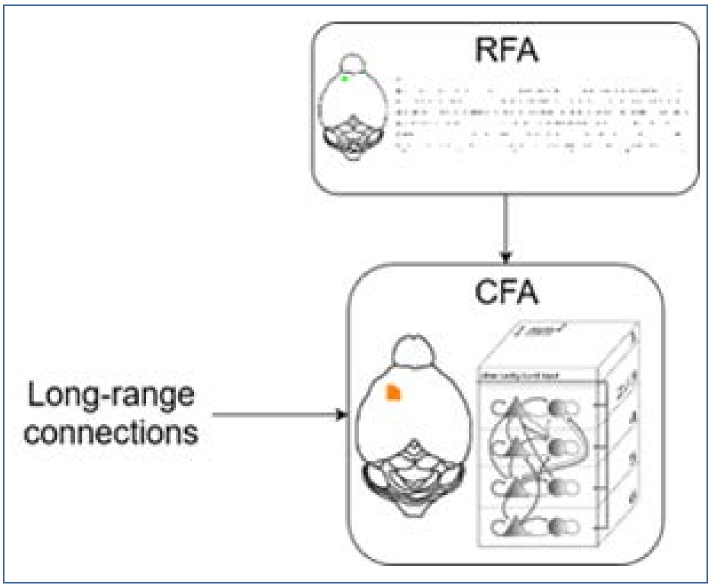

Figure 5: Spiking cortical model, including premotor (RFA) and motor areas (CFA).

Table 1: Output 1 links

Component	Link to	URL
C2596	Data/model Repository	https://gitlab.com/sssa-humanoid-robotics/NeuralModels
	Technical Documentation	https://gitlab.com/sssa-humanoid-robotics/NeuralModels
	User Documentation	https://gitlab.com/sssa-humanoid-robotics/NeuralModels
C2612	Data/model Repository	https://bitbucket.org/lore_ucci/cortex-model/
	Technical Documentation	https://bitbucket.org/lore_ucci/cortex-model/
	User Documentation	https://bitbucket.org/lore_ucci/cortex-model/

3.1.3 Output 2: Virtual stroke rehabilitation experiment

To simulate the experimental trial described above, we combined the following simulated components: the M-Platform, the mouse musculoskeletal forelimb, the spinal cord model for the forelimb muscles and the cortical brain model. All of them were included in the NRP to create a simulated post-stroke rehabilitation experiment.

Alongside the musculoskeletal embodiment described above, we modelled the M-Platform. The slide mechanism was modelled as a prismatic joint, actuated by a PID controller. A control mechanism using a state machine for automatic reset of the sled position was developed, thanks to functionalities already present in the NRP. In the *in-vivo* experiment, a certain threshold of force is needed to move the slide due to friction. In the simulation, we set a muscle activation threshold that, when the subject makes a reaching/pulling action, forces the slide control mechanism to deactivate the PID controller, effectively freeing the slide and allowing the mouse forelimb to carry out the action.

In a first set of tests, only a simplified cortical model was employed. This model consisted of a set of static spike generators reproducing the events detected with the MU detection in the RFA. This was done to test whether it was possible to achieve pre- and post-stroke results by simulating a cortical area that is not directly affected by the stroke. Results, shown in Figure 6, demonstrated that it was possible to simulate the activity of healthy mice, but not of post-stroke mice. More details on this experiment are provided in Deliverable D10.1.1 SGA2 M24.

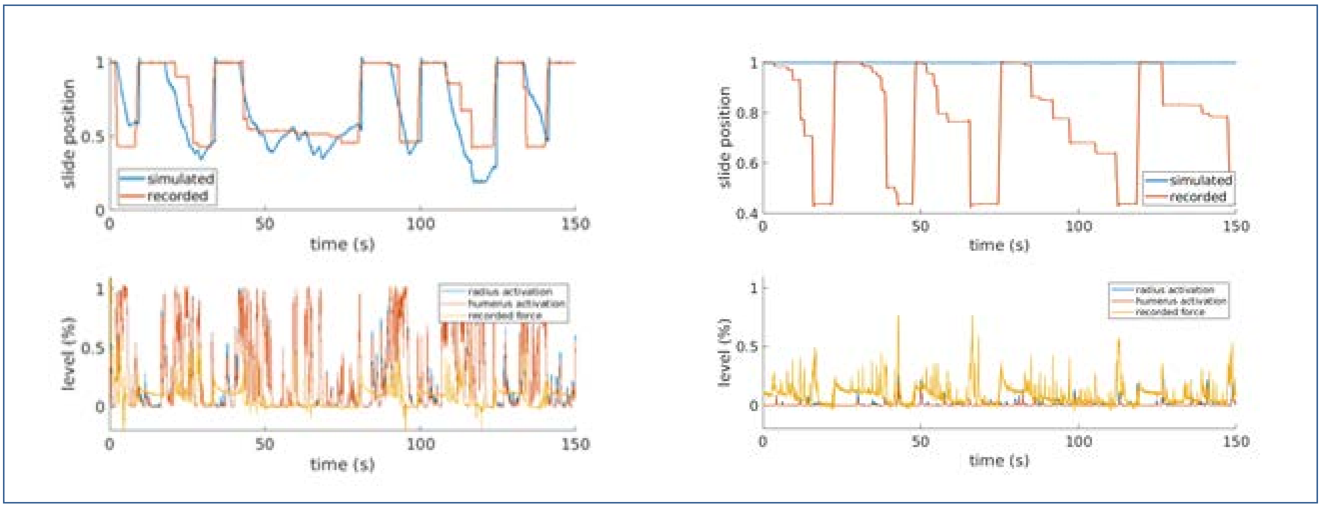

Figure 6: Comparison between experiments and simulation for healthy and stroke animals

Results for the simplified cortical model show that, by simulating only the RFA, it is possible to reproduce data from healthy mice (left), but not of stroke mice (right).

Table 2: Output 2 links

Component	Link to	URL
C2614	Data/model Repository	https://gitlab.com/lore.ucci/closed-loop-mouse-stroke-simulation
	Technical Documentation	https://gitlab.com/lore.ucci/closed-loop-mouse-stroke-simulation
	User Documentation	https://gitlab.com/lore.ucci/closed-loop-mouse-stroke-simulation

3.1.4 Output 3: Deconstruction of spinal circuits engaged by EES

Epidural electrical stimulation (EES) of the lumbosacral spinal cord can restore locomotion in patients with paralysis. However, the underlying mechanisms remain enigmatic. We carried out hybrid computational simulations, combining 3D Finite Element Models (FEM) with compartmental cable models of the lumbosacral spinal cord and neural circuitry model of locomotor circuitry with biomechanical models of the rodent hind limbs. We demonstrated that EES primarily activates proprioceptive and cutaneous low-threshold mechanoreceptor (Aß-LTMR) afferents at amplitudes within the therapeutic range. Moreover, we uncovered that the selective recruitment of proprioceptive feedback circuits enables motor pattern formation. However, the non-physiological recruitment of Aß-LTMR feedback circuits disrupts light touch information encoded in Aß-LTMR afferents and transmitted downstream through glutamatergic interneurons expressing the RORαnuclear orphan receptor. Our simulations (Figure 7) thus uncovered a previously unidentified neural circuitry of Aß-LTMR feedback circuits, which we exposed anatomically using intersectional virus-based tracing experiments in rodents. This new understanding of the mechanisms underlying EES guided the design of a targeted noradrenergic pharmacotherapy that immediately enabled robust locomotion in paralysed rats.

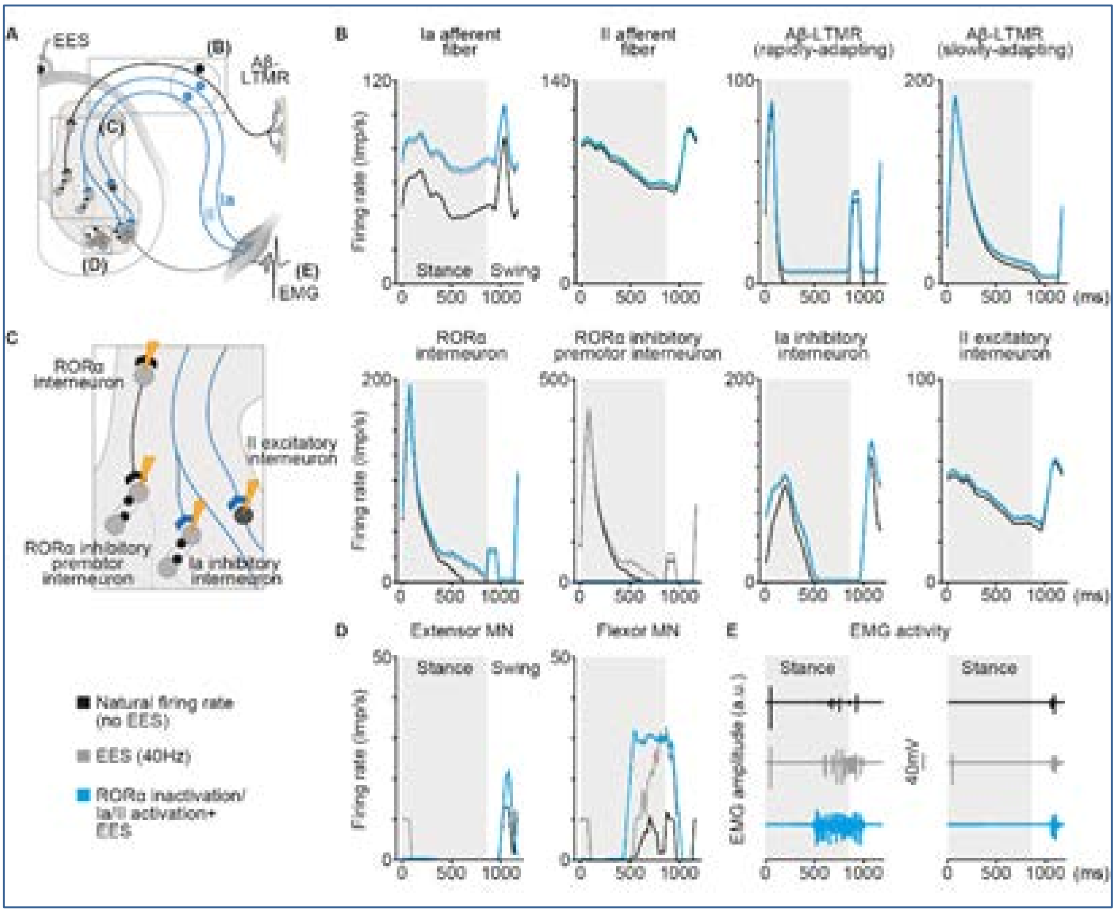

Figure 7: *In-silico* electrophysiology of cell populations during EES-enabled locomotion in rodents

(A) Sketch of the identified neural circuitry. (B) Firing rates of proprioceptive and cutaneous low-threshold mechanoreceptors. (C) Firing rates of interneuron cell populations. (D) Firing rates of motoneuron pools. (E) Estimated EMG activity.

3.2 Validation and Impact

3.2.1 Actual and Potential Use of Output(s)

These biologically realistic models (Output 1 and 2) will advance our knowledge of sensorimotor integration by testing neuroscientific theories through embodiment in closed-loop simulations. Experimenters will also benefit from the detailed model of the mouse and simulations, by being able to test different experimental conditions before performing a real experiment, thus saving time in the experimentation process. Output 3 may provide new inputs for restoring locomotion in humans.

3.2.2 Publications

- [P2374] Vannucci, L., Pasquini, M., Spalletti, C., Caleo, M., Micera, S., Laschi, C. & Falotico, E. (2019). Towards in-silico robotic post-stroke rehabilitation for mice. *In press*: will be published in the Proceedings of the 2019 IEEE International Conference on Cyborg and Bionic Systems (CBS2019) whose publication has been delayed because of delays at the publisher.
    - o This paper presents the *in silico* replication of the effects of stroke and the post-stroke rehabilitation experiment in rodents. It is an initial validation of the experiment simulation described in Output 2.
- [P2485] Salimi-Nezhad, N., Ilbeigi, E., Amiri, M., Falotico, E., & Laschi, C. (2019). A Digital Hardware System for Spiking Network of Tactile Afferents. Frontiers in neuroscience, 13, 1330.
    - o This paper presents models of tactile afferents and their implementation in hardware that offers a low-cost neuromorphic structure for tactile information processing.

