Project Number:	284941	Project Title:	Human Brain Project

Document Title:	HBP Applications Research Plan
Document Filename[(1)]:	HBP_SP11_ApplicationsResearchPlan_D11.4.1_Final.docx
Deliverable Number:	D11.4.1
Deliverable Type:	Report
Work Package(s):	WP11.1, WP11.2, WP11.3
Dissemination Level:	PU
Planned Delivery Date:	M6 / 31 March 2014
Actual Delivery Date:	M13 / 31 October 2014

Author:	Karlheinz MEIER, UHEI (P45)	*Subproject Leader, WP11.3*
Compiling Editor:	Martina SCHMALHOLZ, UHEI (P45)	*Compiled Deliverable*
Contributor:	Greg FRANCIS, EPFL (P1)	*WP11.1, T11.1.1*
Contributor:	Florian RÖHRBEIN, TUM (P53	*T11.1.2*
Contributor:	Ferath KHERIF, CHUV (P23)	*WP11.2, T11.2.1*
Contributor:	Frank GOTTFRIED, SAP (P47)	*T11.3.1*
Contributor:	Christian HUYCK, MU (P91)	*T11.3.2*
Contributor:	Bernabe LINARES, CSIC (P82)	*T11.3.3*
Contributor:	Ulrich RÜCKERT, UNIBI (P102)	*T11.3.4*
Contributor:	Ryad BENOSMAN, UPMC (P107)	*T11.3.5*
Contributor:	Thomas NOWOTNY, UOS (P112)	*T11.3.6*
Reviewer:	Richard WALKER, EPFL (P1)	*Review, editing*
Reviewer:	Guy WILLIS, EPFL (P1)	*Review, editing*
Reviewer:	Celia LUTERBACHER, EPFL (P1)	*Review, editing*

Abstract:	This document describes the early applications of the six Human Brain Project Platforms, to be implemented during the Project's Ramp-Up Phase. The structure of the Applications Subproject corresponds to the three main research areas of the HBP, Future Neuroscience, Future Medicine and Future Computing, with one Work Package for each research area. Because these applications are being developed during the Ramp-Up Phase, they will not be able to use fully developed versions of the HBP Platforms. They will, however, influence the final design of the Platforms and make use of their preliminary software and hardware components. As a result, the applications will have to be closely integrated into the Project. After completion of the Ramp-Up Phase, the application work will be transferred to the Partnering Projects envisaged under the Framework Partnership Agreement (FPA), which defines the HBP's Operational Phase.
Keywords:	Early applications, 6 HBP Platforms, Ramp-Up Phase, Future Neuroscience, Future Medicine, Future Computing, use of early software and hardware components, close integration, Framework Partnership Agreement (FPA)

Document Status

Version	Date	Status	Comments
0.1	8 May 2014	Draft	Merging all parts of contributions to one document
0.2	2 Jun 2014	Draft	Added Abstract, Executive Summary Adjust chapter numbering, Numbers of Figures
0.5	28 Jul 2014	Draft	Incorporate editorial review
0.6	30 Jul 2014	Draft	Copy edit, formatting, etc.
0.9	06.10.2014	Draft	Added Function and KPI sections for all Tasks, final formatting, etc.
1.0	16 Oct	Draft	Final copyedit

Table of Contents <Title>

List of Figures, Tables & Equations

1. Executive Summary

This document describes the early applications of the six Human Brain Project (HBP) Platforms, to be implemented during the Project's Ramp-Up Phase. The structure of the Applications Subproject (SP11) corresponds to the three main research areas of the HBP, Future Neuroscience, Future Medicine and Future Computing, with one Work Package for each research area. Because these applications are being developed during the Ramp-Up Phase, they will not be able to use fully developed versions of the HBP Platforms. They will, however, influence the final design of the Platforms and make use of their preliminary software and hardware components. As a result, the applications will have to be closely integrated into the Project. After completion of the Ramp-Up Phase, the application work will be transferred to the Partnering Projects envisaged under the Framework Partnership Agreement (FPA), which defines the HBP's Operational Phase.

The primary objective of the Future Neuroscience Work Package is to provide an initial demonstration of the value of the Neurorobotics Platform for experimental cognitive neuroscience and to provide feedback to refine the design of the Platform. Specifically, the models, frameworks, and ideas of the Human Brain Project are here used to connect theories of brain computing to psychophysical data that measure human performance in terms of the well-known Weber-Fechner Law. We will also prepare a study of different brain or subsystem capabilities such as perception, attention, coordinated movements, core knowledge, spatial cognition, motivation, emotions, and consciousness. We will emphasise how these capabilities can be developed in a robot designed for behavioural experiments.

The HBP's Medical Informatics Platform will federate imaging, genetic and other clinical data currently stored in archives and databases of hospitals and research institutes. It will provide the tools for epidemiological exploration, numerical and statistical analysis, data visualisation and data mining. In the Future Medicine Work Package, we will use these resources to identify unique biological signatures of brain diseases. Specifically, we will characterise the target population of Alzheimer's patients, scale up a pilot study (200 subjects' records) to a large-scale level (c. 2-4,000 subjects' records), apply novel data-mining approaches in High Performance Computing facilities, and demonstrate that rule-based disease signatures provide comprehensive models that explain the variability among patients and aged healthy subjects in a reduced multidimensional space.

Applications in the Future Computing Work Package will prepare for the implementation of neural circuit models on the two hardware systems provided by the Neuromorphic Computing Platform. The initial network architectures are compatible with the constraints given by the HBP hardware architectures described in the SP9 Platform specification document, and will all be formulated in the network description language PyNN. The Work Package has six Tasks corresponding to six application cases. The first Task was part of the original HBP proposal, whereas the other five applications were brought in subsequently via a Competitive Call process during the first six months of the HBP, along with new Partners to implement them. The applications range from the implementation of reverse engineered biological circuits, through applications in computer vision, to the mining of abstract business data.

2. The Applications Subproject

2.1 The Human Brain Project (HBP)

The Human Brain Project (HBP) is a major international scientific research project, involving over 100 academic and corporate entities in more than 20 countries. Funded by the European Commission (EC), the ten-year, EUR 1 billion Project was launched in 2013 with the goal "to build a completely new ICT infrastructure for neuroscience, and for brain-related research in medicine and computing, catalysing a global collaborative effort to understand the human brain and its diseases and ultimately to emulate its computational capabilities."

The fields of neuroscience, medicine and information technology each have important roles to play in addressing this challenge, but the knowledge and data that each is generating have been very fragmented. The HBP is driving integration of these different contributions.

During the Ramp-Up Phase, the HBP will collect strategic data, develop theoretical frameworks, and perform technical work necessary for the development of six Information and Communication Technology (ICT) Platforms during the Operational Phase. The ICT Platforms, offering services to neuroscientists, clinical researchers and technology developers, comprise Neuroinformatics (a data repository, including brain atlases and analysing tools); Brain Simulation (building ICT models and multi-scale simulations of brains and brain components); Medical Informatics (bringing together information on brain diseases); Neuromorphic Computing (ICT that mimics the functioning of the brain); and Neurorobotics (allowing testing of brain models and simulations in virtual environments). A High Performance Computing Platform will support these Platforms.

2.2 Applications Subproject: Overall Goals and Timing

The six HBP Platforms aim to provide unique capabilities for research that would not be otherwise possible. When the HBP was conceived, it was clear that a set of early application ideas should be developed in parallel with the building of the Platforms, and that these early application ideas should make use of preliminary versions of the Platforms' software and hardware. Therefore, the overall goal of Subproject 11 is to prepare, evaluate and test the early applications of the six HBP Platforms. The Subproject is structured into three Work Packages, covering applications of the HBP Platforms in Future Neuroscience, Future Medicine and Future Computing. Because the Platforms are scheduled to become operational at the end of the Project's Ramp-Up Phase (Month 30), the main application work can only take place afterwards, during the Operational Phase, defined by the HBP Framework Partnership Agreement (FPA). This will run from Month 31 to Month 120.

Four new Tasks related to Future Computing were added to the Subproject 11 as a result of the Competitive Call launched in late 2013. The five new Partners working on the new Tasks only joined the Project at the beginning of Month 7, so it was decided to delay finalisation of the SP11 research plan (i.e. this document) to allow them sufficient time to integrate themselves into the Subproject and define their contributions.

2.3 Applications Subproject: Relations to other Platforms

From the overall Subproject goals, it is evident that close collaboration with all six Platform Subprojects is essential. In the area of Future Neuroscience applications, SP11 interacts primarily with the Neuroinformatics Platform (SP5), the Brain Simulation Platform (SP6) and the Neurorobotics Platform (SP10). For Future Medicine applications, SP11 works mainly with the Medical Informatics Platform (SP8), but also with the High Performance Computing Platform (SP7). Finally, for Future Computing applications, SP11 is closely linked to the Neuromorphic Computing Platform (SP9). As a result, SP11 is probably the most “connected” scientific Subproject in the entire HBP.

3. WP11.1: Future Neuroscience

3.1 The Overall Future Neuroscience Research Plan

The objective of this Work Package (WP) is to provide an initial demonstration of the value of the Neurorobotics Platform (NRP) for experimental cognitive neuroscience, and to provide operational feedback on the design of the Platform. In the Ramp-Up Phase, WP11.1 will use the initial capabilities provided by the NRP to perform proof-of-concept simulation-based research on the multi-level brain mechanisms responsible for visual perception. In addition, this work may contribute to that of SP3's Task 3.1.2 (Understanding the circuits linking perceptions to actions, led by Martin Giese at Tübingen University in Germany). More broadly, this WP aims to demonstrate that the cognitive neuroscience community can use HBP-developed technologies to enhance their scientific progress.

WP11.1 comprises the following Tasks:

- T11.1.1 - Psychophysics of perception: the Weber-Fechner law (led by Michael Herzog at the EPFL, Switzerland, and Eduardo Ros at the University of Granada, Spain).
- T11.1.2 – Integrated brain-body control benchmarks (led by Alois Knoll at the Technical University of Munich, Germany).

3.2 T11.1.1 Psychophysics of Perception: the Weber-Fechner law

This Task utilises the models, frameworks, and ideas of the HBP to connect theories of brain computing to psychophysical data that measure human performance. In close cooperation with T11.1.2, it also explores methodologies for benchmarking computational neuroscience models by explicitly comparing specific psychophysical data against the visual pathway. In a sense, the goal is to provide input to the HBP model systems and measure outputs from those model systems for comparison with known properties of biological counterparts. The work is guided by two overarching ideas:

1) The neurophysiological properties of the cortical column identified by the HBP provide some guidance about which kinds of computations can be performed in cortical circuits for visual perception.
2) Given the uncertainties about the properties of the cortical circuits (e.g., the model is derived from data gathered from the mouse barrel cortex), it is appropriate to investigate very robust behavioural data because the model might produce them even if many details are wrong.

Different brain areas behave differently, in part because they receive different sensory information. A good cortical model can only behave properly if it receives appropriately modelled inputs. For visual perception, the sensory system is the retina of the human eye. Embedded within the retina are a variety of complex neural circuits that convert light energy into neural responses. These neural responses ultimately project to cortical areas as action potentials. The development of a good retinal model is thus crucial to understanding the behaviour of the cortical model and to properly relate model properties to human behaviour.

The work involves identification, construction, and testing of models of the retina and visual cortex. As much as possible, these models are built using the NEST software program, which is also used in other areas of the HBP.

3.3 Software/Hardware Functions (Components of Task 11.1.1)

Development of retina model simulation started at the beginning of the HBP, and is expected to continue for several months. The simulator is conceived as a configurable software system that can be used to carry out physiological reproductions of different retina models. Internal dynamics of neurons are based on published models of a Virtual Retina from the INRIA Sophia Antipolis-Méditerannée group. The goal is to implement and validate different retinal models that can run in the NEST software. The retinal simulator part of the Task breaks down into several key functions:

Task No:	T11.1.1	Partner:	UGR
Function No:	11.1.1.1	Leader:	Eduardo Ros
Function Name:	Identification of retinal model properties		
Use Case A:	Model developers know which retinal models to implement for retina simulator		
Planned Start Date:	October 2013	Planned Completion Date:	December 2013
Requires Functions:			

Task No:	T11.1.1	Partner:	UGR
Function No:	11.1.1.2	Leader:	Eduardo Ros
Function Name:	Adaptation of the INRIA models to the retina simulator		
Use Case A:	Model runs in simulator and produces neural output		
Planned Start Date:	January 2014	Planned Completion Date:	April 2014
Requires Functions:	11.1.1.1		

Task No:	T11.1.1	Partner:	UGR
Function No:	11.1.1.3	Leader:	Eduardo Ros
Function Name:	Connection of the retina simulator to NEST		
Use Case A:	Model runs in NEST and produces neural output		
Planned Start Date:	April 2014	Planned Completion Date:	May 2014
Requires Functions:	11.1.1.1, 11.1.1.2		

Task No:	T11.1.1	Partner:	UGR
Function No:	11.1.1.4	Leader:	Eduardo Ros
Function Name:	Refinement of the INRIA models in the retina simulator		
Use Case A:	Performance improves, bugs are fixed		
Planned Start Date:	May 2014	Planned Completion Date:	June 2014
Requires Functions:	11.1.1.1, 11.1.1.2, 11.1.1.3		

Task No:	T11.1.1	Partner:	UGR
Function No:	11.1.1.5	Leader:	Eduardo Ros
Function Name:	Validation of Simulation Platform against existing neurophysiological data		
Use Case A:	Model matches known neurophysiological properties		
Use Case B:	Understanding reasons for mismatches with known neurophysiological properties		
Planned Start Date:	May 2014	Planned Completion Date:	November 2014
Requires Functions:	11.1.1.1, 11.1.1.2, 11.1.1.3		

Task No:	T11.1.1	Partner:	UGR
Function No:	11.1.1.6	Leader:	Eduardo Ros
Function Name:	Connection of the retinal model to LGN model and then to cortical model		
Use Case A:	Model expanded to include LGN processing		
Use Case B:	Model connects to cortical model.		
Planned Start Date:	May 2014	Planned Completion Date:	November 2014
Requires Functions:	11.1.1.1, 11.1.1.2, 11.1.1.3, 11.1.1.9		

Task No:	T11.1.1, T11.1.2	Partner:	UGR, TUM
Function No:	11.1.1.7	Leader:	Eduardo Ros, Florian Röhrbein
Function Name:	Integration of retinal model with simulation environment developed in SP10		
Use Case A:	Model receives input from simulation environment.		
Planned Start Date:	December 2014	Planned Completion Date:	March 2016
Requires Functions:	11.1.1.0, 11.1.1.1, 11.1.1.2, 11.1.1.3, 11.1.1.9		

The second major part of the Task is to develop and implement a model of visual areas of cortex. As a starting point, a version of the published LAMINART model will be implemented in NEST. The cortical model part of the Task breaks down into several key functions:

Task No:	T11.1.1	Partner:	EPFL
Function No:	11.1.1.8	Leader:	Michael Herzog
Function Name:	Identification of cortical model properties		
Use Case A:	Model developers know which cortical model to implement.		
Planned Start Date:	October 2013	Planned Completion Date:	April 2014
Requires Functions:			

Task No:	T11.1.1	Partner:	EPFL
Function No:	11.1.1.9	Leader:	Michael Herzog
Function Name:	Implementation of the LAMINART model in NEST		
Use Case A:	Simulation runs in simulator and produces model neural output.		
Planned Start Date:	April 2014	Planned Completion Date:	September 2014
Requires Functions:	11.1.1.8		

Task No:	T11.1.1	Partner:	EPFL
Function No:	11.1.1.10	Leader:	Michael Herzog
Function Name:	Refinement of the LAMINART model in NEST		
Use Case A:	Performance improves; bugs are fixed.		
Planned Start Date:	September 2014	Planned Completion Date:	November 2014
Requires Functions:	11.1.1.8, 11.1.1.9		

Task No:	T11.1.1	Partner:	EPFL
Function No:	11.1.1.11	Leader:	Michael Herzog
Function Name:	Validation of NEST implementation computational properties against model aims		
Use Case A:	Model behaviour shown to match previously published simulations		
Use Case B:	Understanding reasons for mismatches with previously published simulations		
Planned Start Date:	October 2014	Planned Completion Date:	December 2014
Requires Functions:	11.1.1.8, 11.1.1.9, 11.1.1.10		

Task No:	T11.1.1	Partner:	EPFL
Function No:	11.1.1.12	Leader:	Michael Herzog
Function Name:	Model refinement to match HBP cortical column statistics		
Use Case A:	Model behaviour shown to match columnar statistics		
Use Case B:	Model revised to agree with columnar statistics		
Use Case C:	Model concluded to be incompatible with columnar statistics		
Planned Start Date:	October 2014	Planned Completion Date:	March 2016
Requires Functions:	11.1.1.8, 11.1.1.9, 11.1.1.10, 11.1.1.11		

The third part of the Task's research plan is to use the model to account for behavioural data that have previously been measured with psychophysical experiments. For each function, the strategy is to identify relevant psychophysical measurements, generate equivalent visual stimuli for input to the model retina, and identify appropriate model measures for comparison with the psychophysical data. When possible, the model behaviour will also be compared with published neurophysiological measurements. These investigations will be run in parallel as much as possible. By necessity, Functions 1, 2, 5, 6, 8 and 9 must be completed before Functions 13-20, but there is no required order for the

other Tasks. When the simulation environment in SP10 becomes available, Function 7 will be initiated and Functions 13-20 will make use of the environment.