4. Key Result KR10.2: Rodent robot

4.1 Outputs

4.1.1 Overview of Outputs

4.1.1.1 List of Outputs contributing to this KR

The following describes the final version of the Neurorobotic mouse (NeRmo) robot, as well as the research work that it has supported over the past year.

The following outputs were achieved in relation to this Key Result:

- Release of the latest (and last) Version 4.1 of NeRmo
- Study of kinematic retargeting in the context of partnering project SoRon
- Connection of NeRmo to neuromorphic hardware (HBP SP9, SpiNNaker 1 & 2)

4.1.1.2 How Outputs relate to each other and the Key Result

The first output encapsulates all the engineering work that went into improving NeRmo as a research platform, including the production of a digital twin (i.e. a simulated model, reproducing the characteristics of the real one, with as much fidelity as possible) on the NRP. Outputs 2 and 3 are built upon and made possible by Output 1.

4.1.2 Output 1: NeRmo v4.1 and its digital twin

NeRmo is a biomimetic physical robot that is modular and low-cost, and it was created with a view to mimic the locomotion of a rodent at a scale similar to its biological model. It is untethered, easy to use and simple to produce; it thus can be used as a universal research platform. It is based on tendon-driven actuation, which enables the implementation of a compliant leg and body design and thus enables adaptive and dynamic walking motions. It can be particularly useful for investigating new, efficient types of locomotion for compliant legged systems. Combined with its digital twin inside the NRP, it is a useful tool to reduce the reality gap between simulation and the real world.

The final version (Release 4.1) features multiple improvements over previous iterations, especially ease of assembly and robustness, to get a wider user group. Deliverable D10.3.1 SGA2 M24 provides an extensive technical description of this release. In summary, the legs were redesigned from the first version, to allow for a simpler manufacturing process and to be more robust to daily use. They now also include position sensors and ground pressure sensors. The head now holds two HD cameras, a button on the nose and a touch sensor on the top of the head. The two cameras are connected to the USB port of the Raspberry Pi via a USB hub also located within the Head. The complete robot with most sensors labelled is depicted in Figure 8. Its digital twin is depicted in Figure 9.

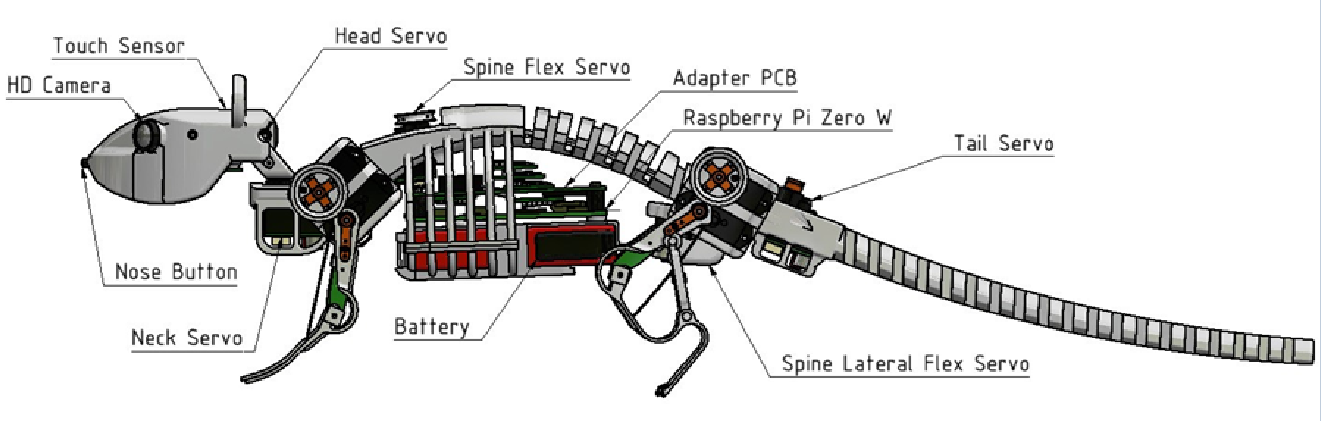

Figure 8: Side view of NeRmo robot with the major parts labelled.

Figure 9: Digital twin of NeRmo inside the NRP

Table 3: Output 1 Links

Component	Link to	URL
C2572	Model Repository	https://github.com/Luchta/nermo_robot
	Technical Documentation	https://collab.humanbrainproject.eu/#/collab/45325/nav/311404
	User Documentation	https://collab.humanbrainproject.eu/#/collab/45325/nav/311404
C2574	Software Repository	https://github.com/Luchta/nermo_code
	Technical Documentation	https://collab.humanbrainproject.eu/#/collab/45325/nav/311404
	User Documentation	https://collab.humanbrainproject.eu/#/collab/45325/nav/311404

4.1.3 Output 2: Study of kinematic retargeting

Motion capture of the robot gait was carried out for comparison to animal motions. This was done within the framework of the HBP Partnering Project SoRon, run by the RIKEN institute (Japan) and the National Institute of Applied Science and Technology (Japan). This project aims to study the so-called "retargeting" of kinematics between dissimilar but related systems (e.g. a rat and a robot rodent). The motion capture system used was a Raptor-12 (Motion Analysis Corp., USA). The experimental setup (Figure 10) was an inclination-adjustable aluminium ramp (width: 0.1 m; length: 1 m) going to a horizontal square platform (0.1 x 0.1 m) with an operant conditioning panel.

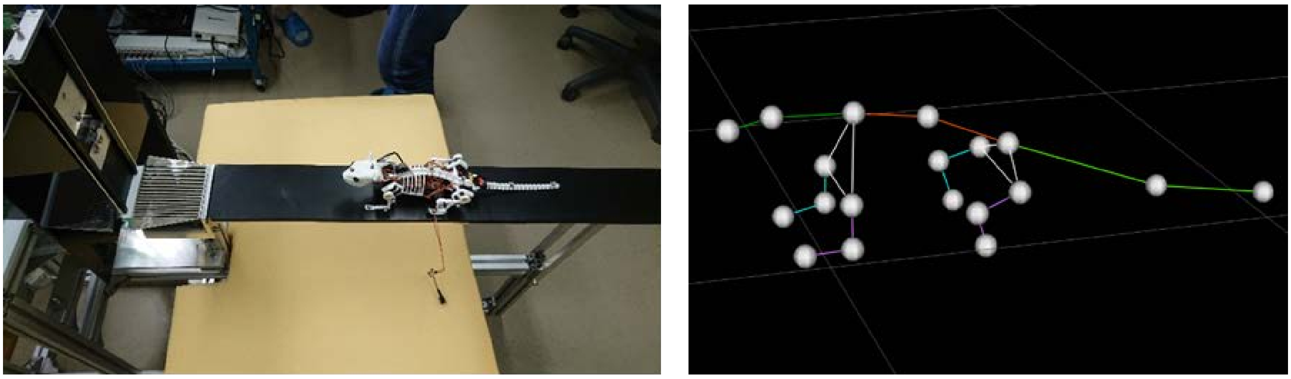

Figure 10: Experimental setup at AIST (left) and resulting marker data after labelling (right)

Multiple motions were created and captured to evaluate the robot:

- A full range sinusoidal actuation of the robot to evaluate joint control
- Multiple trotting gaits
- Multiple bounding gaits
- Task motions: actuating the levers on the conditioning panel
- A simple retargeted motion, calculated from available animal motion capture data

This served multiple purposes. First, those motions are being analysed (publication in preparation) to investigate the robot's control and motion capabilities. The same data is also used to validate the digital twin on the NRP, by calculating the error between the robot's joint angles as well as the error in whole body translation. Finally, the simple retargeted motion was used to evaluate the robustness of the retargeting method developed in RIKEN, by analysing the translation of a reference marker and by using so-called crystalized motion patterns. Preliminary results (Figure 11) show good agreement positional agreement between the trajectory of the animal and that of the robot driven by the retargeted movement.

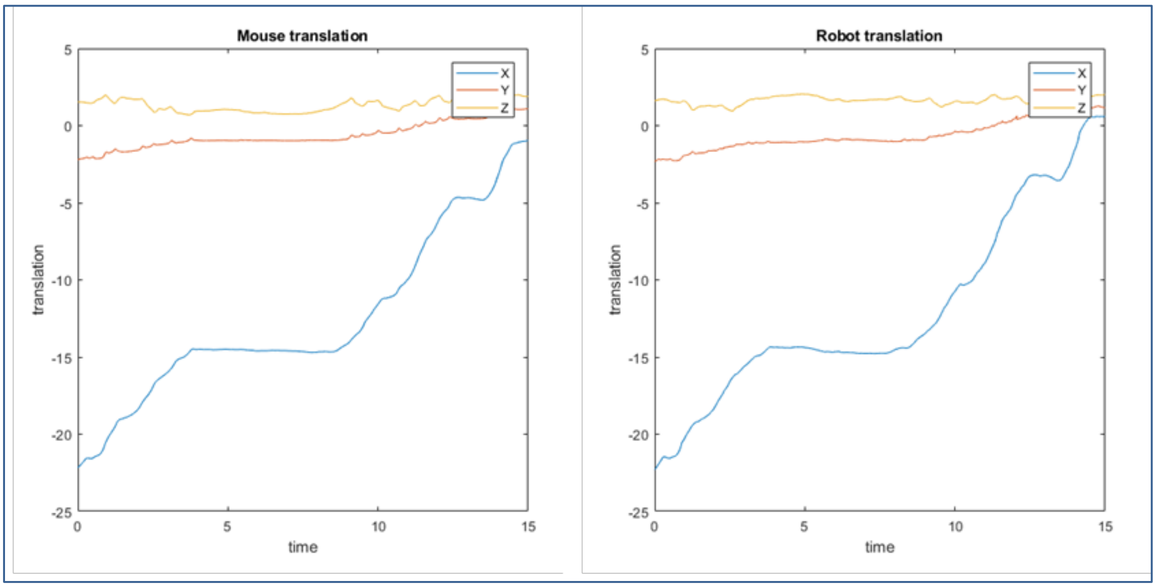

Figure 11: Trajectory of a hip reference marker on an animal (left) and on the robot (right)

The motor commands sent to the robot to generate the trajectory on the right-hand side were position commands based on the retargeted kinematics recorded on the animal during its run (trajectory on the left-hand side).

4.1.4 Output 3: Connection of NeRmo to neuromorphic hardware (SpiNNaker 1 & 2)

The default NeRmo robot platform is controlled autonomously by an on-board Raspberry Pi Zero W module. We proposed integrating stand-alone neuromorphic computing chips (developed in HBP SP9) into NeRmo as proof-of-principle of on-board real-time neuromorphic control. Integration of current SpiNNaker hardware on-board the robot was discussed with research teams in Manchester and Dresden. Based on these discussions, fully autonomous control of NeRmo with on-board neuromorphic hardware was deemed unachievable with the current version of SpiNNaker; instead, SpiNNaker 2 would be required. As such, the best course of action was to design prototypes of stand-alone boards to interface SpiNNaker chips (both generation 1 and generation 2) with NeRmo hardware.

We therefore developed these boards (Figure 12), as well as the firmware for configuring and loading such systems. In our design, a customisable microcontroller interacts with the SpiNNaker chips during boot-up and operation. It continuously converts sensory perception into SpiNNaker packets (SpiNNaker input) and it converts SpiNNaker packets (motor control output) to motor signals for NeRmo. All components can be reduced in size to allow instantiation of such a stand-alone SpiNNaker computing system on board NeRmo.

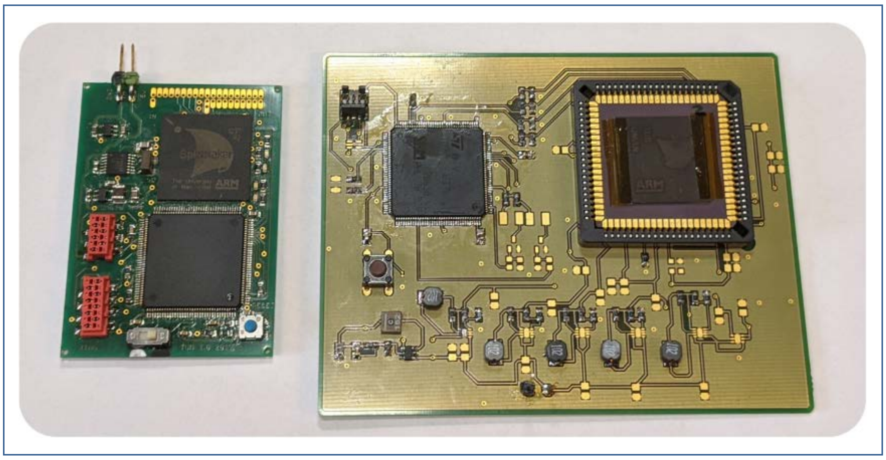

Figure 12: Stand-alone SpiNNaker computing board prototypes

The two stand-alone SpiNNaker computing board prototypes for SpiNNaker 1 (left) and SpiNNaker 2 (right, test-chip).

Independent of on-board neuromorphic control, an initial version of neuronal control of NeRmo on a desktop SpiNNaker computing board for simple motor trajectories (walking forward, turning) was demonstrated. Here, sensory perception and motor commands were exchanged between the desktop neuromorphic computer board and NeRmo through WLAN and Ethernet. Once SpiNNaker 2 (and its software stack) is developed, these existing spiking models will be converted to run on board NeRmo as a stand-alone demonstrator.

Table 4: Output 3 links

Component	Link to	URL
C2574	Data/model Repository	https://gitlab.com/neurocomputing/hbp/sga2/c2574/model
	Technical Documentation	https://collab.humanbrainproject.eu/#/collab/45325/nav/311404
	User Documentation	https://collab.humanbrainproject.eu/#/collab/45325/nav/311404

4.2 Validation and Impact

4.2.1 Actual and Potential Use of Output(s)

Released at the end of a two-year development process, the latest version of NeRmo was an exercise in finding the right balance between functionalities, ease of assembly and price. It is the most capable robot to be developed within the HBP since the start of the Project and also the most robust (for examples of locomotion, see [1] and [2]). The variety of embedded sensors coupled to the embedded compute power make it an attractive research platform.

The work carried out with our Japanese colleagues on kinematic retargeting is an interesting demonstration of the use of a physical robot to validate theoretical calculations. Despite the discontinuation of HBP funding for work on this robot in the next phase of the project (SGA3), it is clear that NeRmo still has much to offer to the neurorobotics community. Future work could focus on the use of the robot in motion-control studies and, most importantly, in exploring the reality gap between simulation and reality, by training the digital twin on the NRP and transferring the resulting controllers to the physical robot.

4.2.2 Publications

- [P2328] Lucas, P., Oota, S., Conradt, J., and Knoll, A. (2019) Development of the neurorobotic mouse. *In press*: will be published in the Proceedings of the 2019 IEEE International Conference on Cyborg and Bionic Systems (CBS2019), the publication of which was delayed because of delays at the publisher.

[1] https://collab.humanbrainproject.eu/#/collab/45325/nav/530902?state=uuid%3D3b61196c-3161-49ec-997b-09c66bfc8767
[2] https://collab.humanbrainproject.eu/#/collab/45325/nav/311406?state=uuid%3D87255824-bed0-4a45-b422-fe161f3ec1ee

5. Key Result KR10.3: Integrated Behavioural Architecture

5.1 Outputs

5.1.1 Overview of Outputs

5.1.1.1 List of Outputs contributing to this KR

The Integrated Behavioural Architecture (IBA) is a software framework, through which models of individual brain functions can be composed into a modular cognitive architecture, that can then be used in simulations on the Neurorobotics Platform (NRP). An essential objective of the IBA is to enable users to reuse components written by others and to integrate their own code into a multi-component cognitive architecture in the NRP at a reasonable cost, in terms of time and effort invested.

The Outputs linked to this Key Result are described below:

- Output 1: IBA as a software framework (C2527)
- Output 2: Library of components for the IBA and related research activities (C2526, C2582, C2942, C3052, C2525, C2580)
- Output 3: Demonstrators based on the IBA

5.1.1.2 How Outputs relate to each other and the Key Result

The IBA is the software framework that enables users to compose modular cognitive architectures on the NRP. It is the core component of KR10.3. The components produced conform to the IBA specifications and run through it on the NRP; they were combined to produce demonstrators.

5.1.2 Output 1: the IBA as a software framework

A detailed description of the IBA as a software framework is provided in D10.2.1 SGA2 M24. The following summarises the essential aspects.

With the IBA, all functional modules of a given cognitive architecture are integrated as ROS nodes. These run in parallel on different processes, communicating via ROS services. The use of ROS provides a convenient communication layer for integrating an arbitrary number of components, as well as a practical manner for achieving concurrency between all the modules (they all run in their own process). While we used Python as a coding language in the current implementation of the IBA, the use of ROS also opens the possibility to use very heterogeneous languages to code each of these modules, as long as they have either a Python or a ROS Application Programming Interface (API). Examples of supported languages or frameworks include: C++, MATLAB, C#, NEST and TensorFlow.

The IBA modules execute concurrently in separate processes at different “tick rates” (i.e. steps of heterogeneous duration in simulation time) that are user-specified. The IBA controls the timing of the communications between these processes (i.e. depending on the tick rate of each component). To achieve this in a manner that is technically manageable, while remaining sufficiently versatile and user-friendly, we impose a requirement that the time step of the Closed Loop Engine (CLE) of the NRP (i.e. the component that guarantees synchronisation between Gazebo and the brain simulation) be a power of two of the tick rate of individual modules (Figure 13-A).

From a user perspective, the synchronisation mechanism is entirely transparent. Users only need specify the tick rate of every functional module in relation the CLE time step, and the IBA then takes over. All user-provided code goes into external module files (one file per module, see Figure 13-B) with a simple structure: at no point do users need to worry about synchronising the execution and communication between different modules; instead they can fully concentrate on the computational aspect of their work.

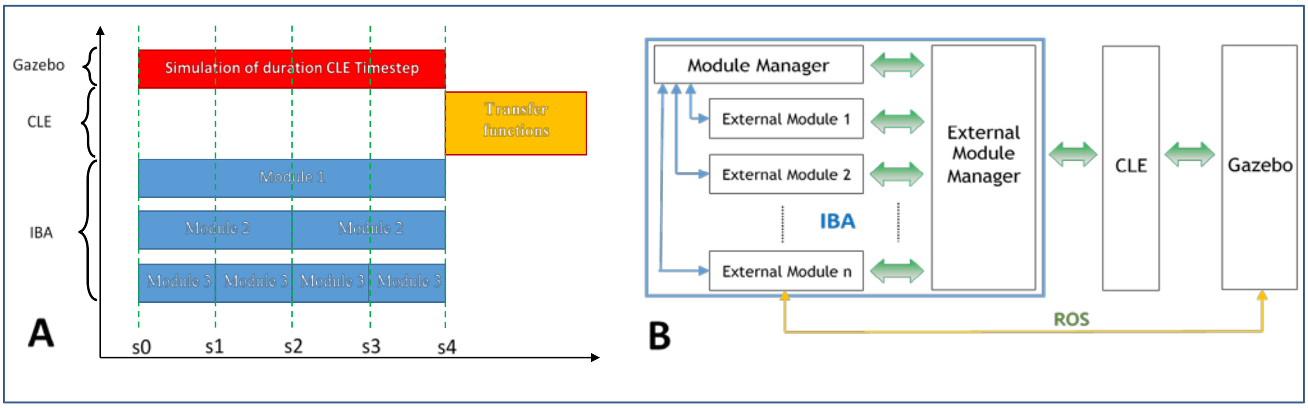

Figure 13: Execution order for one CLE step (A) and block diagram (B) of a simulation with IBA.

In the example depicted in A, three modules run at different tick rates. Each blue box in this figure represents the execution of the simulation time step of the corresponding module, which is running concurrently to all others.

Table 5: Output 1 links

Component	Link to	URL
C2527	Data Repository	https://github.com/HBPNeurorobotics/IBA
	Technical Documentation	https://collab.humanbrainproject.eu/#/collab/78544/nav/531713
	User Documentation	https://collab.humanbrainproject.eu/#/collab/78544/nav/531713

5.1.3 Output 2: Library of components for the IBA and related research activities

In the past year, functional models of various brain areas and models of brain function were developed to provide off-the-shelf component modules for IBA users. These components are:

- A motor primitive module (C2582).
- A visual module implementing predictive coding (C2526).
- A functional cerebellum model (C2942).
- A functional hippocampal module for navigation (C3052).
- A module for visuo-tactile exploration (C2525).
- A module for unsupervised exploration of mechanical resonances of the body (C2580) inspired by modulation of neuronal activity in the spinal circuitry by Raphe nuclei.

These modules only represent the visible part of the output, as all the modelling work and research activities that supported the creation of these modules is better represented through a number of scientific papers that are already published (e.g. P1909, P1503, P2243, P2343, P1526, P2057, P1934 and P2133), under review (P2129) or still in preparation (e.g. one paper on a functional hippocampal model for navigation, etc.).

Table 6: Output 2 links

Component	Link to	URL
C2582	Model Repository	https://github.com/HBPNeurorobotics/GazeboRosPackages/tree/demonstrator6/src/target_reaching_common https://github.com/HBPNeurorobotics/GazeboRosPackages/tree/demonstrator6/src/target_reaching_nengo
	Technical Documentation	https://collab.humanbrainproject.eu/#/collab/79892/nav/540882
	User Documentation	https://collab.humanbrainproject.eu/#/collab/79892/nav/540882
C3052	Model Repository	https://github.com/matteopriorelli/navigation
	Technical Documentation	https://collab.humanbrainproject.eu/#/collab/78682/nav/532632
	User Documentation	https://collab.humanbrainproject.eu/#/collab/78682/nav/532632
C2526	Model Repository	https://bitbucket.org/albornet/active-visual-system-t10.2.1-comp-id-c2526
	Technical Documentation	https://collab.humanbrainproject.eu/#/collab/78638/nav/532343
	User Documentation	https://collab.humanbrainproject.eu/#/collab/78638/nav/532343
C2942	Model Repository	https://github.com/mc-capolei/HBP_SGA2_C2942
	Technical Documentation	https://github.com/mc-capolei/HBP_SGA2_C2942/blob/master/C2942%20Technical%20Documentation.pdf
	User Documentation	https://github.com/mc-capolei/HBP_SGA2_C2942/blob/master/C2942%20User%20manual.pdf
C2525	Model Repository	https://github.com/aalto-intelligent-robotics/ViTa-SLAM
	Technical Documentation	https://github.com/aalto-intelligent-robotics/ViTa-SLAM
	User Documentation	https://github.com/aalto-intelligent-robotics/ViTa-SLAM
C2580	Model Repository	https://github.com/HBPNeurorobotics/UnsupervisedSensorimotorLearning
	Technical Documentation	https://github.com/HBPNeurorobotics/UnsupervisedSensorimotorLearning
	User Documentation	https://github.com/HBPNeurorobotics/UnsupervisedSensorimotorLearning

5.1.4 Output 3: Demonstrators

This Output consists of several demonstrators that showcase the capabilities of the IBA. The first of these illustrates how different modules that have been trained and refined separately can be integrated through the IBA into a control architecture that combines their functionalities for a navigation task (see C2943).

In the first demonstrator, we integrated two navigation-oriented components developed over the past two years. The first such component is an information fusion model, based on rat hippocampus inspired pose cell network (ViTa-SLAM, C2525), which fuses long-range dense camera feed with the short-range sparse haptic feedback to estimate the 3D pose of the agent in the environment. The second is a computational model of the cortico-striatal circuit, that combines Bayesian nonparametric and model-based reinforcement learning (MB-RL) (C3052; see the CDP7 Deliverable D4.4.2 SGA2 M24 for a complete description). The performance of this combination is currently evaluated for scene exploration under sub-optimal visual conditions.

The second demonstrator illustrates how the IBA can be used to investigate the combination of predictive coding and cerebellar function in initiation of movement (Figure 14). Here, the challenge is to adequately time the beginning of movement in a reaching task (robotic arm grasping a moving object from a conveyor belt – see C2943). A visual module (deep network pre-trained with

TensorFlow on the NRP) implements predictive coding and acts as an autoencoder that provides the cerebellar module (Python code) with latent representations of the world. The latter then learns and anticipates the position of the object to trigger the execution of motion primitives (C++ and Python modules).