Task No:	T11.1.1	Partner:	EPFL
Function No:	11.1.1.13	Leader:	Michael Herzog
Function Name:	Model emulation of Weber's law for brightness perception		
Use Case A:	Model matches experimental data		
Use Case B:	Reasons for non-matching experimental data are understood.		
Planned Start Date:	October 2014	Planned Completion Date:	May 2015
Requires Functions:	11.1.1.1, 11.1.1.2, 11.1.1.3, 11.1.1.8, 11.1.1.9, 11.1.1.10, 11.1.1.11		

Task No:	T11.1.1	Partner:	EPFL
Function No:	11.1.1.14	Leader:	Michael Herzog
Function Name:	Model emulation of Weber's law for line length		
Use Case A:	Model matches experimental data		
Use Case B:	Reasons for non-matching experimental data are understood		
Planned Start Date:	October 2014	Planned Completion Date:	May 2015
Requires Functions:	11.1.1.1, 11.1.1.2, 11.1.1.3, 11.1.1.8, 11.1.1.9, 11.1.1.10, 11.1.1.11		

Task No:	T11.1.1	Partner:	EPFL
Function No:	11.1.1.15	Leader:	Michael Herzog
Function Name:	Model emulation of Bloch's law for brightness perception.		
Use Case A:	Model matches experimental data.		
Use Case B:	Reasons for not matching experimental data are understood.		
Planned Start Date:	October 2014	Planned Completion Date:	March 2016
Requires Functions:	11.1.1.1, 11.1.1.2, 11.1.1.3, 11.1.1.8, 11.1.1.9, 11.1.1.10, 11.1.1.11		

Task No:	T11.1.1	Partner:	EPFL
Function No:	11.1.1.16	Leader:	Michael Herzog
Function Name:	Model emulation of illusory contours		
Use Case A:	Model matches experimental data		
Use Case B:	Reasons for non-matching experimental data are understood		
Planned Start Date:	October 2014	Planned Completion Date:	March 2016
Requires Functions:	11.1.1.1, 11.1.1.2, 11.1.1.3, 11.1.1.8, 11.1.1.9, 11.1.1.10, 11.1.1.11		

Task No:	T11.1.1	Partner:	EPFL
Function No:	11.1.1.17	Leader:	Michael Herzog
Function Name:	Model emulation of brightness contrast		
Use Case A:	Model matches experimental data.		
Use Case B:	Reasons for not matching experimental data are understood.		
Planned Start Date:	October 2014	Planned Completion Date:	March 2016
Requires Functions:	11.1.1.1, 11.1.1.2, 11.1.1.3, 11.1.1.8, 11.1.1.9, 11.1.1.10, 11.1.1.11		

Task No:	T11.1.1	Partner:	EPFL
Function No:	11.1.1.18	Leader:	Michael Herzog
Function Name:	Model emulation of visual persistence		
Use Case A:	Model matches experimental data.		
Use Case B:	Reasons for not matching experimental data are understood.		
Planned Start Date:	October 2014	Planned Completion Date:	March 2016
Requires Functions:	11.1.1.1, 11.1.1.2, 11.1.1.3, 11.1.1.8, 11.1.1.9, 11.1.1.10, 11.1.1.11		

Task No:	T11.1.1	Partner:	EPFL
Function No:	11.1.1.19	Leader:	Michael Herzog
Function Name:	Model emulation of backward masking		
Use Case A:	Model matches experimental data.		
Use Case B:	Reasons for not matching experimental data are understood.		
Planned Start Date:	October 2014	Planned Completion Date:	March 2016
Requires Functions:	11.1.1.1, 11.1.1.2, 11.1.1.3, 11.1.1.8, 11.1.1.9, 11.1.1.10, 11.1.1.11		

Task No:	T11.1.1	Partner:	EPFL
Function No:	11.1.1.20	Leader:	Michael Herzog
Function Name:	Model emulation of visual afterimages		
Use Case A:	Model matches experimental data.		
Use Case B:	Reasons for not matching experimental data are understood.		
Planned Start Date:	October 2014	Planned Completion Date:	March 2016
Requires Functions:	11.1.1.1, 11.1.1.2, 11.1.1.3, 11.1.1.8, 11.1.1.9, 11.1.1.10, 11.1.1.11		

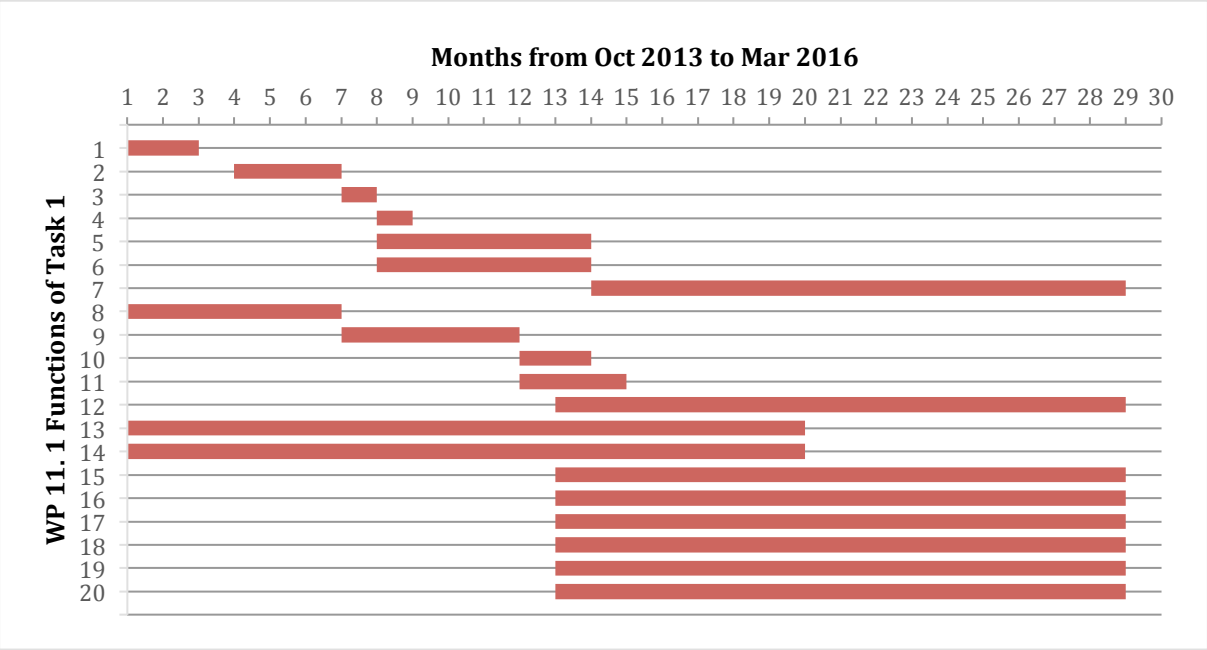

Figure 1: Timeline of Functions in Task 11.1.1

3.4 T11.1.2 Integrated Brain-Body Control Benchmarks

This Task will study selected brain capabilities such as perception, coordinated movements and spatial cognition. It will specifically look at how these capabilities can be replicated in behavioural experiments using robots in closed-loop systems that link brain simulations to simplified demonstrators and/or virtual environments. In the beginning, a robotic arm with limited degrees of freedom will serve as a demonstrator. As soon as we arrive at more complex set-ups, quantitative data on behaviour and cognitive performance can serve as benchmarks for the validation of future brain simulations and simplified brain models. In the long run, Task 11.1.2 will provide benchmarks for replicating classical cognitive neuroscience experiments using neurorobots, and will make comparisons between artificial and living animals. The results of these experiments will help validate brain/cognition models, and explore the structure and function of cognitive abilities for use in robots and other devices. More specifically, this Task will:

- Collaborate with the Cognitive Neuroscience Subproject (SP3) in designing and performing cognitive neuroscience experiments.
- Initially use realistic, physics-based simulations and virtual robots to perform the experiments, and then focus on physical robots after the Ramp-Up Phase.
- Support the construction of experimental setups of increasing complexity, starting with simple experiments that explore and validate cognitive models of perception.

For the Ramp-Up Phase, a musculoskeletal toolkit will be chosen as a robotic platform. It should provide a modular, reconfigurable design based on variable-stiffness joints with motors, links, and sensors for joint position and torque. We will select a real-time model of the cerebellum as a motor controller, and port it to PyNN so that it can be used as front-end for neuronal simulators, including the SpiNNaker system (SP9).

The initial setup will be designed to control a robot with only one joint to follow a simple trajectory, in which the emulated cerebellum can take over control. The cerebellum model should run in real-time on the SpiNNaker system enabling a closed loop. After this

proof-of-concept implementation, we will scale up the robot kinematic complexity including more hardware parts.

3.5 Software/Hardware Functions (Components of Task 11.1.2)

Development of retina model simulation started at the beginning of the HBP, and is expected to continue for several months. The simulator is conceived as a configurable software system that can be used to carry out physiological reproductions of different retina models. Internal dynamics of neurons are based on published models of a Virtual Retina from the INRIA Sophia Antipolis-Méditerannée group. The goal is to implement and validate different retinal models that can run in the NEST software. The retinal simulator part of the Task breaks down into several key functions:

Task No:	T11.1.2	Partner:	TUM
Function No:	11.1.2.1	Leader:	Alois Knoll
Function Name:	Development of suitable set of benchmarks		
Use Case A:	Contribution to MS200 “Sensor and motor models” reached		
Planned Start Date:	January 2014	Planned Completion Date:	March 2014
Requires Functions:			

Task No:	T11.1.2	Partner:	TUM
Function No:	11.1.2.2	Leader:	Alois Knoll
Function Name:	Development of suitable set of benchmarks		
Use Case A:	Contribution to MS201 “Initial experimental design” reached		
Planned Start Date:	April 2014	Planned Completion Date:	September 2014
Requires Functions:	11.1.2.1		

Task No:	T11.1.2	Partner:	TUM
Function No:	11.1.2.3	Leader:	Alois Knoll
Function Name:	Development of suitable set of benchmarks		
Use Case A:	Contribution to MS202 “Robot, environment and experiment implemented” reached		
Planned Start Date:	October 2014	Planned Completion Date:	March 2015
Requires Functions:	11.1.2.2		

Task No:	T11.1.2	Partner:	TUM
Function No:	11.1.2.4	Leader:	Alois Knoll
Function Name:	Development of suitable set of benchmarks		
Use Case A:	Contribution to MS203 “First experiment completed” reached		
Planned Start Date:	April 2015	Planned Completion Date:	March 2016
Requires Functions:	11.1.2.3		

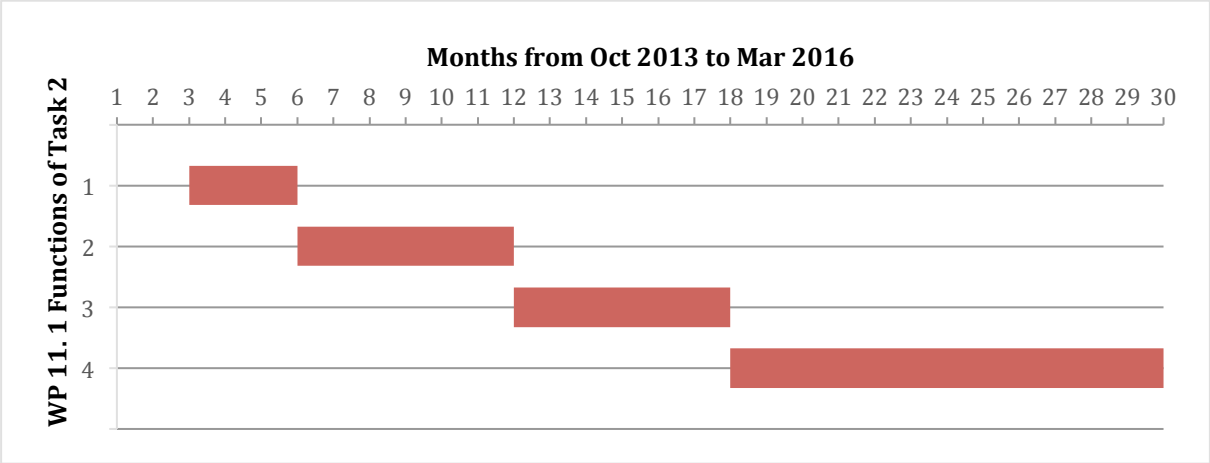

Figure 2: Timeline of Functions in Task 11.1.2

3.6 Scientific Key Performance Indicators for WP11.1

Progress will be measured by assigning a "status" to each function in the research plan. Functions involving the development of model frameworks require more work, and so they are assigned a larger number of statuses than functions that utilise those frameworks to explore relations between the models and human behaviour. Table 1 summarises the possible statuses for each function.

Function	Function Name	Possible KPI statuses	Current KPI status
11.1.2.1	Development of suitable set of benchmarks	1. Initial set of benchmarks 2. Refined set of benchmarks	1
11.1.1.1	Identification of retinal model properties	1. Candidates considered 2. Properties selected	2
11.1.1.2	Adaptation of the INRIA models to the retina simulator	1. Define simulator architecture 2. Define retinal structure 3. Construct model neurons	3
11.1.1.3	Connection of the retina simulator to NEST	1. Define simulator NEST interface 2. Define NEST retina interface	2
11.1.1.4	Refinement of the INRIA models in the retina simulator	1. Speed up code 2. Identify and fix bugs	0
11.1.1.5	Validation of the Simulation Platform against existing neurophysiological data	1. Identify benchmark neurophysiological data 2. Create model stimuli 3. Run simulation on model stimuli 4. Compare model behaviour to data 5. Refine model as needed	0
11.1.1.6	Connection of the retinal model to a LGN model and then to the cortical model	1. Define LGN properties 2. Create LGN simulator 3. Develop interface for retina to LGN models 4. Develop interface for LGN to cortical models	0
11.1.1.7	Integration of the retinal model with the simulation environment being developed in SP10	1. Define retina interface 2. Verify that stimuli give appropriate model responses	0
11.1.1.8	Identification of cortical model properties	1. Candidates considered 2. Properties selected	2
11.1.1.9	Implementation of the LAMINART model in NEST	1. Define cortical column architecture as pixels 2. Define neural layers within a column 3. Define individual neurons within a layer 4. Connect neurons within and between layers	0
11.1.1.10	Refinement of the LAMINART model in NEST	1. Speed up code 2. Identify and fix bugs	0
11.1.1.11	Validation of the computational properties of the NEST implementation with regard to the model aims	1. Identify key computational properties. 2. Run model for appropriate stimulus inputs 3. Compare model behaviour to desired behaviour	0
11.1.1.12	Model refinement to match HBP cortical column statistics	1. Identify useful HBP statistics 2. Measure statistics in current model 3. Compare HBP statistics and model statistics 4. Refine model as appropriate	0
11.1.1.13	Model emulation of Weber's law for brightness perception	1. Identify representative empirical measure of effect 2. Identify stimuli to induce the effect 3. Construct simulated stimuli 4. Identify model behaviour that corresponds to experimental measurement 5. Run model with simulated stimuli 6. Measure model behaviour 7. Compare model behaviour to empirical data	3