The study is original, insofar as it investigates whether predictive coding can improve grasping success when the arm itself gets in front of the camera during the movement (akin to mental imagery).

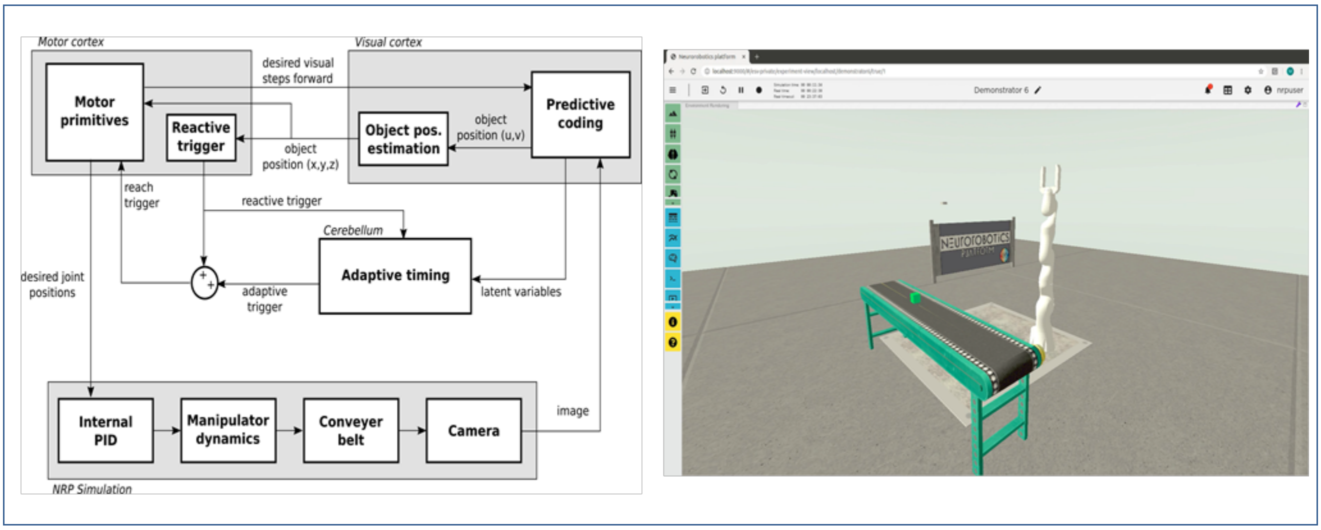

Figure 14: Demonstrator: modular control architecture and experimental setup in the NRP

Table 7: Output 3 links

Component	Link to	URL
C2943	Data Repository	https://emdesk.humanbrainproject.eu/shared/5cad9e4534da9-543b31217f459d8e65f55961e9a64905
	Technical Documentation	N.A.
	User Documentation	N.A.

5.2 Validation and Impact

5.2.1 Actual and Potential Use of Output(s)

The IBA aims to make it possible to study how different functional models can be integrated into a single cognitive architecture, and how they interact with each other to support the emergence of behavioural patterns in closed-loop embodied simulations. Its modular nature enables users to compare models of a given area or function, by swapping them inside their cognitive architecture and running comparative simulations on the NRP. The IBA should therefore be of interest to a large neuroscientific community that is interested in the functional/cognitive aspects of brain models.

From a software/algorithmic perspective, the synchronisation mechanism implemented in the IBA to control the timing of data exchange and communications between modules is robust. As a redesigned software architecture for the NRP is currently under consideration, this mechanism can be easily adapted and reused. This will be essential as, in the next phase of the project, the HBP intends to focus on the study of multi-scale, multi-simulator cognitive architectures. With the IBA, the NRP has already some mechanisms in place to accommodate such simulations. Combined with our work on parallelised/distributed learning (see Section 8.1), the IBA provides an adequate framework to implement multi-level/hierarchical learning strategies (e.g. “learning to learn”) and thereby surpass end-to-end learning methods (e.g. deep reinforcement learning) that currently represent the state of the art in Artificial Intelligence.

5.2.2 Publications

- [P2243] S. Tolu, M.C. Capolei, L. Vannucci, C. Laschi, E. Falotico, M. Vanegas Hernández (2020) A Cerebellum-Inspired Learning Approach for Adaptive and Anticipatory Control. International Journal of Neural Systems, Vol. 30, No. 01, 1950028.
  - Significance: this publication sheds light on an important function of the Cerebellum (anticipatory control), by investigating how it relates to the more "usual" roles ascribed to this brain area, such as modelling forward kinematics.
- [P1919] A. Bornet, J. Kaiser, A. Kroner, E. Falotico, A. Ambrosano, K. Cantero, M.H. Herzog, G. Francis (2019) Running Large-Scale Simulations on the Neurorobotics Platform to Understand Vision – The Case of Visual Crowding. Front. Neurorobot. 13:33. doi: 10.3389/fnbot.2019.00033.
  - Significance: this publication illustrates the usefulness of neurorobotic experiments and the NRP for hard neuroscience problems.
- [P1526] J. Kaiser, M. Hoff, A. Konle, J.C. Vasquez Tieck, D. Kappel, D. Reichard, A. Subramoney, R. Legenstein, A. Roennau, W. Maass, R. Dillmann (2019) Embodied synaptic plasticity with online reinforcement learning. Front. Neurorobot. 13:81. doi: 10.3389/fnbot.2019.00081.
  - Significance: this publication illustrates the convergence of neuromorphic computing and Reinforcement Learning through neurorobotics.

6. Key Result KR10.4: Improved NRP

6.1 Outputs

6.1.1 Overview of Outputs

6.1.1.1 List of Outputs contributing to this KR

In this reporting period, there were two major releases of the NRP: Releases 2.3 and 3.0. The Outputs listed below belong to both of these releases. Furthermore, a more detailed description of the evolution of the NRP between Release 2.0 and 3.0 can be found in D10.4.1 SGA2 M24 "Report on the Neurorobotics Platform Release 3.0". The numbering of corresponding Outputs in the present Deliverable and D10.4.1 is indicated below for convenience.

Each Output is illustrated with a demo video available from our YouTube channel, linked in the table below, and also in every output section.

Output #	Title	Released with	Output # in D10.4.1
1	Fluid simulation	3.0	5
2	Plotting tools in the Web Cockpit	3.0	6
3	Experiment designer	3.0	8
4	Interfacing with the Knowledge Graph and importing SONATA brain files	2.3 & 3.0	12 & 13
5	Large-scale NEST simulations on the NRP	3.0	11

6.1.1.2 How Outputs relate to each other and the Key Result

These Outputs all make major contributions towards improving the capabilities (Outputs 1, 4 and 5) and usability (Outputs 2 and 3) of the NRP as an integral part of the EBRAINS platform.

In particular, they fulfil the high-level expected results for SGA2, namely:

- Convergence with SP6 (Brain Simulation Platform) and SP5 (Neuroinformatics Platform): Output 4.
- User-Centric Web Cockpit interfaces: was already covered in the outputs from M12, but Outputs 2 and 3 contribute to this high-level expected result.
- Support for simulation of fluid materials: Output 1.

6.1.2 Output 1: Fluid simulation

The simulation of complex musculoskeletal models under different scenarios is a necessary step towards understanding closed-loop motor-control behaviours and locomotion patterns. For this purpose, a physics-based fluid particle simulator was integrated into the NRP, enabling the simulation of articulated rigid bodies with fluids. This functionality is provided as an extension of the physics engines of Gazebo, enabling the computation of fluid forces applied on rigid bodies. This can generate sensory feedback, thus enabling the closing of the loop between the robot and the environment. Furthermore, a visualisation of the fluid simulation has been incorporated into the NRP, enabling users to observe the behaviour of solid objects under the effect of fluid forces (Figure 15). This will enable original research through simulation of, e.g. robots moving over complex terrain (e.g. sand), aquatic drones and animals such as C. Elegans, etc.

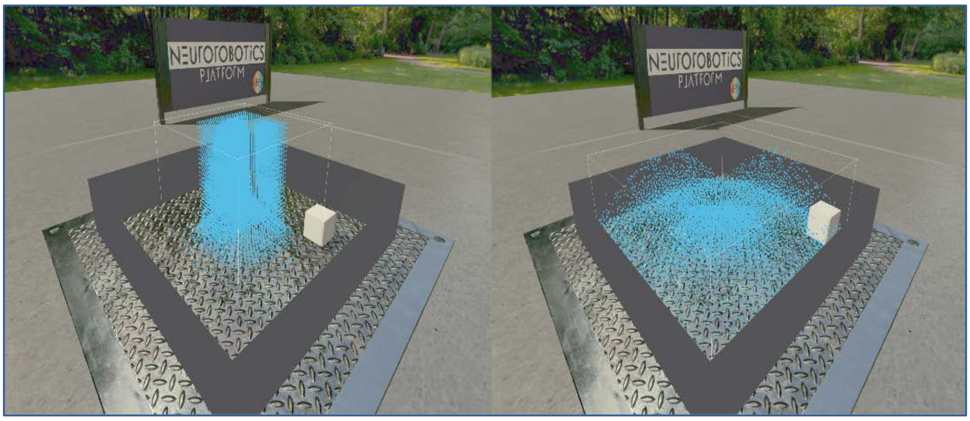

Figure 15: Two timesteps of the coupled Fluid-Rigid simulation

The box moves under the effect of the fluid forces.

Demo available at: https://www.youtube.com/watch?v=Rgy0qR7B_n4

Table 8: Output 1 links

Component	Link to	URL
C2588	Repository	https://bitbucket.org/hbpneurorobotics/exdfrontend/
	Technical Documentation	https://neurorobotics.net/apidoc/ExDFrontend/
	User Documentation	https://neurorobotics.net/Documentation/nrp/user_manual/user_interface/index.html
C2584	Software Repository	https://bitbucket.org/hbpneurorobotics/gazebo/
	Technical Documentation	http://osrf-distributions.s3.amazonaws.com/gazebo/api/9.0.0/index.html
	User Documentation	https://neurorobotics.net/Documentation/nrp/user_manual/user_interface/index.html

6.1.3 Output 2: Plotting tools in the Web Cockpit

After a long-awaited user request, a set of responsive plotting tools has been added to the "Web Cockpit" (i.e. the "frontend" of the NRP). For maximum flexibility, values coming from any Gazebo topic can be displayed, allowing the user to create custom plots from various contexts (like transfer functions or backend tools). Several different plot types are now available in several dimensions: points, lines, pie, bars, filled areas, bubbles, error bars and 3D points (Figure 16).