Function	Function Name	Possible KPI statuses	Current KPI status
11.1.1.14	Model emulation of Weber's law for line length	1. Identify representative empirical measure of effect 2. Identify stimuli to induce the effect 3. Construct simulated stimuli 4. Identify model behaviour that corresponds to experimental measurement 5. Run model with simulated stimuli 6. Measure model behaviour 7. Compare model behaviour to empirical data	2
11.1.1.15	Model emulation of Bloch's law for brightness perception	1. Identify representative empirical measure of effect 2. Identify stimuli to induce the effect 3. Construct simulated stimuli 4. Identify model behaviour that corresponds to experimental measurement 5. Run model with simulated stimuli 6. Measure model behaviour 7. Compare model behaviour to empirical data	0
11.1.1.16	Model emulation of illusory contours	1. Identify representative empirical measure of effect 2. Identify stimuli to induce the effect 3. Construct simulated stimuli 4. Identify model behaviour that corresponds to experimental measurement 5. Run model with simulated stimuli 6. Measure model behaviour 7. Compare model behaviour to empirical data	0
11.1.1.17	Model emulation of brightness contrast	1. Identify representative empirical measure of effect 2. Identify stimuli to induce the effect 3. Construct simulated stimuli 4. Identify model behaviour that corresponds to experimental measurement 5. Run model with simulated stimuli 6. Measure model behaviour 7. Compare model behaviour to empirical data	0
11.1.1.18	Model emulation of visual persistence	1. Identify representative empirical measure of effect 2. Identify stimuli to induce the effect 3. Construct simulated stimuli 4. Identify model behaviour that corresponds to experimental measurement 5. Run model with simulated stimuli 6. Measure model behaviour 7. Compare model behaviour to empirical data	0
11.1.1.19	Model emulation of backward masking	1. Identify representative empirical measure of effect 2. Identify stimuli to induce the effect 3. Construct simulated stimuli 4. Identify model behaviour that corresponds to experimental measurement 5. Run model with simulated stimuli 6. Measure model behaviour 7. Compare model behaviour to empirical data	0

Function	Function Name	Possible KPI statuses	Current KPI status
11.1.1.20	Model emulation of visual afterimages	1. Identify representative empirical measure of effect 2. Identify stimuli to induce the effect 3. Construct simulated stimuli 4. Identify model behaviour that corresponds to experimental measurement 5. Run model with simulated stimuli 6. Measure model behaviour 7. Compare model behaviour to empirical data	0
11.1.2.1	Development of suitable set of benchmarks for MS200	Set of benchmarks	1
11.1.2.2	Development of suitable set of benchmarks for MS201	Set of benchmarks	0
11.1.2.3	Development of suitable set of benchmarks for MS202	Set of benchmarks	0
11.1.2.4	Development of suitable set of benchmarks for MS203	Set of benchmarks	0
Total			16

Table 1: KPI Status Values Assigned to the Functions in WP11.1

4. WP11.2: Future Medicine

4.1 The Overall Future Medicine Research Plan

Today, medical researchers lack data and tools to help them understand how to identify the biological mechanisms that explain the complex nature of brain diseases. Current studies on the diagnosis and treatment of brain diseases often result in dead-end research or weak conclusions, because they face great challenges that stem from a failure to comprehend significant population heterogeneity. In addition, diagnosing brain diseases in terms of symptoms and syndromes makes it very difficult to produce correct diagnoses, or even to select patients for clinical trials.

To address these issues, the Medical Informatics Platform of SP8 will federate imaging, genetic and other clinical data that are currently stored in hospital/research archives and databases, but are unavailable to the wider medical research community. SP8 will also provide tools for epidemiological exploration, numerical and statistical analysis, data visualisation and data mining. WP11.2 will use these resources to identify unique biological signatures of brain diseases.

The biological signatures of diseases are deterministic mathematical constructs that describe variability at the phenomenological level (clinical features with symptoms and syndromes) and at the biological level (genetic, proteomic, etc.). The key property of a biological signature of disease is that it accounts for the fact that a symptom of brain dysfunction can be due to many biological causes (one-to-many symptom mapping) and that a biological cause can be present with many symptoms (many-to-one symptom mapping). In reality, the situation is often one of many-to-many mappings between symptoms and biological causes.

With advanced computing power and data-mining, nearly exhaustive searches of a data space can be performed to identify sets of rules that describe homogeneous populations, to explain their biological data, and to predict patterns of symptoms. Biological signatures of diseases result from a continuous, dynamic data-mining process of clinical data in local data sources. These will be used for diagnosis, more accurate prognosis and new approaches to drug discovery for the development of new medicines. The biological signatures of brain diseases will form the basis for a new multi-dimensional brain disease space, facilitating scientific investigation and permitting personalised medicine. The advantage of disease signatures is that they are based on mechanistic, deterministic and predictive rules, as opposed to purely descriptive (phenomenological or clinical) features.

4.2 T11.2.1 Biological Signatures of Diseases

There is a great amount of uncertainty regarding the accuracy of diagnostic classification in the early stages of AD, because of the underlying heterogeneity in etiologies leading to similar phenotypes. Even with common Alzheimer's disease, the clinical syndromic diagnoses are wrong in 20% of cases. Research in this Task is designed to show that with modern mathematics, powerful information technology, and a large data set, it is possible to identify homogeneous groups of patients characterised by a set of parameterised latent causes, which constitute what we call "disease signatures". The Task will rely on data that are accessible through the Medical Informatics Platform, including data on the longitudinal study of the large cohort of ‘control’ Alzheimer’s patients. To explain the observed heterogeneity, we will use a rule-based clustering algorithm to identify homogeneous subgroups of patients. The hypothesis is that such subgroups are due to the same underlying causes.

4.2.1 Databases

We will use 329 anonymised MRI scans with ancillary clinical and demographic data on patients with dementia from pharmaceutical trials, and the combined longitudinal data sets of 2-3,000 MRI scans with cognitive scores and associated genotyping single nucleotide polymorphism (SNPs) from the 3C-cohort (France). Data storage and pre-processing will be performed at the Laboratoire de Recherche en Neuroimagerie (LREN) under the direction of Ferath Kherif.

4.2.2 Methodology

Our methods will be based on the state-of-the-art machine learning algorithms. Classical univariate methods based on brain morphometry compare sample means and use 2-sample t-tests to assess the significance of the differences, therefore averaging across subjects and ignoring individual differences. When variability between subjects is high, classical tests will fail to show possible true differences. In addition, classical univariate methods do not take into account the rich sources of information that arise from interactions between brain regions. Identification of different combinations of brain areas that are predictive of disease requires complex analyses that also lie beyond currently used multivariate linear methods (e.g. support vector machines, or SVM). With current neuroimaging methods, such network-based analyses result in an intractable problem because of the combinatorial explosion that occurs when taking into account more than one region at a time. With a total of 90 regions from a standard human atlas (e.g., Automated Anatomical Labelling AAL) in this pilot experiment, the number of possible combinations that involve two regions is $4.5*10^3$; for five regions, it is $5.72*10^{12}$ (for the exact number, see Equation 1 below). Multivariate methods like SVM solve this issue by using kernel approaches, but at the expense of massive data reduction and potential loss of information. By contrast, use of an efficient algorithm and powerful computational resources, such as rule-based clustering, results in exhaustive searches of the data without the same loss of information. The algorithm performs an exhaustive search (100% of the data are explained) to predict multiple inputs. The outputs are explicit formal rules that are easy to interpret and allow computer simulation experiments.

$$C_k^n = \frac{n!}{k!\,(n-k)!}$$

Equation 1: Number of Combinations Involving n Regions

In summary, we will develop a model for the identification of biological signatures of neurological and psychiatric disease, and demonstrate its validity for the case of dementia. The model will be: (1) comprehensive, describing the main characteristics of each disease (at this stage dementia) in the simplest possible form; (2) complex, capturing non-linear interactions, confounding factors and hitherto undocumented and unknown inter-individual differences; and (3) causal, making specific predictions about the mechanisms and time onset of disease. For this purpose, we will:

- Scale up a pilot study (200 subjects' records) to a large-scale level (c.a. 2-4,000 subjects' records) from existing data sources (research databases, hospital data, epidemiological studies, and clinical trial data).
- Apply novel data mining approaches in High Performance Computing facilities to extract disease signatures.

4.2.3 Objectives

- Characterise the target population of dementia patients using brain structure information from magnetic resonance imaging (MRI) as well as genotyping, neuropsychological, biological and clinical measures.
- Demonstrate that the rule-based disease-signatures provide comprehensive models that explain the variability among patients and aged healthy subjects in a reduced set of factors than can be interpreted by clinicians.

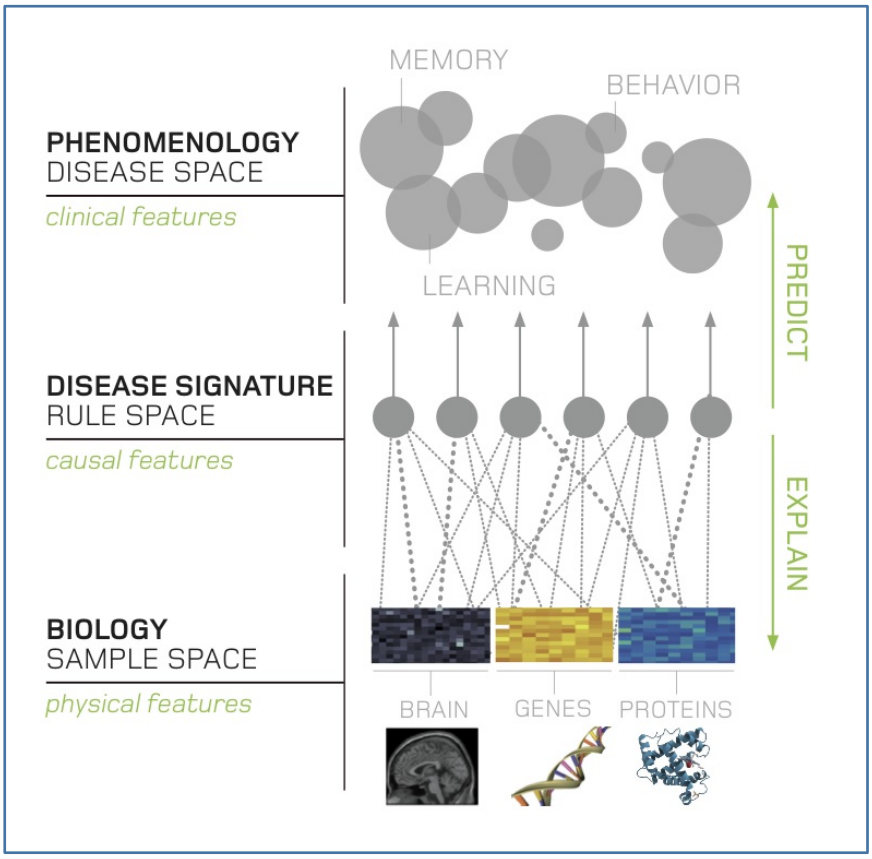

Figure 3: Biological Signature of Brain Diseases/Continuous Data Mining Process

4.3 Software/Hardware Functions (Components of Task 11.2.1)

Task No:	T11.2.1	Partner:	CHUV
Function No:	11.2.1.1	Leader:	Ferath Kherif
Function Name:	Description format for the biological signature of the disease.		
Use Case A:	Feature selection for data mining algorithm		
Use Case B:	Feature description for interpretation of results		
Planned Start Date:	October 2013	Planned Completion Date:	October 2014
Requires Functions:	None		

Task No:	T11.2.1	Partner:	CHUV
Function No:	11.2.1.2	Leader:	Ferath Kherif
Function Name:	Informatics based model for generating biological signature of a disease.		
Use Case A:	Model configuration		
Use Case B:	Model training		
Planned Start Date:	April 2014	Planned Completion Date:	April 2015
Requires Functions:	11.2.1.1		

Task No:	T11.2.1	Partner:	CHUV
Function No:	11.2.1.3	Leader:	Ferath Kherif
Function Name:	Biological signature of major dementia.		
Use Case A:	Internal validation		
Use Case B:	Model prediction		
Planned Start Date:	October 2015	Planned Completion Date:	October 2016
Requires Functions:	11.2.1.2		

Task No:	T11.2.1	Partner:	CHUV
Function No:	11.2.1.4	Leader:	Ferath Kherif
Function Name:	Causal mechanisms for major dementias.		
Use Case A:	Bottom-up generative model		
Use Case B:	Model prediction		
Planned Start Date:	April 2015	Planned Completion Date:	April 2016
Requires Functions:	11.2.1.3		

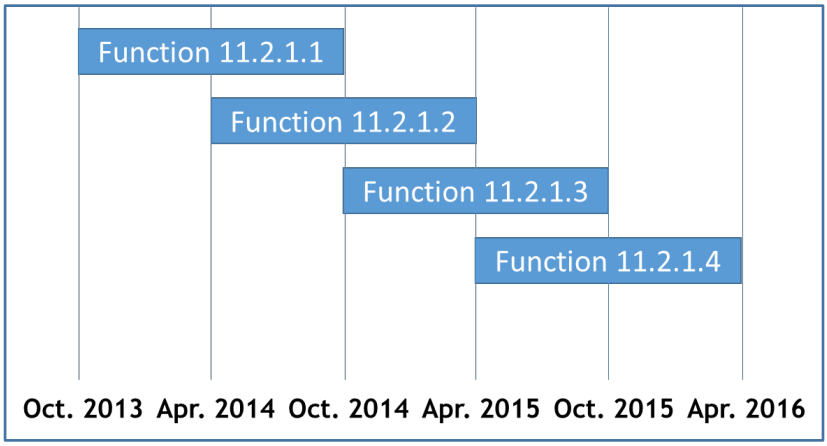

Figure 4: Timeline of Functions in WP11.2

4.4 Scientific Key Performance Indicators for WP11.2

Each function will be implemented according to the timeline given in Figure 4 (above). Progress within a function will be measured by assigning it a "status", as per

Function	Function Name	Possible KPI statuses	Current KPI status	Target Date
11.2.1.1	Description format for the biological signature of the disease	Identify multimodal clinical data Data pre-processing Data aligned Feature selection	3	M3 M6 M9 M12
11.2.1.2	Informatics based model for generating biological signature of a disease	Implement test different algorithms Model configuration Benchmark algorithms Select algorithms	3	M12 M18 M18 M18
11.2.1.3	Biological signature of major dementia	Scale up a pilot study to a large-scale level Apply algorithm Clinical interpretation Internal validation Model prediction	0	M18 M24 M24 M24 M24
11.2.1.4	Causal mechanisms for major dementias	Define multi-scale description of the data Build an *a priori* bio-physiological model Compare model behaviour to data Identify the pathways	0	M24 M24 M30 M30
Totals			6	

Table 2 below.

Table 2: KPI Status Values Assigned to the Functions in WP11.2

5. WP11.3: Future Computing

5.1 The Future Computing Overall Research Plan

Applications in Future Computing will prepare for the implementation of neural circuit models on the hardware systems provided by the Neuromorphic Computing Platform. The initial network architectures are compatible with the constraints given by the HBP hardware architectures, as described in the SP9 Platform specification document, and will all be formulated in the network description language PyNN.

The HBP will offer a unique Neuromorphic Computing Platform to explore the computational capabilities of spiking neural networks. Two complementary hardware systems will be constructed, in Manchester (NM-MC-1) and Heidelberg (NM-PM-1).

The NM-MC-1 system will consist of 0.5 million ARM cores, 18 per chip. The cores provide integer operation capability and each chip has six bi-directional links with a bandwidth for 6 million spikes per second per link. Networks simulated on this system will operate in real time. The NM-PM-1 system will be a physical model based on analogue neurons and synapses with binary, asynchronous and continuous time spike communication. NM-PM-1 will provide 20 individual wafer systems, replicating a total of 4 million neurons and 1 billion synapses. Networks simulated on this system will operate 10,000 times faster than real time.

Both systems are expected to commence hardware operation towards the end of the HBP's Ramp-Up Phase. Preparatory studies to implement network architectures in this Subproject start earlier than this using simulation tools or smaller test set-ups. Initial experiments are restricted to networks with a maximum of a few tens of thousands of neurons and do not yet rely on the availability of spike-timing-depended-plasticity. Special emphasis is given to the specific features of the two complementary systems, i.e. the real-time operation with algorithm-based neural models in NM-MC-1 and the accelerated operation with diverse, analogue neural models in NM-PM-1.

The Work Package has six Tasks, each of which corresponds to a specific application. One Task/application was part of the original HBP proposal:

- T11.3.1 - Neuromorphic data mining systems (led by Frank Gottfried, SAP, Germany)

After selection by independent reviewers through the Competitive Call process, five more Tasks/applications were added. The research groups concerned were not involved in the design and construction of the hardware systems, but they are typical of the future user community.

- T11.3.2 - Port CABot3 to neuromorphic chips and extend (led by Christian Huyck and Michael Butterworth, Middlesex University, UK). CABot3 is an existing simulation system for learning cell assemblies.
- T11.3.3 - Exploitation of feedback in ultra-fast spiking visual architectures (led by Bernabe Linares-Barranco, Instituto de Neurociencias, Valencia, Spain – part of the Consejo Superior de Investigaciones Cientificas or CSIC). This Task will develop spike-based, multi-layer visual circuits for high-level object recognition.
- T11.3.4 - Spiking associative networks for neuromorphic computing system (led by Ulrich Rückert, Bielefeld University, Germany). This Task will implement large-scale associative memory models using spiking neurons.
- T11.3.5 - Asynchronous computational retina (led by Ryad Benosman, Pierre and Marie Curie University, Paris, France). This Task will develop a pure, event-driven visual computation approach that uses precise timing mechanisms to design new computation techniques in visual processing.

- T11.3.6 - Implementing a spiking classifier network on HiCANN (led by Thomas Nowotny, University of Sussex, UK). This Task will implement a scalable spiking neural network for multivariate classification.

The five new groups are part of the Applications Subproject SP11, but it is crucial that they be integrated with the Neuromorphic Subproject (SP9), which provides the tools for their research. For this reason, the groups in WP11.3 take part in all SP9 teleconferences, videoconferences and face-to-face meetings. They have access to all tools developed in SP9 for the operation of the hardware systems and they participate in the training events. Initial feedback has shown that this integration approach is working very well.

5.2 T11.3.1 Neuromorphic Data Mining Systems

Very large, open-ended data streams are generated by real-time transaction systems in various industrial settings such as retail, utilities, surveillance and many more. In contrast to traditional data sets, stream data flow in and out of a computer system, and it may be impossible to store the entire stream due to the large volume. Even when the data are actually stored, multiple scans are getting very expensive so that single-scan analysis methods need to be applied [1].

Within this continuous stream of data, the most important task is the identification of spatial-temporal patterns. Although this has been an active research field for many years, and many researchers in the cognitive and computer sciences have attempted to tackle the problem [2-13], no general-purpose solution is currently available. Therefore, the goal of this activity is to contribute to this active research field through the adaption/modification of existing algorithms. A requirement for the algorithm is the ability to detect hidden patterns and causal relationships. A software implementation with non-spiking neurons first will be developed as a proof-of-concept. Next, the algorithm will be extended to a spiking network system. Finally, the implementation will be migrated to the Neuromorphic Computing Platform provided by the Human Brain Project (HBP). The final proof point will be the ability of the algorithm to handle real-world, business-relevant data sets, and to demonstrate the validity of the Neuromorphic Computing Platform as a tool for implementing a data mining system.

5.2.1 Objectives

The main goal of Task is to demonstrate the validity of the Neuromorphic Computing Platform as a tool for producing and prototyping cognitive devices and systems outside the realm of biology, and in particular, to validate the possibility of developing cognitive business information systems. Basic features of such a cognitive business information system will be the ability to identify and recall spatial-temporal patterns in data streams, and the ability to predict future elements of the sequence.

We will review conventional state-of-the machine learning techniques to identify algorithms that are suitable for implementation in an automated, self-learning computer system that allows for the analysis of (massive) business data. SAP will identify, modify and adapt the algorithm within this Task, while the University of Heidelberg will provide the NM-PM system as part of the HBP.

The major success criterion for this Task will be the demonstration that non-biological data can be processed on the Neuromorphic Computing Platform. We will evaluate several business scenarios as possible test cases. Candidates include business process management/complex event processing, supply chain management, predictive analytics or in-memory database management.

The expected outcome of this activity is a prototypical implementation of an appropriate data-mining algorithm on the Neuromorphic Hardware System, as well as an in-depth understanding of the potential application areas of such a system.

5.2.2 Possible Architecture

We have investigated several different architectures, and most of them address very specific use cases and cannot be easily generalised. A possible architecture inspired by biological principles—like axonal transmission delays [14, 15] and Hebb's learning rule for the creation of cells assemblies—shows promise and has been implemented as a proof-of-concept. The order and ordinal structure of a sequence is reflected in this architecture, as a columnar and horizontal (layered) organisation of cells. The cells form regions and several regions are connected to each other and organised within a hierarchical model. The learning process within the model leads to the establishment of lateral connections between cells (within a region) and feedforward connections between cells of different regions. The learning rule leads to groups of cells (cell assemblies) representing a specific input sequence with appropriate axonal time delays, so that the occurrence of a previously learned sequence triggers the firing of a neuron (group of neurons) of a higher region. This is very similar to the concept of polychronisation described by Izhikevich [2]. The lateral connections between cells within a region are used to establish predictions for the next time step. If a lateral connection becomes active, we expect the cell to be activated by feedforward input in the next time step. A sequence is detected by a number of successful lateral predictions. Sequences in higher layers represent more complex sequences (sequences of sequences), and the output of one layer is the input to the next layer. Therefore, the model implements a separation of different time scales trigged by events from a lower region.

An input layer is randomly connected to the cells of the lowest region. A local winner-take-all (WTA) or a soft WTA method is applied to generate a sparse and distributed representation of the input data within the lowest region. Several authors have previously described this process [6]. The stream of input signals is either 0 or 1.

Several of these building blocks can be connected side-by-side, allowing for parallel learning of multiple dimensions. For instance, the melody and the rhythm of a piece of music can be learned in parallel with connections synchronising the two aspects. The following figure depicts the current scope of the implementation.

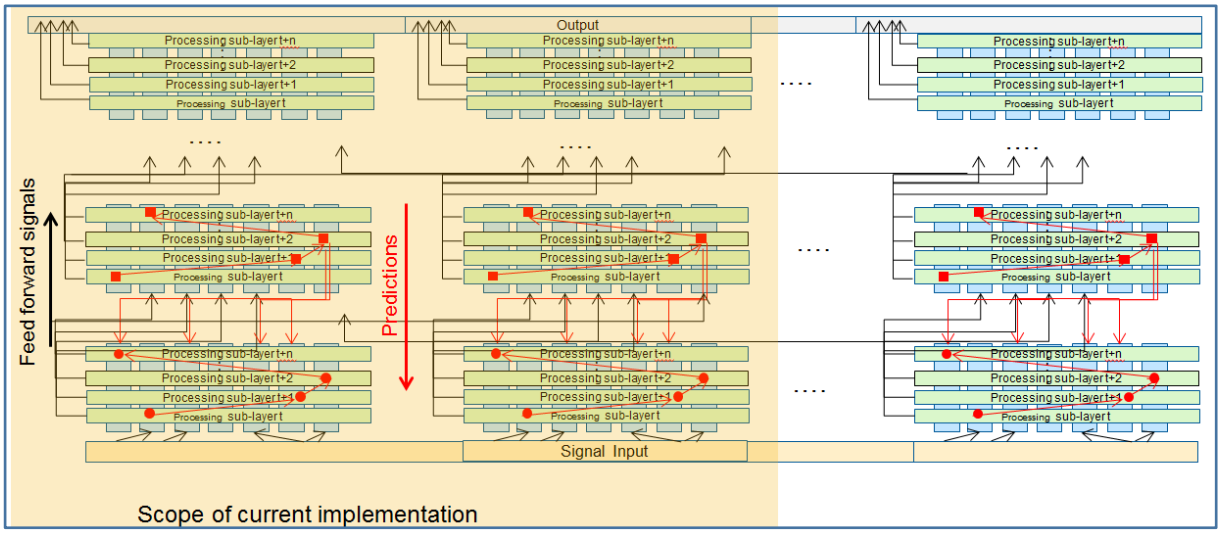

Figure 5: Possible Architecture

The algorithm is currently available as a proof-of-concept software implementation based on static neurons, which rely on internal states rather than on spiking neurons. Preliminary tests with various data streams (time series) have been performed, and initial results look promising. However, compared to existing hardware implementation on the neuromorphic hardware [1], the proposed architecture may turn out to be too complex for hardware implementation during the Ramp-Up Phase, especially since it relies on precise axonal

time delays. Therefore, selected computing motives that constitute building blocks of the proposed architecture will have to be implemented as a first step. We will also investigate alternative architectures, particularly in the area of Hidden Markov Models.

5.2.3 Work Plan

The Work Plan describes the various activities for further implementation of the proposed architecture using non-spiking and spiking cell assemblies, extensive testing, migration to the Neuromorphic Computing Platform and an analysis of the underlying theoretical methods and concepts.

The proposed architecture from the previous section will be further analysed. Applying data sets from a relevant business scenario will allow us to evaluate and assess the potential for future use. In parallel, we will investigate the required adaption and modification of the algorithm necessary for an implementation as a spiking neural network (SNN). The implementation will be performed using the PyNN framework [2]. A key advantage of the PyNN framework is that it provides an interface to the Neuromorphic Computing Platform, and therefore facilitates fast migration to the neuromorphic hardware system.

We will perform a detailed investigation of current alternative approaches. In particular, we will compare the recently developed online approximation schemes with Hidden Markov Models (HMM) using sampling techniques [3, 4]. Other online versions of the Baum-Welch Algorithms will also be included in the investigation. This work will be part of the algorithm development/theory activities. Depending on the outcome of these investigations, major changes and modifications to the proposed architecture may be required.

A total of 30 PM is available for the full duration of the Task. This includes the activities required for the current implementation of the proposed architecture, the comparison to existing state-of-the-art implementation and the activities described in this section. The Task team is staffed with two senior researchers (part-time). A job posting for a master thesis student has been published. At a later stage, an opening for a second master student dedicated to the migration to the Neuromorphic Computing Platform will become available.

5.2.4 Software/Hardware Functions (Components of Task 11.3.1)

The function boxes and timetable below depicts the envisaged start and end dates of the planned activities. These estimates are based on the current status, but are subject to change if new knowledge becomes available.

Task No:	T11.3.1	Partner:	SAP
Function No:	11.3.1.1	Leader:	Frank Gottfried
Function Name:	Evaluation of potential use cases and possible algorithms including PoC software implementations		
Use Case A:	Algorithm identified and possible architecture selected		
Planned Start Date:	October 2013	Planned Completion Date:	Jan 2015
Requires Functions:			

Task No:	T11.3.1	Partner:	SAP
Function No:	11.3.1.2	Leader:	Frank Gottfried
Function Name:	PyNN implementation and migration to Neuromorphic Computer Platform of selected computing motives		
Use Case A:	Architecture runs on simulator package and HW		
Planned Start Date:	Jan 2015	Planned Start Date:	Jan 2015
Requires Functions:	11.3.1.1		

Task No:	T11.3.1	Partner:	SAP
Function No:	11.3.1.3	Leader:	Frank Gottfried
Function Name:	Recruiting of a master student for implementation and testing		
Use Case A:	In-depth analysis performed and potential business applications understood		
Planned Start Date:	Oct 2014	Planned Start Date:	Oct 2014
Requires Functions:			

Task No:	T11.3.1	Partner:	SAP
Function No:	11.3.1.4	Leader:	Frank Gottfried
Function Name:	Evaluation/test with real-world business data		
Use Case A:	Test with relevant data sets can be performed		
Planned Start Date:	October 2015	Planned Start Date:	October 2015
Requires Functions:	11.3.1.2		

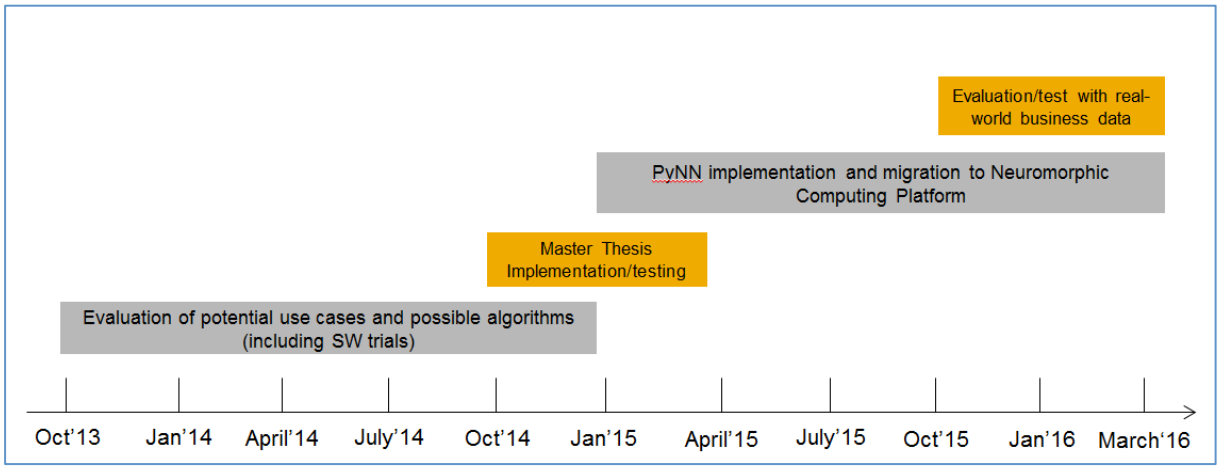

Figure 6: Timeline of Functions in T11.3.1

5.2.5 Scientific Key Performance Indicators for Task 11.3.1

During the Task, we envisage reviewing 2-4 data sets from different business scenarios (some of them mentioned above) as potential application areas. A Key Performance Indicator (KPI) will therefore be the number of data sets evaluated. A second indicator will be the migration of the selected algorithm (or part thereof) to the neuromorphic hardware. Finally, if an appropriate data set is identified and a successful migration to the hardware is performed, a third KPI will be the relative performance improvements compared to available conventional solutions. Details will depend on the specific test case.

5.2.6 T11.3.1 References

[1] Data Mining- Concepts and techniques, Jiawei Han and Micheline Kamber, Elsevier 2006

[2] E. M. Izhikevich. Polychronisation: Computation with Spikes. Neural Computation 18, 245-282 (2006)

[3] D. Kappel, B.Nessler & W. Maass.STDP Installs in Winner-Take-All Circuits an Online Approximation to Hidden Markov Model Learning. PLOS Computational Biology, March 2014, Volume 10, Issue 3.

[4] B. Nessler, M.Peifer, L.Buesing & W. Maass. Bayesian Computation Emerges in Generic Cortical Microcircuits through Spike-Timing-Dependent Plasticity. PLOS Computational Biology, April 2013, Volume 9, Issue 4.

[5] H. Paugam-Moisy, R.Martinez & Samy BengioE.M. Delay learning and polychronisation for reservoir computing. Neurocomputing 71, 1143-1158 (2008)

[6] M. M. Botvinivk & D. C. Plaut. Short-Term Memory for Serial Order: A Recurrent Neural Network Model, Psychological Review, Vol. 113, No. 2, 201-233 (2006)

[7] N. W. Schuck, R. Gaschler & P. A. Frensch. Implicit learning of what comes when and where within a sequence: The time –course of acquiring serial position-item and item-item associations to represent serial order. Advances in Cognitive Psychology, Vol 8(2), 82-97, 2012

[8] W. Maass. On the relevance of time in neural computation and learning. Theoretical Computer Science 261, 157-178 (2001)

[9] J.A. Starzyk & H. He. Anticipation-Based Temporal Sequence Learning in Hierarchical structure. IEEE Transactions on Neural Networks, Vol. 18, No. 2, March 2007.

[10] V.A. Nguyen, J.A. Starzyk, W.-B. Goh & D. Jachyra. Neural Network Structure for Spatio-Temporal long-Term Memory. IEEE Transactions on Neural Networks and Learning Systems, Vol. 23, No. 6, June 2012

[11] Numenta. Hierachical Temporal Memory including HTM Cortical Learning Algorithms, Version 0.2.1, September 12, 2011. Available: http://numenta.org/cla-white-paper.html

[12] B. A. Clegg, G.j. DiGirolamo & S. Keele. Sequence Learning, Trends Cogn Sci. 1998 Aug 1;2(8):275-81.