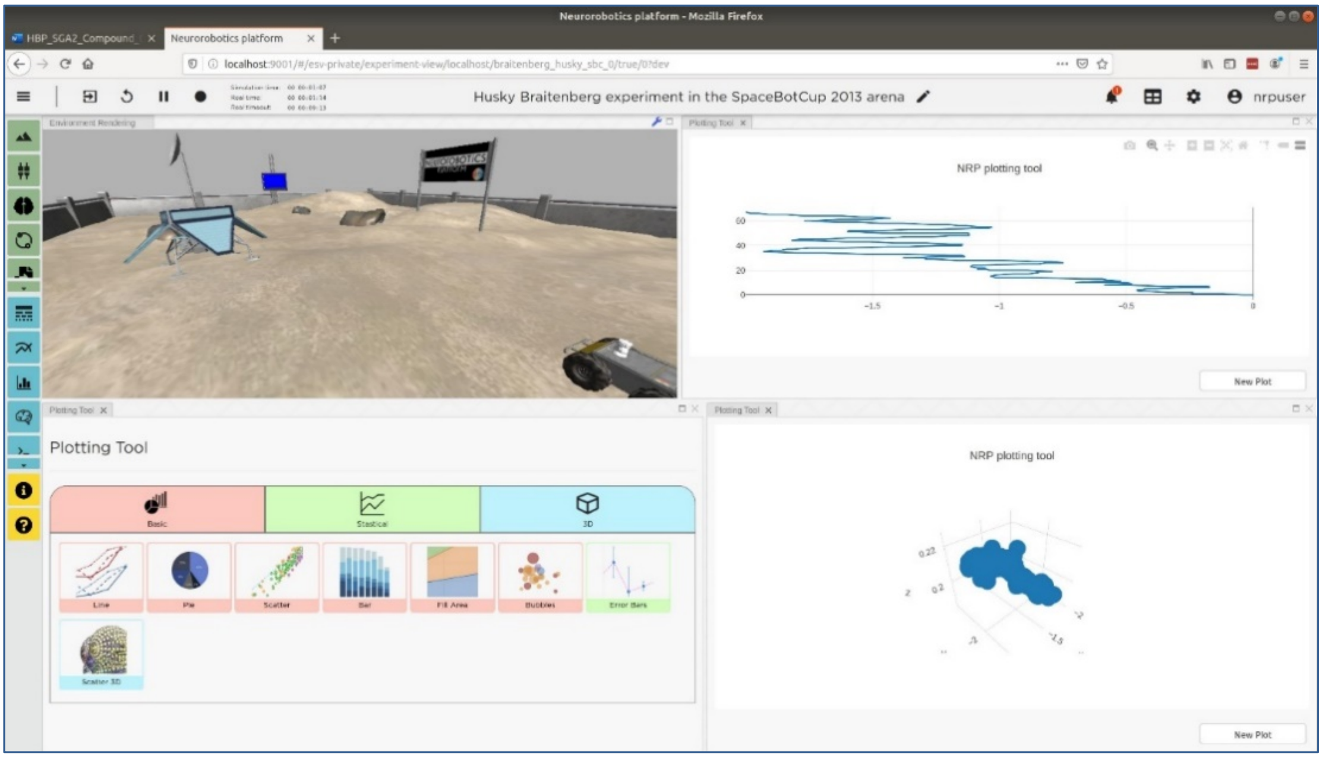

Figure 16: The Plotting Tool pane and two different plotter types

Demo available at: https://www.youtube.com/watch?v=3p-B9odA-I4

Table 9: Output 2 links

Component	Link to	URL
C2583	Repository	N/A
	HLST-NRP user portal	https://neurorobotics.net/submitbug.html
	User Forum	https://forum.humanbrainproject.eu/c/neurorobotics
C2588	Repository	https://bitbucket.org/hbpneurorobotics/exdfrontend/
	Technical Documentation	https://neurorobotics.net/apidoc/ExDFrontend/
	User Documentation	https://neurorobotics.net/Documentation/nrp/user_manual/user_interface/index.html

6.1.4 Output 3: Experiment designer

The Neurorobotics Platform attracts users from interdisciplinary teams with various backgrounds. It is essential for the entire team to be able to implement experiments and understand existing implementations. The NRP uses SMACH state machines to provide a structured method for scripting experiments. Until now, the interface for this state machine was a text editor for Python code. This was a sub-optimal solution for anyone but seasoned computer scientists.

With the new experiment designer, we offer a graphical user interface to implement the behaviour of experiments and add events to the simulation work flow. Users can simply draw states and transitions to create a visual state machine diagram (Figure 17). Several tools are available to allow the user to modify the state machine diagrams. Actions can be defined that are then executed at runtime to control the experiment environment. The possible outcomes of these actions are used as conditions for transitions to other states.

This new graphical editor provides an accessible interface for users without programming skills, can improve the speed of implementations, reduce errors and increase the overall understanding of the experiments.

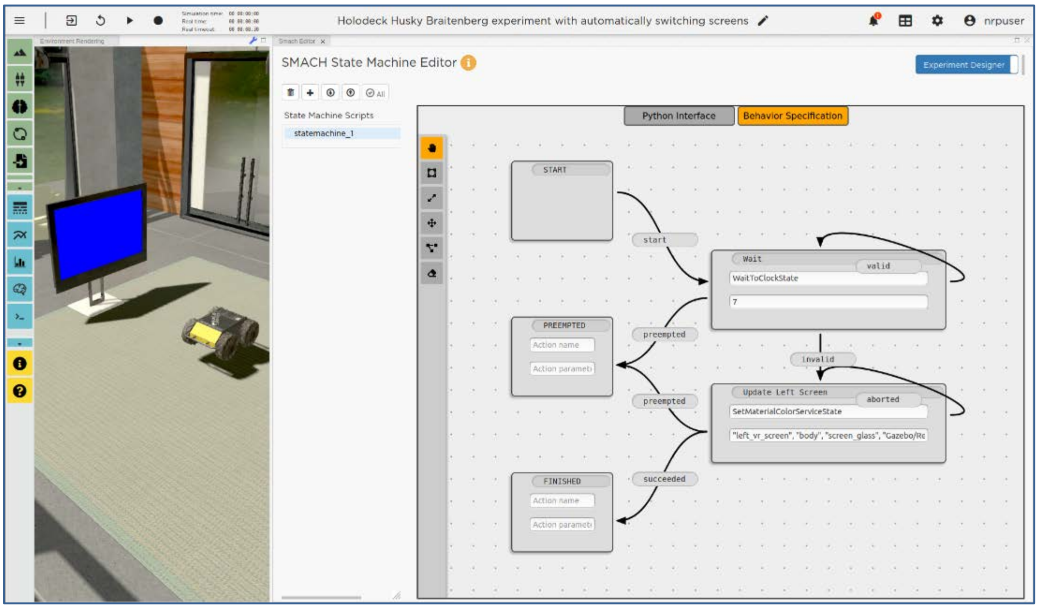

Figure 17: The graphical experiment designer makes it easier to add events

Demo available at: https://www.youtube.com/watch?v=0a_S37RuemA

Table 10: Output 3 links

Component	Link to	URL
C2588	Repository	https://bitbucket.org/hbpneurorobotics/exdfrontend/
	Technical Documentation	https://neurorobotics.net/apidoc/ExDFrontend/
	User Documentation	https://neurorobotics.net/Documentation/nrp/user_manual/user_interface/index.html
C2589	Repository	https://bitbucket.org/hbpneurorobotics/frontendstatemachineeditor
	Technical Documentation	https://neurorobotics.net/apidoc/ExDFrontend/
	User Documentation	https://neurorobotics.net/Documentation/nrp/user_manual/user_interface/edit/7-gz3d-edit-environment.html

6.1.5 Output 4: Interfacing with the Knowledge Graph and importing SONATA brain files

An interface between the NRP and the EBRAINS Knowledge Graph (KG) was implemented, thus paving the way for the practical connection of the Brain Simulation Platform (BSP) to the NRP. Users are now able to easily download brain models available on the KG from the NRP frontend. The online brains are available inside the object library in the same way as the SP10 template brains.

After having ran an experiment with a brain from the KG, users will be able to upload the data collected inside their experiment onto the KG. The data are attached to a simulation's node of the graph, which reports the author, the date and other meta-data. Thanks to the connection between brain models, simulations and data nodes, the users can navigate the KG to find and download experimental data related to the brain models stored therein.

In addition, users are now able to import brains described in SONATA into the NRP, using PyNN as the neuronal simulator's interface. After having imported the brain into the simulation, the user can proceed as usual (e.g. defining populations of neurons, adding new variables, etc.).

Demos available at: https://www.youtube.com/watch?v=jB4SsV35iTk and https://www.youtube.com/watch?v=OYJU1Zv4eKw

Table 11: Output 4 links

Component	Link to	URL
C2585	Repository	https://bitbucket.org/hbpneurorobotics/cle/
	Technical Documentation	https://neurorobotics.net/apidoc/CLE
	User Documentation	N.A. (the CLE works in a manner that is transparent to the user).
C2588	Repository	https://bitbucket.org/hbpneurorobotics/exdfrontend/
	Technical Documentation	https://neurorobotics.net/apidoc/ExDFrontend/
	User Documentation	https://neurorobotics.net/Documentation/nrp/user_manual/user_interface/index.html
C2594	Repository	https://bitbucket.org/hbpneurorobotics/nrpdocker
C2594	Technical Documentation	https://hbpneurorobotics.atlassian.net/wiki/spaces/HSP10/pages/14942234/Docker+deployment
	User Documentation	https://neurorobotics.net/local_install.html
C2592	Repository	https://bitbucket.org/hbpneurorobotics/exdbackend/src/development/hbp_nrp_cleserver/hbp_nrp_cleserver/bibi_config/
	Technical Documentation	https://neurorobotics.net/apidoc/ExDBackend/
	User Documentation	https://neurorobotics.net/Documentation/nrp/specifications/experiment_files.html?highlight=interfacing#bibi-file

6.1.6 Output 5: Large-scale NEST simulations on the NRP

In order to bring clear value to its users, the NRP must be able to run very large-scale brain simulations. This can only be achieved by leveraging the power of the HBP HPC infrastructure. We thus set out to achieve distributed NEST simulations on the Piz Daint cluster at CSCS. Even though NEST and PyNN have native support for distributed simulations (through MPI communication), the particularities of the NRP architecture demanded additional work in order to enable this support within NRP closed loop simulations. Indeed, the brain simulation in the NRP runs in the same process as the closed loop engine (CLE) component that synchronises NEST and Gazebo simulations. It thus became necessary to distribute part of the CLE along with the brain simulation (Figure 18). On each MPI rank the CLE extracts information from the simulated neurons at that rank. At each simulation step this information is gathered and processed at rank 0, which handles the overall synchronisation of the simulations.

Demo available at: https://www.youtube.com/watch?v=3cYGOC7teZ4

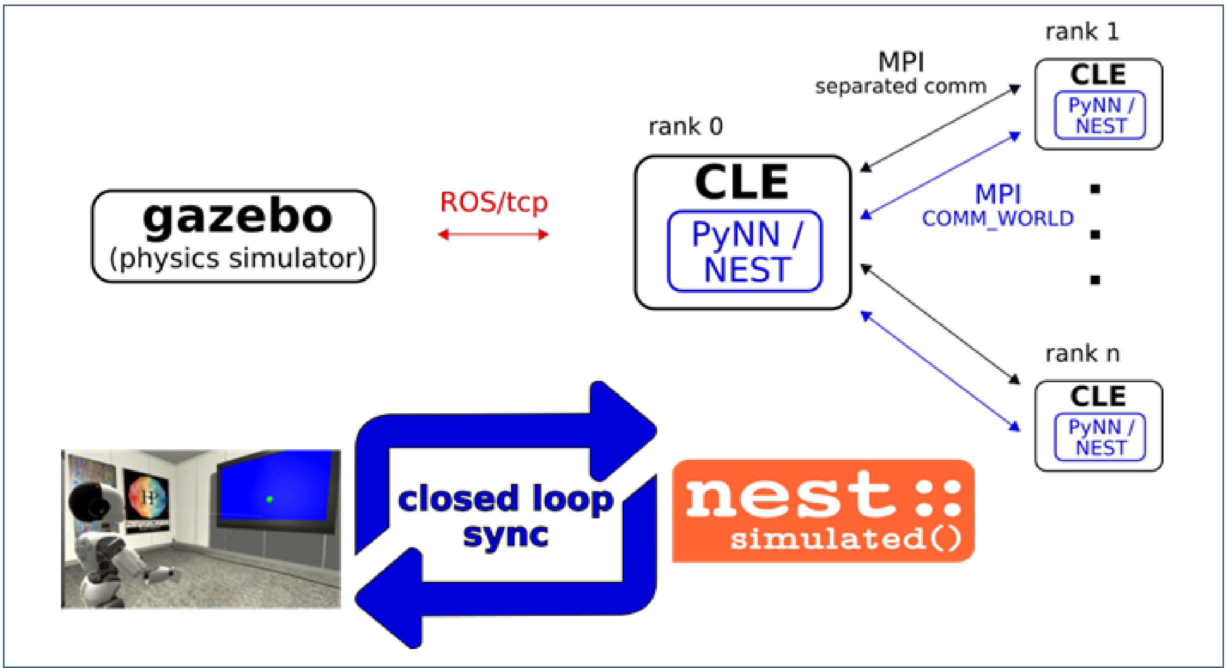

Figure 18: Co-distribution scheme of NEST and the CLE on the Piz Daint cluster

6.2 Validation and Impact

6.2.1 Actual and Potential Use of Output(s)

The Outputs described above add essential value to the Neurorobotics Platform and the HBP in general. For a complete list of Outputs, please refer to D10.4.1 SGA2 M24 "Report on the Neurorobotics Platform Release 3.0". These Outputs have considerably increased the usability to the NRP, with a new interface, the possibility to use multiple robots, the complete workflow to create experiments from scratch in the Web Cockpit, new plotting tools, the graphical experiment designer and the web robot designer. These developments all lower the entry barrier for new NRP users.