[13] K.J. Land, A.H. Waibel & G.E.Hinton. A Time-Delay Neural Network Architecture for Isolated Word Recognition. Neural Networks, Vol. 3, pp. 23 -43, 1990

[14] Guo-qiang Bi & Mu-ming Poo. Distributed synaptic modification in neural networks induced by patterned stimulation. Nature, Vol 401, 21 October 1999

[15] S. Xu, W. Jiang, Mu-ming Poo & Y. Dan. Activity Recall in Visual Cortical Ensemble, Nat Neurosci. 2012 January 22; 15(3): 449–S2.

[16] T.Pfeil, A. Grübl, S. Jeltsch, E. Müller, P. Müller, M.A. Petrovici, M. Schmuker, D. Brüderle, J. Schemmel & K. Meier. Six networks on a universal neuromorphic computing substrate. arXiv:1210.7083v4, 21 Feb 2013

[17] Davison AP, Brüderle D, Eppler JM, Kremkow J, Muller E, Pecevski DA, Perrinet L and Yger P (2008) PyNN: a common interface for neuronal network simulators. *Front. Neuroinform.* **2**:11

5.3 T11.3.2 Port CABot3 to Neuromorphic Chips and Extend

NEAL will implement a novel computing paradigm: an embodied agent in spiking neurons on the neuromorphic systems. The earliest phase will use the Neuromorphic Platform to implement a simplified version of a brain model by translating the existing CABot3 system to PyNN, and then onto the neuromorphic chips. This relies on the existing CABot3 code, written in Java, but also on PyNN, and eventually on SpiNNaker and HICANN (SP9).

CABot3 is implemented entirely in our own Fatiguing Leaky Integrate and Fire (FLIF) neural model written in Java. It uses over 100,000 neurons, but simulates the entire system in roughly real-time on a PC, since the discrete time steps correlate with 10ms. CABot3 is broken into 46 subnets, enabling some degree of modularity. We have already implemented the model without fatigue in PyNN. As several of the subnets do not take advantage of fatigue, several subsystems (e.g., the planning subnets) can be implemented immediately. We expect that the fatigue model will be relatively easy to implement in standard PyNN neural models. Moreover, as part of cell assembly (CA) learning, a different fatigue model will be used.

Learning is a similar problem because CABot3 uses both long and short-term synaptic modification (Huyck, 2009). We expect long-term learning to be relatively straightforward, but short-term plasticity may be more of a challenge. Moreover, these challenges may need a different solution for each of the chips. These proposed initial experiments will use up to 100,000 neurons, but this should not exceed the “maximum of a few tens of thousands of neurons” mentioned in the call, because the time constant is so large. Similarly, the proposed system does not take advantage of STDP; learning is based entirely on co-firing in a given 10ms cycle.

The second phase will explore CAs, a generic circuit concept. This will meet the theoretical foundations objective by furthering generic CA models. CAs are a long-standing and well-supported theory linking psychology and biology (Hebb, 1949; Huyck & Passmore, 2013). During the development of CABot3, we showed that a sufficiently large spiking neural network is Turing complete (Byrne & Huyck, 2010), and implemented an embodied agent solely in spiking neurons. During development, it became clear that the difficult question was not how to program a neural system to function; rather, the question was which neural model, network and topology were needed so that it could learn to function. Prior to CABot and since finishing, we have studied learning with these systems. Our earlier learning work (Huyck & Orengo, 2005; Huyck, 2007; Nadh & Huyck, 2012) focused on nets where all neurons were stimulated by the environment. The extension of our FLIF model to allow spontaneous firing from hypo-fatigue has enabled the growth of neural circuits in areas that are not directly stimulated from the environment (Huyck & Mitchell, 2013). This is essentially still a sensory task, since firing stops when the stimulus stops. CABot3 needed CAs to persist mainly for relatively precise times to support natural language parsing dynamics, but also for planning. In CABot3, CAs were mostly programmed by setting synaptic weights. Some progress has been made using small world topologies, so that the CAs persistently fire longer when they are activated more strongly, and when they are reactivated. NEAL will combine these two strands so that CAs, which persist for psychologically realistic times, can be learned.

Neuromorphic Computing Systems Objective:	Achieved by:
Implement novel computing paradigms	Implement an embodied agent in spiking neurons.
Generic circuit concepts of spiking neurons	Develop improved cell assembly models.
Operational Objective Area:	**Achieved by:**
Neuromorphic Platform	Implementing simplified versions of brain models.
Theoretical foundations	Furthering generic CA models.
Brain Simulation Platform	Building point models for simulating brain areas.
Cognitive Architecture	Building neuro-cognitive models to extract principles.

Table 3: Objectives and Approach of T11.3.2

Some of the evaluations of NEAL will meet the cognitive architecture objective by building neuro-cognitive models to extract generic principles. CABot3 already has a cognitive model of parsing and one of rule learning. The new simulations will learn novel categories of visual items from the environment. There will be a range of 3D shapes with a range of visual textures. These will initially be linked to labels, but later they will be learned in an entirely unsupervised fashion. Categories will have a special behaviour in the environment. We have a neuro-cognitive model of a rule-learning system that makes use of reinforcement learning from environmental feedback (Belavkin & Huyck, 2010), and this mechanism will be used to determine the categories. A classic experiment (Shepard, Hovland, & Jenkins, 1961) will be used as a cognitive test.

CABot3 plans to use a Maes net (Maes, 1989) that has been programmed (in FLIF neurons), but NEAL will learn new plans. Initially, we will accomplish this by integrating our reinforcement mechanism with the Maes net, and by direct user instruction. The existing net is made of orthogonal CAs (sharing no neurons), and the reinforcement mechanism will modify the strength of associations between CAs. The user instructions will create new CAs. An overlapping model will then replace this approach, in which basic units will be learned associating goals, actions, and facts from the environment. This will include extra subnets to generate and evaluate plans supporting the generation of more complex plans. NEAL will also take advantage of the larger number of neurons. We are looking for long-term systems that last for days, and that learn throughout. This is a developmental neuro-psychology problem. A comparison of the two systems using STDP combined with a compensatory mechanism for the CA formation problem is particularly promising. The variability of analogue neurons and synapses may lead to particularly powerful attractors, and CAs are attractors.

The system meets an objective of the Brain Simulation Platform: it is a brain model based on a point model level of description. The Task will contribute to the HBP by providing an extensible agent that can be used in a 3D environment, and by providing advancements for the HBP's neuromorphic systems to learn CAs. The agent will be useful for researchers to use during and after the Ramp-Up Phase. It will be a working embodied agent in a simulated environment, implemented entirely in simulated neurons, that provides a modifiable early link in the Project between robotic systems, cognitive architectures, brain data, and neuromorphic hardware. It will provide existing neural language, vision, planning, and action modules, which have a reasonable degree of modularity. Improved models of CA learning will provide insight into the theoretical problem of concept formation in neural systems. This will be linked to the environment, psychology and known and posited neural behaviour, leading to a significant impact in both the short and longer term.

CA learning really is open-ended research. We will develop a working system that generalises; we will explore the space of working systems, and work in promising areas. We will explore ranges of options by exploiting the variability of both analogue and discrete neurons and synapses. We will interact with the HBP community to learn what is neuro-psychologically plausible, and what can be implemented.

5.3.1 Methodology and Associated Work Plan

The overall strategy is to build working systems. Initially, we will build agents functioning in an environment (sub-Task 1 (ST1) and ST2). Later, we will expand these agents (ST3 and ST4) so that they can learn more about their environment, and thus be more effective in that environment. While doing this, working agents will perform tasks as cognitive models; this will have some correlation with biological data, though anything that is particularly solid may be beyond the scope of this Task.

We have discussed SpiNNaker with Furber for several years now, and have planned on putting our model onto it from our first conversation. Consequently, ideas of working on SpiNNaker are more fully formed than those for working on HICANN, and though the plan reflects this bias, we are confident that the agent will run on HICANN, and that it will provide useful variance for exploration of CA dynamics.

NEAL consists of one Task that is broken into four sub-Tasks (see Table 4 below).

5.3.1.1 Sub-Task 1: Transfer FLIF model to PyNN and neuromorphic chips

We have already translated the Leaky Integrate and Fire component of our neural model to PyNN, and will immediately begin translating some of the CABot3 components (e.g. early vision and planning) to PyNN. Later, we will add the fatigue component to the model. A standard PyNN model should be sufficient. There are two variants of fatigue, and only the first, waypoint 1.1 (M1.1), needs to be implemented for CABot3. The second will be translated after Month 9.

Learning will be included in the system. PyNN has good support for LTP, and it will be included in the first few months. A critical component of the CABot3 model is binding via STP (M1.2). This should work in PyNN, and all basic neural and synaptic components will run on both chips (M1.3).

Finally, parts of CABot3 require precise timing and it is not clear how readily this will translate to either chip. We are hopeful that the 10ms integration constant will solve the problems, but we may need to increase the number of neurons for CABot3. Even if CABot3 does not require more neurons, there is scope for exploration of the dynamics and robustness of neural processing circuits with more neurons. That is, by using more neurons, the systems will be more effective, and cope with more hardware failures.

5.3.1.2 Sub-Task 2: Transfer CABot3

CABot3 runs in a virtual 3D environment with a mobile avatar in the environment. Input from the environment is a pixel image from the avatar's camera, and text commands are issued from a user. We will begin by translating the vision, control, and planning subnets of CABot3 into PyNN. These are not dependent on real-time behaviour, or on the fatigue model. We will integrate the environment via Python, and the agent will provide both input to and symbolic motion output from the PyNN based CABot3 (M2.1).

LTP and STP primitives will be implemented as the fatigue (ST1). The language and learning subnets will be integrated into the PyNN agent, then into the SpiNNaker agent (M2.2), and finally into the HICANN agent. All three agents will be tested and compared. Tests will include the ability to parse commands, view the environment, build a simple spatial cognitive map of the environment, and learn which rule is a correct rule. These will be compared with the existing Java-based system and each other. We expect that all will behave almost identically; the neuromorphic agents could use extra neurons.

5.3.1.3 Sub-Task 3: Learn CAs

We will use the PyNN model to learn simple categories, and then use the modified fatigue model to include spontaneous activation from the new fatigue model. This will bring us to the current state of the Java model on PyNN and the chips. We will then add new subnets replacing the higher-level vision subnets, and the system will learn the visual categories by exploring. Instances of categories will be co-presented with labels so the system can learn words. Labelling provides a ready test and links to the language system.

We will then develop a system that learns CAs that persist for the time that psychological short-term memories persist (M3.1). If there is more evidence for a CA, it will persist longer. If a CA is reactivated, by for instance being presented a gain, it will persist longer the second time. Our current idea is that our existing sensory-like learning mechanism can be integrated with subnets that support persistence and top-down effects – an extension of the tripartite theory. Persistence will be compared to the ACT-R memory model. A new environment will then be used; it is quite easy to develop new 3D environments using video game technology. There will be more sophisticated tasks, such as searching for particular types of objects to fulfil current goals or needs. Environmental feedback will enable the system to learn things that will help its performance. We will test this on categories with varying degrees of feature overlap (Shepard, Hovland, & Jenkins, 1961) to allow a cognitive test (M3.2). Its performance in the environment will also improve.

5.3.1.4 Sub-Task 4: Learn Plans

We will change associations within Maes net elements to improve an existing plan driven by reinforcement learning. This will involve the modification of associations between existing CAs, which implement the Maes net. Similarly, new elements within the net will be created and linked in response to user commands, and the associations between these will be modified to improve the overall plan. Together, these constitute M4.1.

Next, we will learn overlapping CAs for plans. The earlier plans, including the learned plans, will use CAs that do not share neurons. The new plan will take advantage of shared neurons to learn more sophisticated and neuro-psychologically plausible plans. This will be driven by separate plan generation subnets, and we will consider turning learning on and off by neuromodulators. This is a speculative venture, so it does not have a waypoint. The system will be tested on the new 3D environment, which is not a cognitive test (M3.3 and M4.2).

5.3.2 Waypoint Descriptions (see Table 4 below)

Waypoints are directly linked to Tasks, and proceed sequentially through the Task. ST1 has three waypoints. M1.1 is to put the CABot3 FLIF model onto PyNN by Month 2. M1.2 is to include the STP model in PyNN by Month 6. M1.3 is to have both FLIF models, and long and short-term synaptic modification on PyNN and both chips. M1.1 and M1.2 are necessary to move the Task forward; M1.3 is more difficult but is also largely non-blocking.

ST2 has two waypoints. M2.1 is a simple version of the agent in PyNN by month 3; this will support movement onto SpiNNaker and HICANN. M2.2 has the complete agent on SpiNNaker via PyNN; it takes advantage of the first Task. CABot3 will then be implemented on HICANN.

The largest sub-Task is ST3: Learning CAs. This is much more exploratory than ST1 and ST2, but at least these three waypoints will be met. M3.1 is properly persistent learned categories. M3.2 is a cognitive model of a classification task, and M3.3 improves the overall agent's performance.

Sub-Task ST4 is learning plans, and thus is learning process. M4.1 is relatively straightforward, porting our existing reinforcement learning to the Maes net, and responding to user commands to add new elements to that net. M4.2 is a catchall for the

Task, allowing the Task to wrap up with one agent, or a variant for each chip; this could include the more sophisticated planning system.

The functions map to these sub-Tasks in a straightforward manner. Functions 11.3.2.1, 11.3.2.2 and 11.3.2.5 are part of ST1. Functions 11.3.2.3, 11.3.2.4, 11.3.2.6, and 11.3.2.8 are part of ST2; the key function is 11.3.2.6 is getting CABot3 running on SpiNNaker. Once the agent is running on SpiNNaker in a closed loop we can explore the system using many more neurons. Functions 11.3.2.7, 11.3.2.9, 11.3.2.10, 11.3.2.11, 11.3.2.12, 11.3.2.13 and 11.3.2.15 will all be used to help us develop systems that learn better cell assemblies, ST3. Knowledge gained along the way will inform ST4, but functions 11.3.2.14 and 11.3.2.16 are fully in ST4 - learning plans.

Objectives
Provide an embodied cognitive agent in spiking neurons on SpiNNaker and HICANN that can be readily modified and extended.
Extend the agent to learn environmentally useful and neuro-psychologically realistic Cell Assemblies.
Description of Work and Role of the Partners
ST1: Transfer FLIF model to PyNN and neuromorphic chips (Months 1-12) Middlesex (MU) will execute all. T1.1 will take 4 person months, and use Huyck and the RA. Translate both variants of the fatiguing FLIF model to PyNN and SpiNNaker and HICANN. Integrate LTP, and STP models with PyNN and the chips.
ST2: Transfer CABot3 (Months 1-9) MU: T1.2 will take 4 person months, and use the RA, Mitchell, and Huyck. Take existing CABot3 Java code and move portions based on LIF model to PyNN. Integrate the 3D environment with PyNN and the chips. Translate the full CABot3 system to PyNN and chips. As soon as the chips become available, the PyNN agent will be integrated with SpiNNaker, and then with HICANN.
ST3: Learn CAs (Months 6-24) MU: The most time will be spent on this Sub-Task, requiring 7 person-months by Huyck and the RA. Explore the short and long-term dynamics of CA persistence and creation. Use compensatory Hebbian learning with subnets of differing topologies. Explore varying neural models and STDP. A new 3D environment and task will support the development of a more sophisticated agent that can learn a wider range of semantics.
ST4: Learn Plans (Months 13-24) MU: T1.4 will take 4 person months by Mitchell, the RA, and Huyck. Modify existing plans in response to environmental feedback. Create new plans and plan elements from user instruction. Expand plan capability by learning overlapping CAs for plan elements.

Table 4: Sub-Task Descriptions

Waypoint number	Waypoints name	Lead proposer short name & number	Delivery month	Comments
1.1	FLIF in PyNN	MU 1	2	
1.2	STP in PyNN	MU 1	6	
1.3	Model on Chips	MU 1	12	Both Chips
2.1	Simple Agent on PyNN	MU 1	3	
2.2	CABot3 on SpiNNaker	MU 1	9	HICANN date flexible
3.1	Learned CAs persist like STM	MU 1	15	Based on ACT-R model
3.2	Classification Cognitive Model	MU 1	18	Based on Shepard et al.
3.3	Learned CAs help the agent	MU 1	21	Game based evaluation
4.1	Weight plans and cache commands	MU 1	18	
4.2	Agent complete	MU 1	24	Possibly 1 version for each chip

Table 5: Waypoint Descriptions

5.3.3 Software/Hardware Functions (Components of Task 11.3.2)

The function boxes and timetable below depicts the envisaged start and end dates of the planned activities. These estimates are based on the current status, but are subject to change if new knowledge becomes available.