Most importantly, the NRP took several decisive steps towards EBRAINS; firstly by integrating an interface to the KG and supporting brain file formats like SONATA, then by deploying the NRP on the HBP HPC infrastructure and running large-scale NEST simulations. This opens up many unique possibilities for neuroscientific experiments in the next phase of the project (SGA3).

6.2.2 Publications

All publications referring to the NRP can be understood to be based on the work described above.

7. Key Result KR10.5: Modular neural control for physical robots under real-time constraints

7.1 Outputs

7.1.1 Overview of Outputs

Research into neuronal motor control for robots explores real-time capable distributed neuronal methods to operate intricate robotic actuators, such as typically found in compliant systems or soft robotics. This Key Result regroups various contributions towards establishing the expertise and tools required to accurately actuate such novel robots.

7.1.1.1 List of Outputs contributing to this KR

The Outputs are as follows:

- Output 1: real-time execution of a meso-scale cerebellar model accelerated on GPU and multi-CPU to control a collaborative industrial robot.
- Output 2: neuronal models on neuromorphic hardware to control a modular stiff robot.
- Output 3: neuronal models on neuromorphic hardware to control a compliant robot.

7.1.1.2 How Outputs relate to each other and the Key Result

The transversal elements common to these Outputs are the implementation of cerebellar-inspired models based on spiking neural networks and the use of the NRP as a common tool for their validation as controllers for torque-driven robots. The cerebellar controller in Output 3 is a simplified network derived from the one developed in Output 1. The cerebellar controller in Output 2 shares architectural features with the previous two, but differs in terms of where learning is applied.

7.1.2 Output 1: STDP in large cerebellar spiking simulations

A real-time cerebellar spiking neural network incorporating a continuous learning process was validated with NEST and integrated into a closed-loop system to control a virtual Baxter collaborative robot (“cobot”) in NRP. The controller was then adapted to and tested with the physical Baxter cobot (Figure 19). To cope with real-time constraints, the model was run on GPU and multi-CPU setups.

The Baxter cobot is an inherently compliant collaborative robot with 6 degrees of freedom, meant to operate autonomously and reactively in complex unstructured environments. Passive intrinsic compliance demands torque control, which deals with the robot’s inner dynamics, i.e. the evolution through time of the state of the physical system. Temporal coding in spiking neurons is thus highly relevant in this context, as it can capture the temporal evolution of analogue sensorimotor signals. To implement an effective real-time dialogue between the network spike domain and sensorimotor analogue domain, a set of analogue-to-spike/spike-to-analogue modules compatible with the Robot Operating System (ROS) were used.

The cerebellar network was divided into six identical microcircuits (also referred to as micro-complexes), each focusing on controlling a different robot joint. Each micro-complex consisted of five neural layers (Figure 19-c): 1) mossy fibres (MFs); 2) granule cells (GCs); 3) climbing fibres (CFs); 4) Purkinje cells (PCs); and 5) deep cerebellar nuclei (DCN). Overall, the model consisted of ~62k leaky integrate and fire (LIF) neurons, with ~36.4 million synapses, 36 million of which were endowed with spike-timing-dependent plasticity (STDP) to mimic biological neural processing.

Circular, figure-eight, and point-target-reaching trajectories constitute standard tasks for cerebellar benchmarking, and were used as such (Figure 19-d). Benchmarking was completed with a set of human-robot interactions, to test compliance. The torque control approach proved to outperform the accuracy of the default, factory-installed position control in such a set of tasks (see P2362).

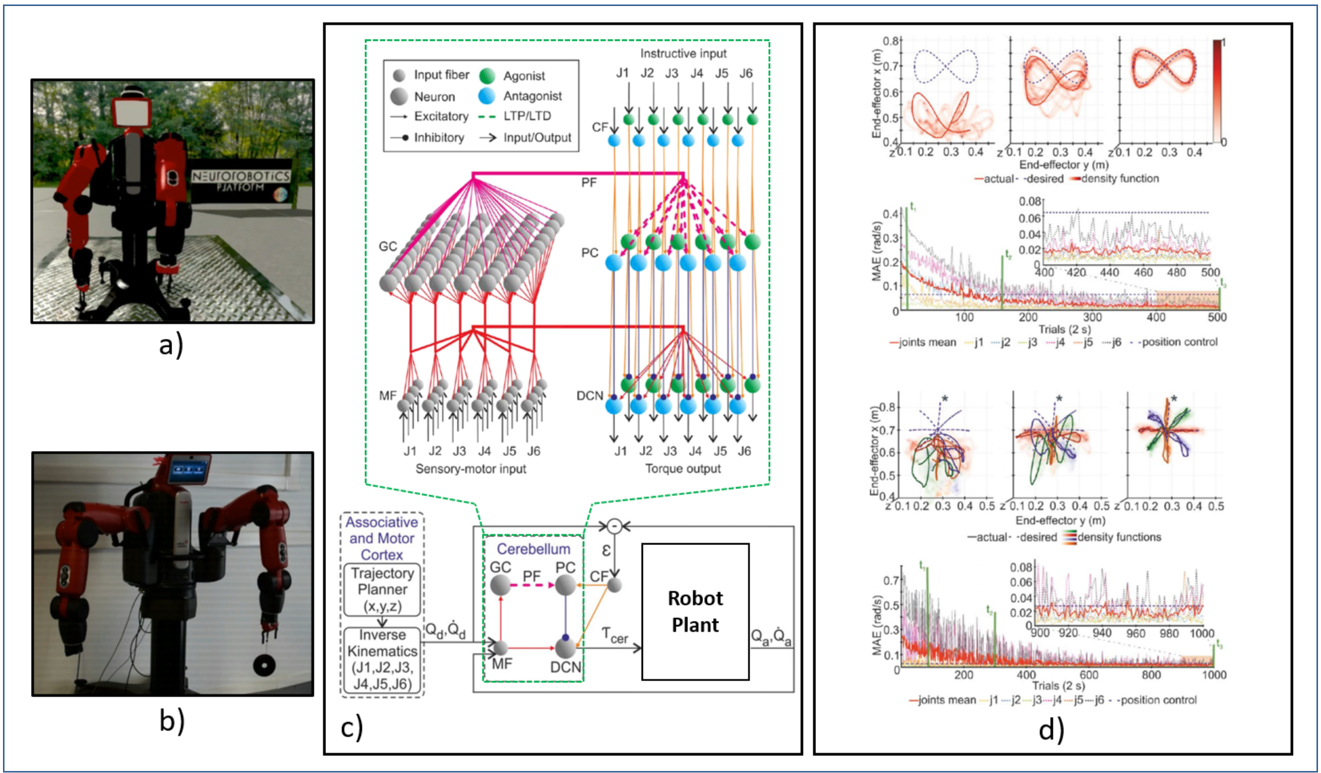

Figure 19: Large cerebellar model implemented on the NRP and on the actual robot

a) Virtual Baxter robot in NRP. b) Physical Baxter robot. c) Bio-inspired Controller: cerebellar closed-loop control with zoom in neural structure. d) Results for two tasks: figure-eight-trajectory and point-target-reaching.

Table 12: Output 1 links

Component	Link to	URL
C2561	Model Repository	https://gitlab.com/neurocomputing/hbp/sga2/c2561/model
	Technical Documentation	https://gitlab.com/neurocomputing/hbp/sga2/c2561/technical https://collab.humanbrainproject.eu/ - /collab/80250/nav/543314
	User Documentation	https://gitlab.com/neurocomputing/hbp/sga2/c2561/user

7.1.3 Output 2: Cerebellar-like neuronal control on neuromorphic hardware

In a simpler approach than that used in Output 1, a modular spiking cerebellar-like circuit was implemented on neuromorphic hardware (SpiNNaker). The model maintains the characteristic division into distinct cerebellar micro-complexes, each responsible for learning the internal model of a given robot module or robot joint. Depending on the internal mode (user defined: inverse or forward), the microcircuits received different sensorimotor information streams to give motor or sensory outcomes. The architectures were tested on both virtual and physical Fable robots (Figure 20), to perform tasks with changing kinematic and dynamic conditions: figure-eight trajectory with constant or varying amplitude and with/without external load at the end-effector.

The composite architecture combines the advantages of forward and inverse models and the learning of the latter is improved by the prediction error of the first. Tests showed that the architectures have different adaptation scales, depending on the disturbances and contexts. Different configurations using the modular Fable robot were tested for the validation and generalisation of the control and learning mechanisms, both in manipulation and locomotion tasks within the NRP.

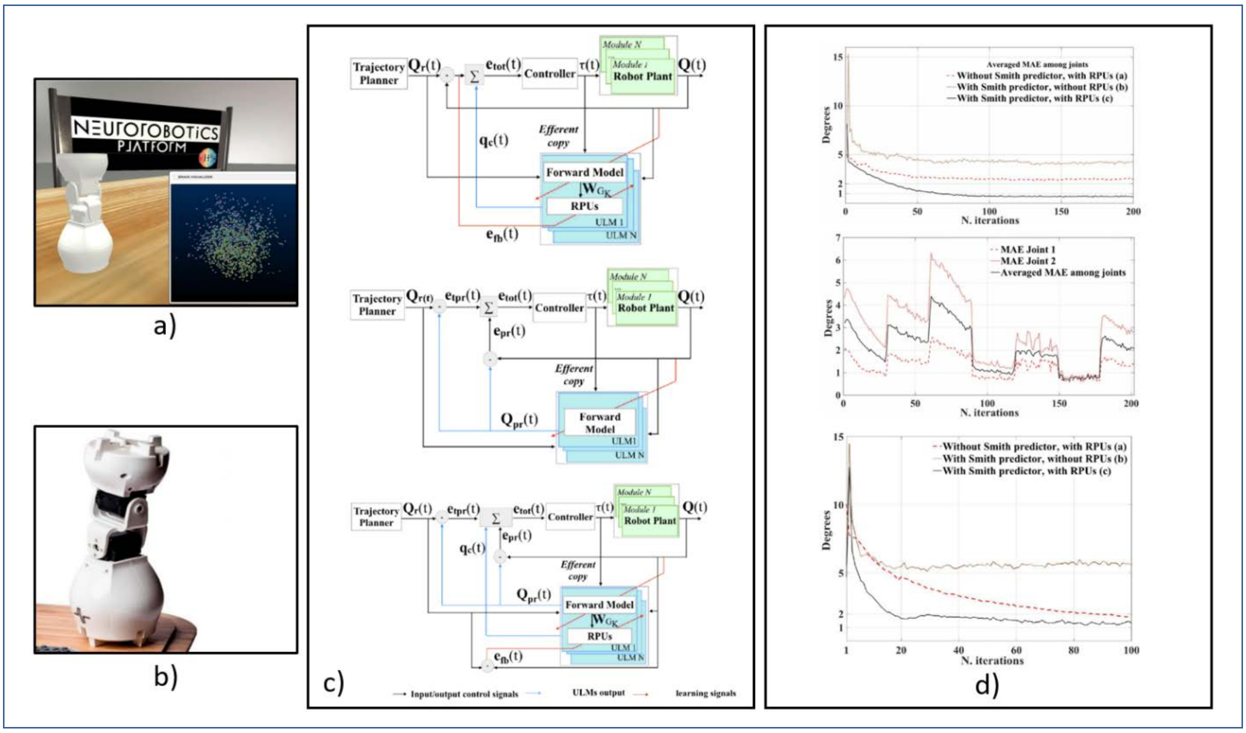

Figure 20: Simplified cerebellar model running on SpiNNaker

a) Virtual Fable robot in NRP. b) Physical Fable robot. c) Bio-inspired Controllers: cerebellum without/with Smith Predictive Mechanism and with/without readout plasticity units (RPUs). d) Results for three tasks involving figure-eight trajectory: with constant amplitude, with varying amplitude and with external load at the end-effector.

Table 13: Output 2 links

Component	Link to	URL
C2566	Model Repository	https://gitlab.com/neurocomputing/hbp/sga2/c2566/model
	Technical Documentation	https://gitlab.com/neurocomputing/hbp/sga2/c2566/technical
	User Documentation	https://gitlab.com/neurocomputing/hbp/sga2/c2566/user

7.1.4 *Output 3: Neuronal real-time controllers for manipulation task*

A musculoskeletal robotic arm, with two joints and four tendon-driven artificial muscles, was used both in simulation in the NRP and as a physical real-time system to develop and compare three brain-inspired controllers (Sections 7.1.4.1-3, see Figure 21). The task intended to benchmark the controllers consisted of picking up an *a priori* unknown object, learning about its implicit properties (e.g. mass and inertial tensor) through active manipulation, and based on such throwing the object towards a target. Completion of the throwing motion with the three controllers is still a work in progress.

7.1.4.1 Spiking Cerebellum

A simplified cerebellum controller (one micro-complex for each actuator) was implemented, derived from the model used in Output 1. The controller received muscle states as inputs, and provided force control as output. Training happened through dynamic plasticity updates during object manipulation (STDP learning). The controller was run on the SpiNNaker chip. A position control task with this model is represented in panel d) of Figure 21, which shows the position error of a single joint in the simulated robot as it moves towards a desired angle. Each joint is actuated by two muscles; the oscillations show how both muscles attempt to compensate for the error induced by the opposing muscle.

7.1.4.2 Recurrent network

A model-free approach that uses a recurrent neural network to learn the robot forward dynamics was developed. The mapping between sequences of muscle commands and end-effector positions was learned with a LSTM network, the dataset of which was created using a motor babbling procedure. After training the forward model, an optimisation algorithm used the LSTM-trained artificial neural network to compose the muscle commands necessary to perform non-trivial tasks like reaching and throwing of an object.

7.1.4.3 Feed-Forward network

A one-layer, fully connected feedforward neural network was implemented to learn the sensory-motor mapping between muscle states (length and applied force) and muscle commands with a variation of differential Hebbian learning. After a self-exploration phase, the network drives a periodic motion in resonance with the whole system (arm and attached object). This motion is leveraged to release the object in an exact state (position and velocity) and generate the desired throwing trajectory.

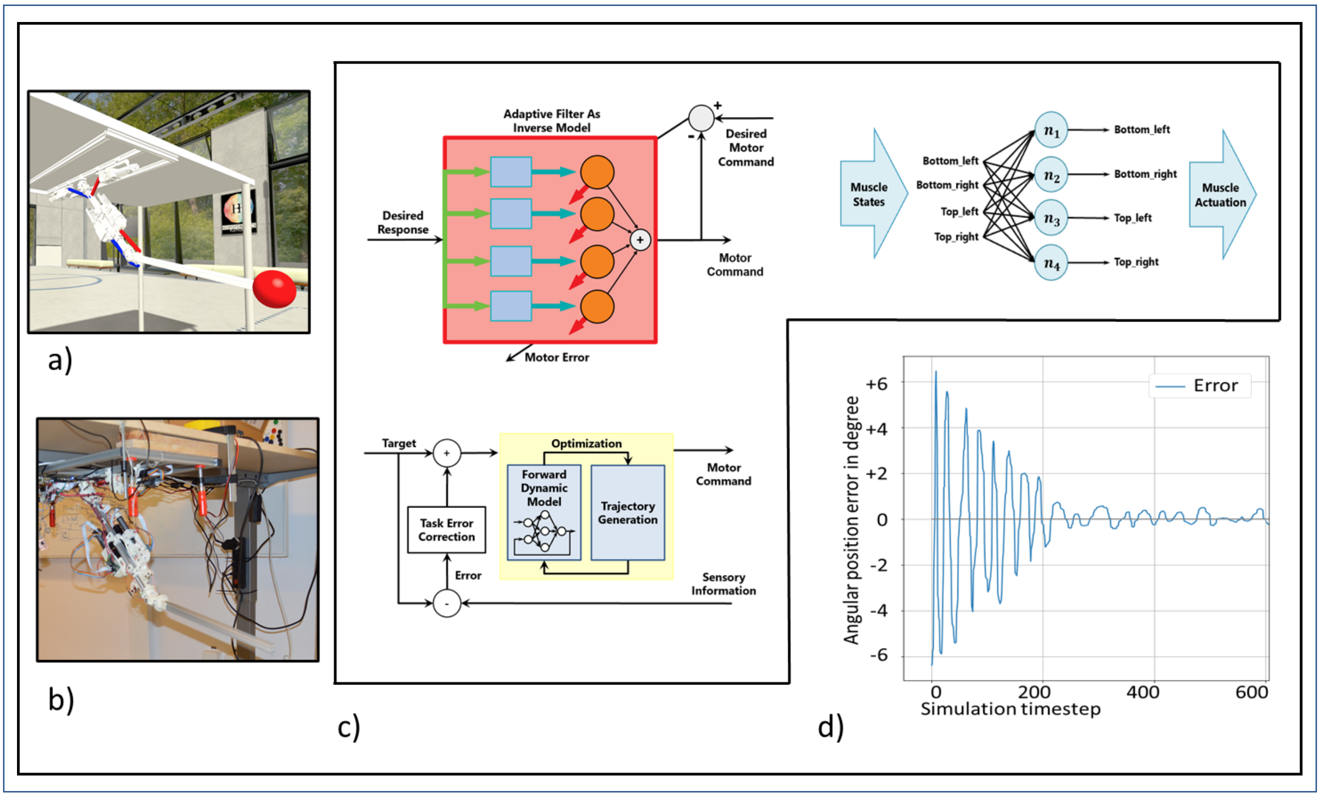

Figure 21: Multiple controllers explored in the object manipulation task

a) Virtual myo-robotic arm in NRP. b) Physical myo-robotic arm. c) Bio-inspired Controllers: spiking cerebellum, recurrent NN, feedforward NN. d) Position control task with a spiking Cerebellum network (1 simulation timestep corresponds to 10 ms).

Table 14: Output 3 links

Component	Link to	URL
C2561	Data Repository	https://gitlab.com/neurocomputing/hbp/sga2/c2561/model
	Technical Documentation	https://gitlab.com/neurocomputing/hbp/sga2/c2561/technical
	User Documentation	https://gitlab.com/neurocomputing/hbp/sga2/c2561/user
C2571	Model Repository	https://bitbucket.org/alex_vds_ugent/flexsensor/
	Technical Documentation	https://bitbucket.org/alex_vds_ugent/flexsensor/src/master/doc/
	User Documentation	https://bitbucket.org/alex_vds_ugent/flexsensor/src/master/doc/
C2566	Data Repository	https://gitlab.com/neurocomputing/hbp/sga2/c2566/model
	Technical Documentation	https://gitlab.com/neurocomputing/hbp/sga2/c2566/technical
	User Documentation	https://gitlab.com/neurocomputing/hbp/sga2/c2566/user

7.2 Validation and Impact

7.2.1 *Actual and Potential Use of Output(s)*

The models from Outputs 1, 2 and 3 are in use on their respective robots, both in simulation (NRP) and on the existing physical instance of the robot. Various additional robotic systems (particularly those with compliant actuation, e.g. the HBP robot mouse, KR10.2) can benefit from the significant real-time adaptability that cerebellar controllers offer. Indeed, such controllers promise improved operation of future low-cost robots, for which mechanical specifications may not be achieved with great precision. Adaptive cerebellum-inspired controllers could adjust to changes in system properties during runtime. They could also adapt to wear-induced mechanical changes throughout the operational life of a robot. The long-term vision of using such control models for compliant robots is to achieve safe human-robot interactions (e.g. in factory co-working settings or on neuro-prosthetic devices).

7.2.2 *Publications*

- [P2362] I. Abadía, F. Naveros, J.A. Garrido, E. Ros, and N.R. Luque (2019) On Robot Compliance: A Cerebellar Control Approach. IEEE Transactions on Cybernetics, Vol. 99, pp.1–14. DOI: 10.1109/TCYB.2019.2945498.
  - This publication is significant insofar as the proposed brain-inspired approach for compliant control outperforms the accuracy of the default factory-installed position control in a set of tasks.

8. Outputs not directly linked to Key Results

8.1 Output 1: Parallelisation of NRP simulations for data-intensive learning paradigms

Virtualisation of experiments on the NRP opens up many interesting possibilities. For example, the NRP not only makes experiments possible that could not be conducted in the real world (e.g. recording spikes from an arbitrary number of neurons during execution of a task); it also enables execution of multiple experiments in parallel, in a manner that is only constrained by compute power and space for data storage. This latter ability is particularly adapted to speed up learning in embodied settings, with paradigms that are data-intensive or converge slowly (e.g. Reinforcement

Learning). While it was well understood that the architecture of the NRP could inherently support this distributed parallel execution of experiments, an actual implementation had so far been missing.

We thus developed a new deployment scheme that builds on the existing NRP infrastructure. Its main components are outlined in Figure 22. Single NRP instances are deployed using Docker containers, which makes every NRP instance an independent unit that can carry out a simulation. The number of parallel experiments (Experiment 1, …, Experiment n) is therefore only limited by the number of containers that can be launched at the same time. To enable users to interact with the experiment a high level of abstraction, we introduced an experiment API (nrp.experiment). This is specific to the type of experiment (e.g. grasping objects and navigating in an environment) and needs to be implemented every time a new class of experiments is added.

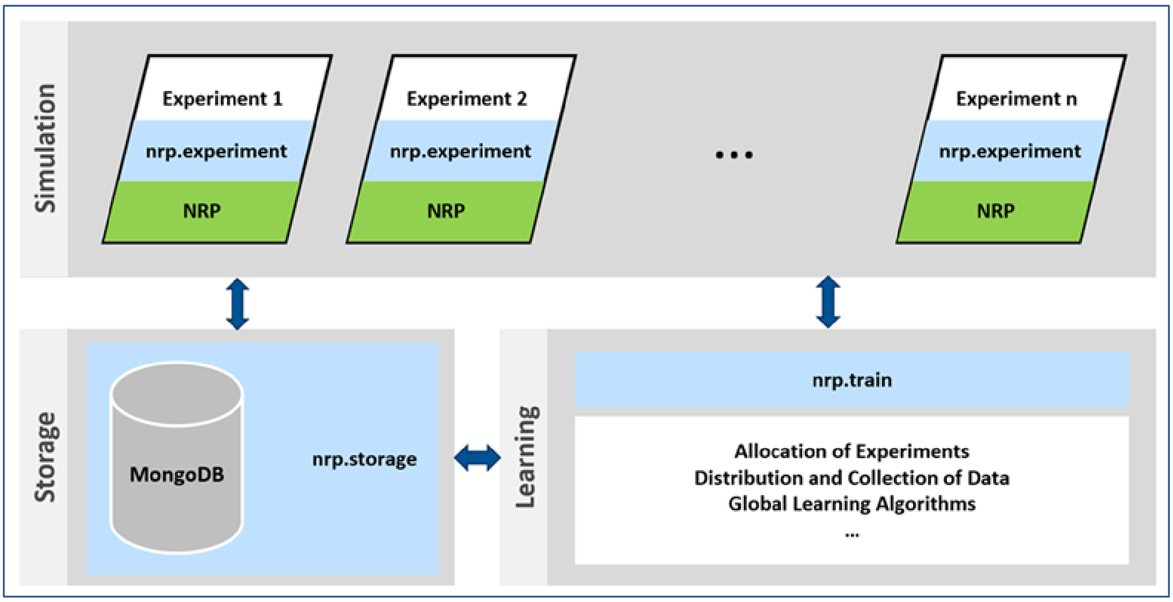

Figure 22: High-level view of parallel distributed data collection and learning with the NRP

This approach allows users to access different experiment instances of the same class with the same code. Changing low-level details such as controllers or replacing a robot by another one with a similar kinematic structure will therefore not affect the user's code for controlling the experiment. This is very similar to common tools from machine learning such as OpenAI Gym and considerably increases productivity, since the same code can be reused across different experiments that can run in parallel. Data from all simulations are stored and shared in a new storage component that is based on MongoDB, a document database (nrp.storage). The third component of the new setup is the nrp.train, which contains all code required for the coordination. It will host different types of models (from both neuroscience and machine learning) and will support different modes of distributing the workload across simulations.

To validate the architecture outlined above, we implemented a benchmark virtual experiment based on a real-world experiment carried out with physical robots at Google[3]. The goal of that work was to learn grasping solely based on RGB camera input, by collecting data from up to 18 robots that were operating and parallel and grasped objects over a period of two months. While the experiment was extremely costly in terms of resources (high number of robots and long duration of the experiment), the underlying approach can scale extremely well to a much higher number of robots.