Task No:	T11.3.2	Partner:	MU
Function No:	11.3.2.1	Leader:	Chris Huyck
Function Name:	Transfer FLIF model (variant **1**) to PyNN		
Use Case A:	PyNN		
Planned Start Date:	01/04/2014	Planned Completion Date:	30/05/2014
Requires Functions:	None		

Task No:	T11.3.2	Partner:	MU
Function No:	11.3.2.2	Leader:	Chris Huyck
Function Name:	Transfer FLIF model (variant 2) to Manchester neuromorphic system (NM-MC-1)		
Use Case A:	SpiNNaker chip		
Planned Start Date:	01/03/2015	Planned Completion Date:	31/03/2015
Requires Functions:	None		

Task No:	T11.3.2	Partner:	MU
Function No:	11.3.2.3	Leader:	Chris Huyck
Function Name:	Translate CABot1 to PyNN)		
Use Case A:	PyNN		
Planned Start Date:	01/05/2015	Planned Completion Date:	29/05/2015
Requires Functions:	11.3.2.1		

Task No:	T11.3.2	Partner:	MU
Function No:	11.3.2.4	Leader:	Chris Huyck
Function Name:	Translate CABot1 to Manchester emulator and SpiNNaker chip		
Use Case A:	SpiNNaker chip		
Planned Start Date:	01/08/2014	Planned Completion Date:	31/10/2014
Requires Functions:	11.3.2.1 and 11.3.2.3		

Task No:	T11.3.2	Partner:	MU
Function No:	11.3.2.5	Leader:	Chris Huyck
Function Name:	Translate STP binding mechanism to PyNN and SpiNNaker		
Use Case A:	SpiNNaker chip		
Planned Start Date:	01/10/2014	Planned Completion Date:	31/10/2014
Requires Functions:	11.3.2.1		

Task No:	T11.3.2	Partner:	MU
Function No:	11.3.2.6	Leader:	Chris Huyck
Function Name:	Translate the 3D environment to PyNN and integrate it to SpiNNaker chip		
Use Case A:	SpiNNaker chip		
Planned Start Date:	01/11/2014	Planned Completion Date:	15/11/2014
Requires Functions:	SpiNNaker chip (we need to sort this problem out with Manchester). SP10 will eventually give us an environment but not for quite some time.		

Task No:	T11.3.2	Partner:	MU
Function No:	11.3.2.7	Leader:	Chris Huyck
Function Name:	Implement CABot3 on SpiNNaker		
Use Case A:	SpiNNaker		
Planned Start Date:	01/09/2014	Planned Completion Date:	31/12/2014
Requires Functions:	11.3.2.4, 11.3.2.5, and 11.3.2.6		

Task No:	T11.3.2	Partner:	MU
Function No:	11.3.2.8	Leader:	Chris Huyck
Function Name:	Translate CABot3 to Heidelberg system (NM-PM-1)		
Use Case A:	HICANN		
Planned Start Date:	5/01/2015	Planned Completion Date:	31/03/2015
Requires Functions:	11.3.1		

Task No:	T11.3.2	Partner:	MU
Function No:	11.3.2.9	Leader:	Chris Huyck
Function Name:	Learn Visual Objects in the Environment		
Use Case A:	SpiNNaker		
Planned Start Date:	05/01/2015	Planned Completion Date:	31/03/2015
Requires Functions:	11.3.2.6		

Task No:	T11.3.2	Partner:	MU
Function No:	11.3.2.10	Leader:	Chris Huyck
Function Name:	Explore the short- and long-term dynamics of CA persistence and creation		
Use Case A:	PyNN, SpiNNaker, and HICANN		
Planned Start Date:	01/02/2015	Planned Completion Date:	30/06/2015
Requires Functions:	None		

Task No:	T11.3.2	Partner:	MU
Function No:	11.3.2.11	Leader:	Chris Huyck
Function Name:	Use compensatory Hebbian learning with subnets of different topologies		
Use Case A:	PyNN, SpiNNaker, and HICANN		
Planned Start Date:	01/02/2015	Planned Completion Date:	30/06/2015
Requires Functions:	None		

Task No:	T11.3.2	Partner:	MU
Function No:	11.3.2.12	Leader:	Chris Huyck
Function Name:	Explore varying neural models and spike-timing dependent plasticity		
Use Case A:	PyNN, SpiNNaker, and HICANN		
Planned Start Date:	01/07/2015	Planned Completion Date:	30/10/2015
Requires Functions:	None		

Task No:	T11.3.2	Partner:	MU
Function No:	11.3.2.13	Leader:	Chris Huyck
Function Name:	Cognitive model for categorisation		
Use Case A:	PyNN, SpiNNaker, and HICANN		
Planned Start Date:	01/04/2015	Planned Completion Date:	30/09/2015
Requires Functions:	11.3.2.9		

Task No:	T11.3.2	Partner:	MU
Function No:	11.3.2.14	Leader:	Ian Mitchell
Function Name:	Cache plans		
Use Case A:	PyNN, SpiNNaker, and HICANN		
Planned Start Date:	01/04/2015	Planned Completion Date:	30/09/2015
Requires Functions:	11.3.2.6		

Task No:	T11.3.2	Partner:	MU
Function No:	11.3.2.15	Leader:	Chris Huyck
Function Name:	Cognitive model of Wisconsin Card Sorting Task		
Use Case A:	PyNN, SpiNNaker, and HICANN		
Planned Start Date:	01/11/2015	Planned Completion Date:	31/03/2016
Requires Functions:	11.3.2.6		

Task No:	T11.3.2	Partner:	MU
Function No:	11.3.2.16	Leader:	Ian Mitchell
Function Name:	Learn plans dynamically		
Use Case A:	PyNN, SpiNNaker, and HICANN		
Planned Start Date:	01/10/2015	Planned Completion Date:	31/03/2016
Requires Functions:	11.3.2.6, and 11.3.2.14		

5.3.4 Scientific Key Performance Indicators for Task11.3.2

One easily measurable indicator is the number of sub-networks completed. The original CABot3 was broken into 46 subnets, and this should be roughly the number needed for the translated version for, e.g., M2.2. The measure can be for translation to PyNN, to the SpiNNaker simulator or chip, and to the HICANN simulator or chip. It is not entirely clear how new subnets will be added for the learning tasks, but it is likely that several will be added.

An additional indicator is the comparison of our existing Java FLIF CABot3 system with the newly created agents on SpiNNaker, HICANN and simulated in PyNN. The comparison will be complex, but for instance, the neuromorphic agents should perform much faster. Another set of indicators comes with the cognitive models. The system will learn visual

categories in a cognitively viable way, and it should duplicate human results in the Wisconsin Card Sorting task.

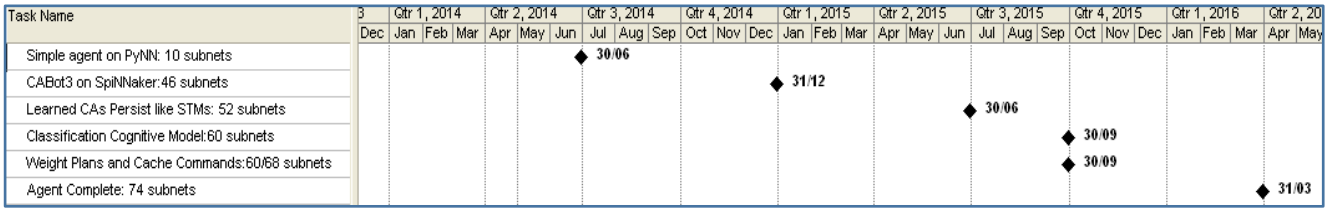

Table 6: Timeline of Functions in T11.3.2

5.4 T11.3.3 Exploiting of Feedback in Ultra-Fast Spiking Visual Architectures

In this Task, we will port architectures for spike-based visual object recognition to PyNN, and begin preliminary simulations on the HBP Platforms. The feedforward architectures will be similar to those simulations recently reported by IMSE [Perez13], where recognition was performed with delays in the range of 1-2ms.

We will perform exhaustive optimisations, both on system parameters (such as synaptic values, neurons thresholds, delays, etc.) and structural properties (such as number of layers, number of feature extractors per layer, number of neurons per feature extractor, etc.). We will add Gradual Attentional Feedback to the architectures to carefully assess performance variations in object recognition, while degrading the visual stimuli. We will implement additional hardware interfaces between Address Event Representation (AER) sensors and processing modules with the Neuromorphic systems. Finally, we will analyse mismatch impact on performance.

Summary of Objectives: port object recognition architecture to NM-MC platform, perform exhaustive parameter optimisations, explore impact of feedback, and assess effect mismatch of NM-PM through NM-MC simulations.

5.4.1 Detailed Task Description

Step A.1: Feedforward Architecture. It is widely established that visual processing in brains is structured in sequential layers. Early layers extract very simple features (like oriented edges in V1), while subsequent layers combine features from previous layers to detect gradually more complex shapes, forms, and figures. Full object recognition and scene analyses are performed in later layers [LeCun98, Riesenhuber99, Serre07]. However, such models are usually not implemented with spiking neurons, but with continuous neurons fed by input images from commercial cameras. The recent availability of spiking retina sensors [Lichsteiner08, Posch11, Lenero11, Serrano13a] and hardware spiking processing modules [Serrano06, 08, Camunas11, 12] has revealed interesting computational capabilities of spiking neural networks. In multi-layer spiking systems, computations are performed spike-by-spike (event-by-event) as soon as they become available. Thus, when a stream of spikes (representing a collection of visual features) is fed to an event-driven processing layer, events at the output become available shortly after the onset of the input stream, making both input and output streams almost simultaneous [Farabet12]. This extends to multi-layer systems, and we call it the pseudo-simultaneity property of multi-layer visual spiking processing [Farabet12, Perez13]. Recently, extremely fast (1-2ms from stimulus onset) and effective object recognition in multi-layer feedforward spiking vision systems has been demonstrated through simulations for very small scale systems (of about 5k neurons) with retina inputs of 32x32 pixels [Perez13]. In this Task, a first goal is to port such multi-layer vision systems to the Neuromorphic Platform (SP9) providing their description in the PyNN language, while scaling them up (initially) to several tens of thousands of neurons with retina inputs of up to 128x128 pixels.

Step A.2: Optimisations. The real-time and accelerated-time capabilities of the two neuromorphic systems is a unique feature that allows for exhaustive parameter search and optimisation of the architectures. Search/optimisation for small variations in the structure of the networks—such as the number of feature maps per layer, size of layers, size of feature maps, fan-out of neurons, etc.—is also viable. Such exhaustive search/optimisation is absolutely unviable using conventional simulation tools, if one wants to simulate reasonably long vision sequences while iterating them thousands of times for exhaustive parameter optimisations. In this sense, the two systems are interestingly complementary. The Manchester system (NM-MC), which works in real time, allows for much finer parameter adjustments and even different neural models, as it is based on digital computations with 32-bit integer precision. The Heidelberg system (NM-PM), although much faster, has a fixed physical model. However, both allow for large degrees of freedom in adjusting parameters. The goal is to define a set of figures of merit to characterise the performance of each visual processing iteration (e.g., improvement of success rate in object recognition), and run conventional optimisation routines that propose parameter variations for each iteration. Spiking neural networks have extra (timing) parameters [Perez13] normally unavailable in conventional vision processing neural networks, which allow for extra optimisation capabilities and possibly improved final performance.

Step A.3: Feedback. In a next step, the objective is to analyse the impact of gradually adding pseudo-simultaneous attentional feedback to the computing structures for enhancing object recognition capability when stimuli degrade. This is one of the most challenging goals of this Task. It relies on the fact that having the pseudo-simultaneity property allows to naturally inserting instantaneous feedback paths. This contrasts strongly with conventional image processing structures in computer vision, where all processing is feedforward (it is not viable to take an output image from a late stage and combine it with an input image of an early stage, and to iterate this feedback until convergence to find the solution). However, in multi-layer spiking visual systems, later layers begin to produce spikes while the input receives the first front of spikes. Thus, feedback connections translate immediately the effect of later layers' activity back to early layers. This can be used to explore immediate attentional feedback mechanisms to assess how the visual recognition capability of the network is enhanced, while providing gradually degraded visual stimuli.

Ultimately, we will be able to analyse stability effects, identify feedback paths for stability compensation, and monitor the attentional capabilities of the system. The availability of the neuromorphic systems is a unique opportunity for this study, since such simulations would otherwise be very slow given the complexity of the structures being analysed (multi-layer vision systems), and given the difficulty of searching for optimum parameters. Furthermore, very long simulation times may be required to identify instability in connectivity. However, in a real-time or accelerated-time system, instabilities become apparent immediately. Benchmarks would be the improvement factor of success rate in object recognition.

Step B: Interfacing. This Task will also interface the neuromorphic system(s) with existing AER (Address Event Representation) spiking visual sensory and pre-processing hardware. The proposing group has a number of AER retina sensors [Costas07, Lenero10, 12, Serrano13] as well as visual filtering modules implemented on dedicated chips [Serrano06, 08, Camunas11, 12] or Field Programmable Gate Arrays (FPGAs) [Zamarreno13], all of them communicating through the AER protocol. The NM-MC preliminary boards (4-chip and 48-chip SpiNNaker boards [Painkras13]) allow for AER interfacing either through lower speed parallel connectors and/or high-speed gigabits-per-second serial connections. Parallel interfaces are already available. Here, we plan to directly interface retinas and filtering modules with the NM-MC system, which operates in real time, and which consequently interfaces naturally with sensory devices. For the NM-PM system, since it operates in accelerated-time, interfacing must be done by providing recordings and define

them as stimuli in PyNN descriptions for the accelerated simulations. Thus, here we will provide a nice interactive demonstrator for live visual object recognition.

Note that bio-inspired visual processing is very neuron-hungry. For example, if the input retina has 128x128 (16384) pixels, and we have a V1 layer with 8 orientations and 5 scales, this V1 layer would already consume 655360 neurons. However, by relaying the initial layers to external hardware, we could limit the number of neurons within the HBP Platform to the "more intelligent" and less numerous neurons of later layers, while still studying larger scale visual systems.

At a more advanced stage, we will use depth clues for improved object segmentation to extend the visual architectures with stereovision capability for combined depth perception and object recognition. The group has some initial experience with event-driven stereovision exploiting event-driven filters [Serrano13b, Camunas14]. The idea is to test depth performance by using architectures that perform event-driven processing on two parallel channels (one per retina) while inserting cross-coupling filters, following the bio-inspired proposals of Qian [Qian94] and Shi [Wang10].

Step C. Variability. Parameter variability is natural in physical systems like the brain or the NM-PM system. The NM-MC system is based on digital ARM processors and a powerful digital interconnect AER mesh; consequently, it does not suffer from parameter variability. However, since parameters can be programmed with 32-bit precision, it is possible to artificially add highly controlled variability into all network parameters. This approach allows us to study the impact of variability, and to analyse how the systems degrade (if they degrade) when incrementally increasing variability. It also allows us to identify critical parameters, the variability of which degrades performance. On the other hand, the NM-PM system has the natural variability of its analogue components. When performing exhaustive parameter optimisations, we will be able to assess whether this variability is helpful by comparing the situations with and without mismatch. Also, by making comparisons with the gradual variability analyses from the NM-MC system, we can determine at which point the NM-PM system lies. We can also determine how its variability changes the system with respect to the ideal mismatch-less case.

5.4.2 Software/Hardware Functions (Components of Task 11.3.3)

Task No:	T11.3.3	Partner:	CSIC
Function No:	11.3.3.1	Leader:	Bernabe Linares-Barranco
Function Name:	Step A.1. Description of multi-layer vision systems in PyNN		
Use Case A:	PyNN for feedforward architecture		
Planned Start Date:	April 2014	Planned Completion Date:	October 2014
Requires Functions:			

Task No:	T11.3.3	Partner:	CSIC
Function No:	11.3.3.2	Leader:	Bernabe Linares-Barranco
Function Name:	Step A.2. Simulations with exhaustive parameter optimisations		
Use Case A:	Maximisation of recognition performance		
Planned Start Date:	November 2014	Planned Completion Date:	October 2015
Requires Functions:	11.3.3.1		

Task No:	T11.3.3	Partner:	CSIC
Function No:	11.3.3.3	Leader:	Bernabe Linares-Barranco
Function Name:	Step A.3. Exploring immediate attentional feedback mechanisms		
Use Case A:	PyNN for feed-back architecture		
Planned Start Date:	Nov 2014	Planned Completion Date:	March 2016
Requires Functions:	11.3.3.1		

Task No:	T11.3.3	Partner:	CSIC
Function No:	11.3.3.4	Leader:	Bernabe Linares-Barranco
Function Name:	Step B: Characterisation of interfaces		
Use Case A:	Allow interfacing between AER PCBs and SpiNNaker		
Planned Start Date:	April 2014	Planned Completion Date:	October 2014
Requires Functions:	None		

Task No:	T11.3.3	Partner:	CSIC
Function No:	11.3.3.5	Leader:	Bernabe Linares-Barranco
Function Name:	Step B. Integrate multi-layer vision systems to the Neuromorphic Platform		
Use Case A:	Verify object recognition with visual sensor		
Planned Start Date:	October 2014	Planned Completion Date:	Oct 2015
Requires Functions:	11.3.3.4		

Task No:	T11.3.3	Partner:	CSIC
Function No:	11.3.3.6	Leader:	Bernabe Linares-Barranco
Function Name:	Step B. PyNN for stereo architecture		
Use Case A:	Combined depth perception and object recognition		
Planned Start Date:	Nov 2015	Planned Completion Date:	March 2016
Requires Functions:	11.3.3.1, 11.3.3.2, 11.3.3.4, 11.3.3.5		

Task No:	T11.3.3	Partner:	CSIC
Function No:	11.3.3.7	Leader:	Bernabe Linares-Barranco
Function Name:	Step C. PyNN for mismatch		
Use Case A:	Prepare PyNN for mismatch study		
Planned Start Date:	Nov 2014	Planned Completion Date:	March 2015
Requires Functions:	11.3.3.1		

Task No:	T11.3.3	Partner:	CSIC
Function No:	11.3.3.8	Leader:	Bernabe Linares-Barranco
Function Name:	Step C. Exhaustive mismatch analysis on SpiNNaker		
Use Case A:	Explore impact of random mismatch on object recognition performance		
Planned Start Date:	April 2015	Planned Completion Date:	Dec 2015
Requires Functions:	11.3.3.1, 11.3.3.7		

Task No:	T11.3.3	Partner:	MU
Function No:	11.3.3.9	Leader:	TBD
Function Name:	Step C. Mismatch characterisation and assessment for NM-PM system		
Use Case A:	To explore the effect of NM-PM natural mismatch, but with NM-MC		
Planned Start Date:	Jan 2016	Planned Completion Date:	March 2016
Requires Functions:	11.3.3.2, 11.3.3.8		

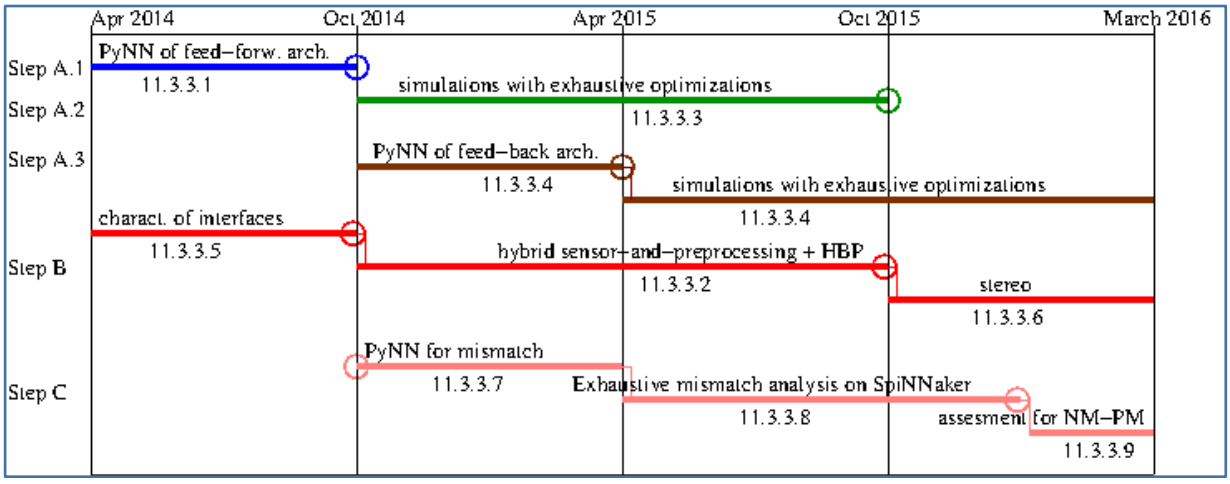

Figure 7: Timeline of Functions in T11.3.3

5.4.3 Scientific Key Performance Indicators for Task11.3.3

Step A.1: six-month duration. Each month: 20% completion (except first month).

Step A.2: 12-month duration. Each month: 8.3% completion.

Step A.3: 18-month duration. Each month: 5.5% completion.

Step B: 24-month duration. Each month: 4.17% completion.

Step C: 18-month duration. Each month: 5.5% completion.

5.4.4 T11.3.3 References

[Camunas11] L. Camuñas-Mesa, A. Acosta-Jiménez, C. Zamarreño-Ramos, T. Serrano-Gotarredona, and B. Linares-Barranco, "A 32x32 Pixel Convolution Processor Chip for Address Event Vision Sensors with 155ns Event Latency and 20Meps Throughput," *IEEE Trans. Circ. and Syst. Part-I*, vol. 58, No. 4, pp. 777-790, April 2011.

[Camunas12] L. Camuñas-Mesa, C. Zamarreño-Ramos, A. Linares-Barranco, A. Acosta-Jiménez, T. Serrano-Gotarredona, and B. Linares-Barranco, "An Event-Driven Multi-Kernel Convolution Processor Module for Event-Driven Vision Sensors," *IEEE J. of Solid-State Circuits*, vol. 47, No. 2, pp. 504-517, Feb. 2012.

[Camunas14] L. A. Camuñas-Mesa, T. Serrano-Gotarredona, S. Ieng, R. Benosman and B. Linares-Barranco, "On the use of Orientation Filters for 3D Reconstruction in Event-Driven Stereo Vision," *Frontiers in Neuromorphic Engineering*, *Front. Neurosci.*, under Review.

[Costas07] J. Costas-Santos, T. Serrano-Gotarredona, R. Serrano-Gotarredona and B. Linares-Barranco, "A Spatial Contrast Retina with On-chip Calibration for Neuromorphic Spike-Based AER Vision Systems," *IEEE Trans. Circuits and Systems, Part-I: Regular Papers*, vol. 54, No. 7, pp. 1444-1458, July 2007.

[Farabet12] C. Farabet, R. Paz, J. Pérez-Carrasco, C. Zamarreño-Ramos, A. Linares-Barranco, Y. LeCun, E. Culurciello, T. Serrano-Gotarredona, and B. Linares-Barranco, "Comparison Between Frame-Constrained Fix-Pixel-Value and Frame-Free Spiking-Dynamic-Pixel ConvNets for Visual Processing," *Frontiers in Neuromorphic Engineering*, Front. Neurosci. **6**:32. doi: 10.3389/fnins.2012.00032, 12 April 2012. http://www.frontiersin.org/Neuromorphic_Engineering/10.3389/fnins.2012.00032/abstract.

[LeCun98] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-Based Learning Applied to Document Recognition," Proc. IEEE, vol. 86, no. 11, pp. 2278-2324, Nov. 1998.

[Leñero10] J. A. Leñero-Bardallo, T. Serrano-Gotarredona, B. Linares-Barranco, "A 5-Decade Dynamic Range Ambient-Light-Independent Calibrated Signed-Spatial-Contrast AER Retina with 0.1ms Latency and Optional Time-to-First-Spike Mode," *IEEE Trans. Circ. and Syst., Part-I*, vol. 57, No. 10, pp. 2632-2643, October 2010.

[Leñero11] J. A. Leñero-Bardallo, T. Serrano-Gotarredona, and B. Linares-Barranco, "A 3.6 s latency asynchronous frame-free event-driven dynamic-vision-sensor," *IEEE J. Solid-State Circuits*, vol. 46, no. 6, pp. 1443–55, Jun. 2011.

[Lichtsteiner08] P. Lichtsteiner, C. Posch, and T. Delbrück, "A 128x128 120 dB 15us latency asynchronous temporal contrast vision sensor," *IEEE J. Solid-State Circuits*, vol. 43, no. 2, pp. 566–576, Feb. 2008.

[Painkras13] E. Painkras, L. A. Plana, J. Garside, S. Temple, F. Galluppi, C. Patterson, D. R. Lester, A. D. Brown, and S. B. Furber, "SpiNNaker: a 1-W 18-Core System-on-Chip for MassivelyParallel Neural Network Simulation," IEEE J. Solid-State Circ., vol. 48, No. 8, pp. 1943-1953, Aug. 2013.

[Perez13] J. A. Pérez-Carrasco, B. Zhao, C. Serrano, B. Acha, T. Serrano-Gotarredona, S. Chen and B. Linares-Barranco, "Mapping from Frame-Driven to Frame-Free Event-Driven Vision Systems by Low-Rate Rate-Coding and Coincidence Processing. Application to Feed-Forward ConvNets," IEEE Trans. on Pattern Analysis and Machine Intelligence, vol. 35, No. 11, pp. 2706-2719, Nov. 2013.

[Posch11] C. Posch, D.Matolin, and R.Wohlgenannt, "A QVGA 143 dB dynamic range frame-free PWM image sensor with lossless pixel-level video compression and time-domain CDS," *IEEE J. Solid-State Circuits*, vol. 46, no. 1, pp. 259–275, Jan. 2011.

[Qian94] N. Qian, "Computing Stereo Disparity and Motion with Known Binocular Cell Properties," *Neural Computation*, 6, pp. 390-404, 1994.

[Riesenhuber99] M. Riesenhuber and T. Poggio, "Hierarchical Models of Object Recognition in Cortex," *Nature Neuroscience*, vol. 2, no. 11, pp. 1019-1025, 1999.

[Serrano06] R. Serrano-Gotarredona, T. Serrano-Gotarredona, A. Acosta-Jimenez, and B. Linares-Barranco, "A Neuromorphic Cortical-Layer Microchip for Spike-Based Event

Processing Vision Systems," *IEEE Trans. Circ. and Syst.-I,* vol. 53, No. 12, pp. 2548-2566, Dec. 2006.
[Serrano08] R. Serrano-Gotarredona, T. Serrano-Gotarredona, A. Acosta-Jimenez, C. Serrano-Gotarredona, J. A. Perez-Carrasco, A. Linares-Barranco, G. Jimenez-Moreno, A. Civit-Ballcels, and B. Linares-Barranco, "On Real-Time AER 2D Convolutions Hardware for Neuromorphic Spike Based Cortical Processing," *IEEE Trans. on Neural Networks,* vol. 19, No. 7, pp. 1196-1219, July 2008.
[Serrano09] R. Serrano-Gotarredona, M. Oster, P. Lichtsteiner, A. Linares-Barranco, R. Paz-Vicente, F. Gómez-Rodríguez, L. Camuñas-Mesa, R. Berner, M. Rivas, T. Delbrück, S. C. Liu, R. Douglas, P. Häfliger, G. Jiménez-Moreno, A. Civit, T. Serrano-Gotarredona, A. Acosta-Jiménez, B. Linares-Barranco, "CAVIAR: A 45k-Neuron, 5M-Synapse, 12G-connects/sec AER Hardware Sensory-Processing-Learning-Actuating System for High Speed Visual Object Recognition and Tracking," *IEEE Trans. on Neural Networks,* vol. 20, No. 9, pp. 1417-1438, Sep. 2009.
[Serrano13a] T. Serrano-Gotarredona and B. Linares-Barranco, "A 128x128 1.5% Contrast Sensitivity 0.9% FPN 3us Latency 4mW Asynchronous Frame-Free Dynamic Vision Sensor Using Transimpedance Amplifiers," IEEE J. Solid-State Circuits, vol.48, No. 3, pp. 827-838, March 2013.
[Serrano13b] T. Serrano-Gotarredona, J. Park, A. Linares-Barranco, A. Jiménez, R. Benosman, and B. Linares-Barranco, "Improved contrast sensitivity DVS and its application to event-driven stereo vision," *IEEE Int. Symp. Circ. and Syst.*, pp. 2420-2423, 2013.
[Serre07] T. Serre, L. Wolf, S. Bileschi, M. Riesenhuber, and T. Poggio, "Robust Object Recognition with Cortex-Like Mechanisms," IEEE. Trans. Pattern Analysis and Mach. Int., vol. 29, No. 3, pp. 411-426, March 2007.
[Wang10] Y. Wang and B. E. Shi, "Autonomous Development of Vergence Control Driven by Disparity Energy Neuron Populations," Neural Computation, 22, pp. 730-751, 2010.
[Zamarreno13] C. Zamarreño-Ramos, A. Linares-Barranco, T. Serrano-Gotarredona, and B. Linares-Barranco, "Multi-Casting Mesh AER: A Scalable Assembly Approach for Reconfigurable Neuromorphic Structured AER Systems. Application to ConvNets," IEEE Trans. on Biomed. Circ. and Syst., vol. 7, No. 1, pp. 82-102, Feb. 2013.

5.5 T11.3.4 Spiking Associative Networks for Neuromorphic Computing Systems

The goal of this Task is to explore the computational capabilities of spiking neural networks for associative memory (SAM: spiking associative memory) with the help of the two complementary HBP hardware systems in Heidelberg and Manchester. Neural associative memories can be seen as models for local networks in the cerebral cortex [1] and are closely related to assembly theory [2]. Furthermore, they offer efficient methods for fault-tolerant information retrieval in technical cognitive systems, with possibly better performance than classical algorithms of computer science like content addressable memories, search-trees, or hash-tables [3].

5.5.1 Objectives and Approach

We start with the classical binary associative net with binary neurons and synapses proposed by Steinbuch [4], and theoretically analysed by Willshaw et al. [5] and Palm [6], hereafter referred to as the BiNAM (binary neural associative memory). There are a number of alternative models for associative memory; the most prominent is the Hopfield model [7]. However, the BiNAM is the simplest architecture to start with. It has a number of advantages over other models both for brain modelling and technical application, and is theoretically well understood [8]. To achieve a high storage capacity, the BiNAM requires sparse representation of patterns (sparse coding). This gives excellent support for the address event representation (AER) of spikes used in both HBP hardware systems, and relaxes the requirements for analogue computation in neurons implemented in the Heidelberg neuromorphic system. Furthermore, our group has implemented resource-efficient VLSI hardware for BiNAMs in the past [9, 10].

In its basic form, the BiNAM can already be interpreted as a spiking model for associative memory since the neuron inputs and outputs are binary, and could be interpreted as single spikes. However, the input-output mapping (association) is done in one step and assumes perfect synchronisation for all “spikes” in the input pattern and for all neurons receiving this input pattern. The temporal aspect of spike trains is almost neglected. Hence, we want to extend the basic BiNAM with the temporal aspect of spikes (SAM), which will be investigated on a local level (the main focus of this Task) and a global level (for future research). On the local level, the role of spikes for separation of superimposed output patterns will be analysed. Superposition of output patterns is a well-known problem of distributed information storage in neural associative memories and can be solved partly by spikes as demonstrated in [11], [12], [13]. On the global level, we will study synchronisation aspects (binding and competition of distributed representations) in networks of SAMs (spiking associative networks, or SANs). Of course, both levels are highly interdependent. However, local synchrony is the prerequisite for global interaction of locally spiking populations.

As a thorough mathematical analysis for spiking networks is still out of reach, we have to apply simulations for verification of the SAM models. We will use the Heidelberg neuromorphic system for fast design space exploration of selected parameters of basic associative memory architectures. The aim is to approximate by simulations the set of pareto-optimal system configurations (pareto-set). The performance measures will be storage capacity, retrieval time, energy consumption, and robustness to internal as well as external noise. Based on a PyNN description, we will use the Heidelberg neuromorphic system because of its “faster than real-time simulation capabilities”. This system allows configuration of at least 8 out of the 24 electrical parameters, which translate to biological parameters from the PyNN description [14]. Hence, even if the Heidelberg neuromorphic system does not offer the full flavour of configurable SAM model parameters, the design space for parameter optimisation is huge.