We replicated the experimental setup used at Google as a digital twin on the NRP (see Figure 23). The new experimental setup has been successfully executed in parallel on up to 75 NRP simulations running on the HBP infrastructure at CSCS with our new architecture for parallel distributed execution. Here, the number of simulations was only limited by compute resources. This enabled the collection of vast amounts of data (hundreds of Gigabytes), meaning the real-world experiment could be reproduced in about a week. With that data, offline end-to-end training of action policies

[3] Levine, Sergey, et al. "Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection." The International Journal of Robotics Research 37.4-5 (2018): 421-436.

was performed as a prerequisite of sim-to-real transfer and exploration of the reality gap (publication in preparation).

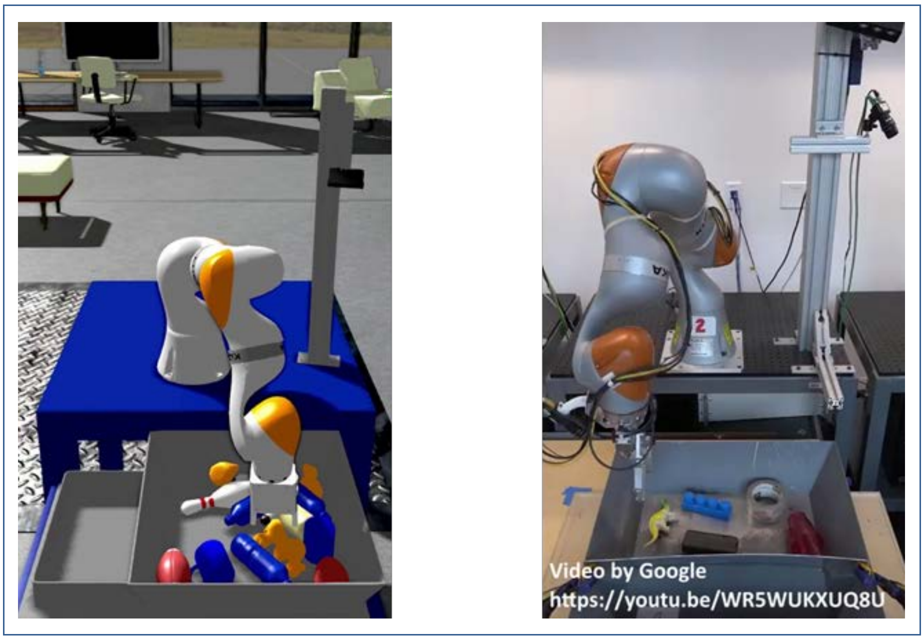

Figure 23: Virtual experiment in the NRP and actual setup by Google (latest version)

This work is a strong proof of concept for distributed / parallelised learning with the NRP on the HBP infrastructure. It paves the way for further developments and implementation of learning/optimisation frameworks (e.g. Pagmo, DMTK, etc.) that will uniquely enable large-scale learning experiments with the NRP. In combination with the IBA and redesigned interfaces (see next section), this capability can be instrumental in implementing hierarchical learning in cognitive architectures and, as such, is expected to be an essential component for the work currently planned for the next phase of the project.

8.2 Output 2: Roadmap for the evolution of the NRP

The work carried out in SGA2 to achieve several Outputs reported herein (in particular, IBA and large-scale NEST simulations on Piz Daint) made it impossible to ignore some limitations of the general software architecture of the NRP. This prompted an intensive collaboration between specialists of NEST (JUELICH), HPC (CSCS) and Neurorobotics (TUM, SSSA, UGR) over the second year of SGA2, resulting in many fruitful discussions as well as a common understanding of how the NRP could benefit from its upcoming integration into EBRAINS.

The gist of the conclusions reached over the past few months is that the architecture of the NRP must evolve: in particular, a clean delineation must be achieved between its different components, starting with the interface between NEST and the CLE. The initial work carried out in the past year with NEST explores the standardisation of the coupling between NRP with simulators, a first step towards generalisation of interfaces. In the long term, it is proposed that a client-server architecture with a yet-to-be-created API be implemented for NEST (see Figure 24), and ultimately other simulators (especially TVB). This new interface architecture decouples the execution model between the CLE and the brain simulators, allowing more flexibility in the type and execution modes of the latter.

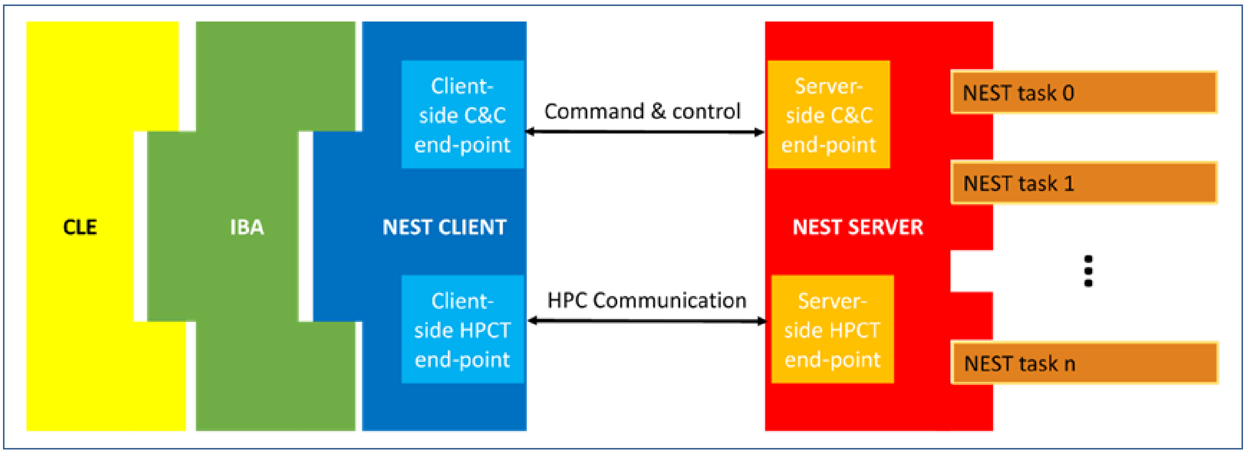

Figure 24: Proposed novel architecture and interfaces between NRP components

Transitioning to such an architecture will allow for easier maintenance of the NRP code base, irrespective of the development cycle of components such as NEST, PyNN, etc. It will also create the opportunity to establish similar APIs for different simulators, thereby providing a clear path towards multi-scale, multi-simulator experiments on the NRP; this clearly aligns well with the scientific goals of the HBP over the next few years, and it positions the NRP as the central nexus for several planned research efforts.

Finally, if this clean delineation of components and the availability of well-defined, well-documented interfaces with the NRP and other brain simulators of EBRAINS is successfully implemented (which remains a significant technical challenge), then users will be encouraged to develop, maintain and connect their own new services to those already in existence. Freed from the need to delve into the code of every component of EBRAINS, they will face a much lower threshold of entry as EBRAINS contributors, which we see as a clear prerequisite for a strong expansion of the EBRAINS user community.

9. Conclusion and Outlook

The Key Results presented above address many different areas of research and exhibit varying degrees of maturity. **KR10.4** (the NRP itself) holds a central position around which other KRs were articulated. Besides the continuous work to provide the NRP with improvements and new functionalities, a significant number of activities were dedicated to connecting the NRP to other HBP Platforms. In particular, deployment of complex NRP simulations on the HBP HPC infrastructure (large-scale NEST simulations, parallel execution of many back ends for learning in the “Google experiment”) was an essential step to provide the NRP with the tools required to retain an advantage over its competition (e.g. OpenAI Gym, Nvidia Isaac, Facebook Habitat, AWS RoboMaker, etc.) and differentiate itself in the growing field of simulators for embodied AI. To this day, the NRP is the only free, open-source framework that is specifically adapted for spiking network simulation and, as such, can support the *in-silico* prototyping of neuromorphic applications. This latter point was further reinforced by the development of a “plug-and-play” interface with SpiNNaker boards, as well as the integration of Intel Loihi chips as back ends to the NRP (note: the work on Loihi chips was not funded by the HBP).

The part of our work with the clearest translational potential in medicine is **KR10.1**. The latter regroups our various contributions towards simulating the neural mechanisms supporting legged locomotion and stroke rehabilitation, focusing on models of the motor cortex and its interaction with spinal cord signals. Significant results were obtained that demonstrate the relevance of simulation as a valid research tool to eventually restore locomotion in humans with epidural spinal cord stimulation. More generally, this work is essential to validate any *in silico* approach to embodiment that deals with motor control. In particular, the work focused on the simulation of stroke in mice holds much promise for understanding the mechanisms of neuro-rehabilitation. In addition, a detailed mouse model is now available in the NRP for neuroscientific experiments, which should attract more users to the NRP.

KR10.2 follows the completion of the final release of the rodent robot (Version 4.1). No additional design work on hardware (mechanical design, electronics, etc.) will be performed in the next phase of the Project, as it was decided that activities linked to mechanical design of robotic platforms were beyond the purview of the HBP. NeRmo is nevertheless now in use at several Partner institutions in the HBP, as well as in Japan. In that sense, it already fulfils its initial promise of becoming a useful research platform. The recent addition of its digital twin on the NRP makes it especially suitable for studying sim-to-real transfer learning. While we have already started working on this topic, further efforts will be required to provide a clear demonstration of the fact that the NRP can support virtual prototyping and transfer learning from simulated to actual robots. As such, it will certainly become a major showcase for potential industrial partners in the field of robotics. The compute power embedded in NeRmo is already sufficient for that purpose, but future users (especially within HBP) may require support for embedded neuromorphic chips. Such a development is thus currently under consideration for the next phase of the Project. With that addition, NeRmo will become the perfect extension of NRP simulations into the real world, for prototyping many neuromorphic applications, such as legged locomotion, navigation, visual processing, decision-making, etc.

A necessity for implementing modular brain simulations on the NRP, **KR10.3** provides an interesting path to integrate further neuroscience and neurorobotics research. The software framework developed proved functional and sufficiently user-friendly to implement two demonstrators implementing non-trivial behaviour through integration of multiple neuro-computational modules. These demonstrators leverage heterogeneous components (in terms of both function and implementation) and let users ask questions regarding the computational synergies and behavioural impact of models that, studied in isolation, would not reveal much in that regard, even embodied. In spite of its limitations, especially as they relate to the communication layer (see D10.2.1 SGA2 M24 for further discussions on this topic), the IBA is therefore a most relevant framework to study how the brain orchestrates together areas of narrow functional expertise. This topic currently attracts much attention in both neuroscience and AI (see for example Deco, G. *et al*. (2019) Revisiting the Global Workspace: Orchestration of the functional hierarchical organisation of the human brain. https://doi.org/10.1101/859579). The IBA and its future iterations will therefore be of major interest for the next phase of the project, where modular cognitive architectures will take centre stage.

KR10.5 is an essential step towards validating *in silico* experiments as practical tools for robotics, and brain-derived controllers as viable alternatives to their “classical” counterparts derived from control theory and/or machine learning. For the latter aspect in particular, the concept of using cerebellar-like modules to adapt to variations in robot characteristics (e.g. variations inherent to the production process, or evolution with wear and tear) is highly relevant for practical applications, insofar as it could increase the operational life span of costly systems, reduce downtime due to maintenance, etc. While these are also potential benefits of controllers derived from adaptive control theory, our increasing understanding of the inner workings of, and synergies between, different brain areas involved in motor control holds much promise to go beyond the current state of the art. Theoretical works validated through simulations are a good starting point; it is, however, essential to keep confronting neuroscience-based theories with the practical challenges of real-world robotic embodiment.

Taken together, these various results highlight the contributions that Neurorobotics can make to neuroscience and robotics, and how these, in turn, can support each other and provide a path for currently separated research lines to converge. This will be essential to achieve the type of impact sought by the HBP, i.e. progress in neuroscience research and development of new computing paradigms.

Furthermore, it is clear that the uptake of neurorobotics in the neuroscience community has accelerated over the past year, with multiple groups inside (E. d’ANGELO; M. MIGLIORE) and outside of the HBP (B. PORR) now working with the NRP independently of SP10. With the NRP development roadmap firmly set on fully leveraging the emergence of EBRAINS in general and its HPC infrastructure in particular, it is expected that this movement will gain momentum. We can already report, for example, that a Japanese consortium will start working with some current SP10 partners in the coming months, supported in that endeavour by several HBP vouchers. Such collaborations are essential to nurture, as they will underpin the practical implementation of use cases that truly leverage EBRAINS as a multi-service provider with the NRP as a focal point.