An important issue in spiking neural networks is pattern binding or synchronisation, which has a high impact on SAM performance. Synchronisation is important for combining several SAMs to form a network of SAMs (SAN: spiking associative network) or to embed SAMs into cognitive architectures. For the spiking model, the spikes may arrive asynchronously, and the following questions arise: what spike delay for a given input pattern is acceptable and manageable by the system? How precisely can the outputs of neurons be synchronised in the context of AER and serial communication in both HBP hardware systems? Are special synchroniser units beneficial as suggested in [15]? We want to figure out minimal requirements for synchronisation in SAMs.

As the Heidelberg system has restrictions with respect to the accuracy of synapses (4 Bit) and neuron internal analogue computation, we will use the Manchester system to analyse the effect of more sensitive synapses (32 Bit). We will also use the Manchester system to analyse the effect of higher computational accuracy (fixed-point arithmetic) on synchronisation and the considered performance measures. At the end of this Task, we want to be able to connect basic SAMs to form large spiking associative networks (SANs), as the cortex might not be one huge associative memory but a network of interconnected (auto- and hetero-associative) SAMs (in addition to other functions of specific cortical areas).

The spiking BiNAM should be seen only as a very abstract model of a local cortical network [16]. Because of its reduced complexity, it is a good model to start with, but it only accounts for a small number of neurophysiological facts (e.g., binary pattern vector represents the binary nature of spikes, binary synaptic matrix reflects the assumption that synapses are either excitatory or inhibitory, sparse activation patterns). Nevertheless, it is still a challenge to find resource-efficient implementations in the sense of pareto-optima. Once this simple model has been successfully implemented, we can extend this model step-by-step—e.g. with non-binary synapses, inhibitory neurons, and more complex feedback structures—and we can study the effects on SAM behaviour quantified by the performance measures. This will give us some hints about the purpose or the advantages of spikes and corresponding system parameters for SAMs based on quantitative performance measures.

Due to their intuitive architecture and interpretable input/output behaviour, SAMs could also be used for hardware evaluation and tests. We will do a first evaluation of the applicability of this alternative automatic test procedure based on specially defined SAM test patterns for the neuromorphic hardware used in the HBP. Besides hardware defects, architectural bottlenecks could be detected and reported for the next HBP hardware revisions.

5.5.2 Methodology and Associated Work Plan

The Task will have two Work Packages (modelling SAMs with PyNN and HBP hardware; SAM design space exploration). First, we will focus on the implementation of SAMs in the simulator-independent network description language PyNN, and their mapping onto the HBP hardware. The first SAM version will be based on the leaky integrated-and-fire neuron as implemented in the Heidelberg hardware [14] and the spiking associative memory based on BiNAM as implemented by Knoblauch et al [13,17]. Step-by-step, we will extend this model, e.g. with non-binary synapses, inhibitory neurons, and more complex feedback structures. We will study the effects on SAM behaviour quantified by the performance measures (storage capacity, retrieval time, energy consumption, and robustness). For functional verification, we will use a software simulator (e.g. Neuron, Nest, PCSIM). We will coordinate the selection of the simulator with Heidelberg and Manchester to maintain compatibility between the HBP's hardware suppliers.

The next step will be the mapping of the SAM to the virtual hardware simulator of the Heidelberg group, which is a detailed simulation of the final wafer-scale hardware system implemented in C++/SystemC [14]. The virtual hardware offers an early modeller's

perspective on the capabilities of the future wafer-scale system. The last step will be the mapping to the hardware systems (Heidelberg, Manchester) as soon as they are available for this Task. For a smooth transition from simulation to real hardware, we intend to use the HICANN prototype in Heidelberg. The Key Performance Indicator (KPI) for this work is the number of implemented SAMs in PyNN. We will implement four SAM models in this Task (one model every six months).

For the performance evaluation and system optimisation, we need benchmark data sets. Because SAMs can store a huge number of sparsely coded patterns (about $n^2/\log^2 n$, n is the number of neurons), these pattern sets have to be automatically generated. For testing purposes, the characteristics of these pattern sets have to be configurable (e.g., the degree of correlation between spikes, sparseness, distribution of the activated inputs, noise level, distortion grade of patterns, spike delay, asynchronism, etc.). Basically, we need three benchmark pattern sets: a set for testing the normal association functionality without errors or noise; a set with noisy and erroneous patterns for testing the robustness of the associative functionality; and a set for testing the correct functionality of the computing hardware. The last pattern set is especially interesting for the two HBP hardware systems, and must be individually designed for them. This Task focuses on the first two benchmark data sets. The researcher financed by this Task will be responsible for this Work Package (19 PM financed by the requested Task budget; 5 PM financed using own resources). The KPI for this work is the number of generated benchmark pattern sets. We will implement two pattern sets: the first after 12 months and the second after 18 months.

Once we have a first functional version of the SAM, we will prepare the design space exploration for SAM architectures (second Work Package). The parameter spaces have to be systematically swept to approximate the set of pareto-optimal SAM configurations. We have to distinguish the biologically motivated parameter set from the theoretical model (or the PyNN description), and the electrical parameter set given by the hardware system (especially from the neuromorphic Heidelberg system). Both should be considered and optimised. At this point, it is not clear how to organise the optimisation. One option is to optimise the biological parameters first, translate them into the electrical parameter space, and continue the optimisation. The other option is to do it the other way around. As there is no bijective mapping between both parameter sets (to our knowledge), both approaches may lead to implausible results.

As soon as the HBP hardware is available, the Heidelberg wafer-scale system can be used for a fast search in the parameter space for narrowing the space down to an interesting region, which can then be investigated using the SpiNNaker system from Manchester [18] or a software simulator with higher precision. The strategy for approximating the pareto-set will be based on a combination of two optimisation techniques that we have already used in the context of circuit optimisation [19]. It is a combination of evolutionary optimisation and a set-oriented numerical method. A senior researcher from our research group will execute this Work Package in close cooperation with the researcher financed by the Task budget. In the long term, we want to come up with a tool chain for automatic multi-objective optimisation of SAN architectures. The KPI for this work is the number of simulated SAMs on the HBP hardware systems, which should increase linearly each month starting from Month 24.

Within this Task, close cooperation with the research groups from Heidelberg (WP9.1) and Manchester (WP9.2) is of course mandatory. We know both groups and their hardware systems quite well. In addition, we will cooperate closely with the research group of Prof. Anders Lansner (Stockholm University and Royal Institute of Technology, Sweden) as this group is working on bio-inspired associative memory models in the HBP (WP9.3). Last but not least, our research on SAMs will contribute to principles of brain computation and the integration of the SAM models into HBP hardware systems (WP9.6).

5.5.3 Progress to Date

Task 11.3.4 started in April 2014 with the implementation of a first version of a spiking associative memory (SAM) in PyNN based on leaky integrate-and-fire (IaF) neurons. The SAM has a single layer structure and the synapses of the IaF neurons are binary. For an automatic generation of SAM benchmark sets (input/output spike patterns) we specified a generic program package as well as a tool for analysing the simulation outputs (spike trains) for comprehensive reporting of the performance evaluation.

5.5.4 Software/Hardware Functions (Components of Task 11.3.4)

Task No:	T11.3.4	Partner:	UNIBI
Function No:	11.3.4.1	Leader:	Ulrich Rückert
Function Name:	Implementation of SAM architectures in PyNN		
Use Case A:	Modelling of different SAM architectures in Software		
Planned Start Date:	April 2014	Planned Completion Date:	June 2015
Requires Functions:	None		

Task No:	T11.3.4	Partner:	UNIBI
Function No:	11.3.4.2	Leader:	Ulrich Rückert
Function Name:	Automatic performance evaluation		
Use Case A:	Tools for analysing the simulation outputs (spike trains) and for comprehensive reporting of the performance		
Planned Start Date:	October 2014	Planned Completion Date:	March 2015
Requires Functions:	11.3.4.1		

Task No:	T11.3.4	Partner:	UNIBI
Function No:	11.3.4.3	Leader:	Ulrich Rückert
Function Name:	Automatic generation of benchmark data sets		
Use Case A:	Tool for automatic generation of SAM benchmark sets with specified characteristics		
Planned Start Date:	January 2015	Planned Completion Date:	June 2015
Requires Functions:	11.3.4.1		

Task No:	T11.3.4	Partner:	UNIBI
Function No:	11.3.4.4	Leader:	Ulrich Rückert
Function Name:	Mapping SAM architectures on the virtual hardware simulator		
Use Case A:	Implementation of SAM architectures on the virtual hardware simulator of the Heidelberg group		
Planned Start Date:	January 2015	Planned Completion Date:	March 2015
Requires Functions:	11.3.3.1		

Task No:	T11.3.4	Partner:	UNIBI
Function No:	11.3.4.5	Leader:	Ulrich Rückert
Function Name:	Mapping SAM architectures on HBP hardware platform NM-PM		
Use Case A:	Transition from hardware emulation to SAM implementation on the Heidelberg hardware platform NM-PM		
Planned Start Date:	July 2015	Planned Completion Date:	March 2016
Requires Functions:	11.3.4.4		

Task No:	T11.3.4	Partner:	UNIBI
Function No:	11.3.4.6	Leader:	Ulrich Rückert
Function Name:	Mapping SAM architectures on HBP hardware platform NM-MC		
Use Case A:	Transition from the simulation to SAM implementation on the Manchester hardware platform NM-MC		
Planned Start Date:	October 2015	Planned Completion Date:	March 2016
Requires Functions:	11.3.4.1		

5.5.5 Waypoints and Scientific Key Performance Indicators for Task11.3.4

This Task has two Milestones: MS313 - the SAM emulation on the virtual hardware (delivery Month 19), and MS314 - SAM implementation on the HBP hardware (delivery Month 30). The KPIs for this Task are the number of implemented SAM models (up to four), the number of generated benchmark sets (up to two), and the number of simulations on the HBP hardware systems (more than six).

Milestones	Delivery Month
MS313: Emulation of SAM on virtual Hardware	19
MS314: Implementation of SAM on HBP Hardware	30
Key Performance Indicators	**Delivery Month**
Number of implemented SAMs in PyNN: 1 - 4:	12, 18, 24, 30
Number of generated benchmark pattern sets 1 - 2:	18, 24
Number of simulated SAMs on HBP hardware: 1 - 6	24, 25, 26, 27, 28 29, 30

Table 7: Timeline of T11.3.4

5.5.6 T11.3.4 References

[1] Braitenberg, V.; Schüz, A.: Anatomy of the cortex. Statistics and geometry. Springer-Verlag, Berlin, 1991.
[2] Braitenberg, V.: Cell assemblies in the cerebral cortex. In R. Heim and G. Palm, editors, Lecture notes in biomathematics (21). Theoretical approaches to complex systems, pp. 171–188. Springer-Verlag, Berlin Heidelberg New York, 1978.
[3] Schmidt, M., Rückert, U.: Content-Based Information Retrieval using an embedded Neural Associative Memory. 9th Euromicro Workshop on Parallel and Distributed Processing Mantova, Italien, pp. 443-450, 2001.
[4] Steinbuch, K.: Die Lernmatrix. Kybernetik, 1:36–45, 1961.
[5] Willshaw, D.J., Buneman, O.P., Longuet-Higgins, H.C.: Non-holographic associative memory. Nature, 222, pp. 960–962, 1969.
[6] Palm, G.: On associative memories. Biological Cybernetics, 36:19–31, 1980.
[7] Hopfield, J.J.: Neural networks and physical systems with emergent collective computational abilities. Proceedings of the National Academy of Science, USA, 79:2554-2558, 1982.
[8] Palm, G.: Neural associative memories and sparse coding. Neural Networks, 37, pp. 165 – 171, 2013.
[9] Rückert, U.: ULSI Architectures for Artificial Neural Networks. In: IEEE Micro 22, No. 3, pp. 10-19, 2002.
[10] Schaefer, M.; Schoenauer, T.; Wolff, C.; Hartmann, G.; Klar, H.; Rückert, U.: Simulation of spiking neural networks – architectures and implementations, Neurocomputing 48, pp. 647-679, 2002.
[11] Wennekers, T., Palm, G.: On the relation between neural modelling and experimental neuroscience. Theory in Bioscience, 116, pp. 273–289, 1997.
[12] Sommer, F.T., Wennekers T.: Associative memory in networks of spiking neurons. Neural Networks, 14(6/7), pp. 825–834, 2001.
[13] Knoblauch, A., Palm, G.: Pattern separation and synchronisation in spiking associative memories and visual areas. Neural Networks, 14, pp. 763–780, 2001.
[14] Brüderle, D., et al.: Establishing a novel modeling tool: a python-based interface for a neuromorphic hardware system. Frontiers in Neuroinformatics, Vol. 3, Article 17, pp. 1-10, 2009.
[15] Witkowski, U., Rückert, U.: Pattern Synchronization for Associative Memory in pulse coded Neural Networks. The 47th IEEE Int. Midwest Symposium on Circuits and Systems, pp. II.381 – II.384, 2004.
[16] Lansner, A.: Associative memory models: from the cell-assembly theory to biophysically detailed cortex simulations, Trends in Neurosciences, Vol. 32, No 3, pp. 178-186, 2008.
[17] Knoblauch, A., Palm, G., Sommer, F. T.: Memory capacities for synaptic and structural plasticity. Neural Computation, 22, 289–341, 2010.
[18] Furber, S., et al. Overview of the SpiNNaker system architecture. IEEE Trans. On Computers. http://doi.ieeecomputersociety.org/10.1109/TC.2012.142, 2012.
[19] Blesken, M.; Lütkemeier, S.; Rückert, U. Multiobjective Optimization for Transistor Sizing of Sub-threshold CMOS Logic Standard Cells. 2010 IEEE International Symposium on Circuits and System Proceedings, 2010.

5.6 T11.3.5 Asynchronous Computational Retina

5.6.1 Objectives

This Task will develop a pure, event-driven visual computation approach that will use precise timing mechanisms to design new computation techniques in visual processing. The Task will produce a full event-driven visual processing system linking a neuromorphic retina directly to the SpiNNaker system by an Asynchronous Event Representation (AER) bus. The architecture will allow the first real-time development and implementation of new, visual, event-driven computation techniques. We will implement event-driven early vision models and 3D stereovision in the SpiNNaker board using a precise timing mechanism. They will be fed directly by the output of a neuromorphic retina, ATIS (Asynchronous Time-based Image Sensor), developed by the UPMC's research team. The consistency and the robustness of the implemented models and algorithms will be constantly analysed via extensive evaluations conducted throughout the Task.

5.6.2 Components

The ATIS used in this work is an event-based time-domain, developed by members of the team, which encodes image sensors with 304x240 pixel resolution. The sensor contains an array of fully autonomous pixels that combines a luminance change detector circuit and a conditional exposure measurement block. These sensors are novel vision devices that, like their biological counterparts, are driven by "events" occurring within the scene—as opposed to conventional image sensors that are driven by artificial timing and control signals (e.g., a frame clock) with no relation to the source of the visual information. The ATIS output fits the SpiNNaker massively parallel multi-core architecture. It will benefit from SpiNNaker's ability to simulate millions of neurons, and make use of its computing power to develop and map event-driven computation architecture.

Existing work that uses neuromorphic silicon retina visual information based on the event-driven paradigm is still rare. The most-used approaches map multidimensional raw data, obtained from images or videos taken using conventional cameras, into a temporal space before feeding the data to spiking networks. Bohte et al. (2002) proposed a population coding method that encodes an input variable using multiple overlapping Radial Basis Functions (RBFs). The bank of RBFs is designed to cover the whole data range. Coarsely, each RBF is associated with a neuron that fires with a delayed time proportional to the response of the real value to the RBF.

Wu et al. (2007) and Kerr et al. (2011) consider a rate-based spiking encoding method that converts the responses of convolutional filters (like Difference of Gaussians, or DoG) applied on images into spike trains, for which the spiking frequency is proportional to the responses. Masquelier and Thorpe (2007), Weidenbacher and Neumann (2008) and Wysoski et al. (2008) applied similar spike timing encoding schemes. A grey-scale intensity image is filtered with DoG or/and Gabor filters. The obtained feature maps are converted into spike ranks by sorting all activities over all feature maps in descending order. This time-to-first-spike coding framework corresponds to the rank coding that was introduced in Thorpe and Gatrais (1998) and efficiently encodes image information in VanRullen and Thorpe (2001).

Escobar et al. (2009) proposed a similar two-step approach using an energy-based V1 model (Adelson and Bergen, 1985). First, analogue processing is done on blocks of frames (from a video) through spatiotemporal energy filters in order to emulate V1 simple cells. Second, the complex cells are modelled as conductance-driven integrate-and-fire neurons, where the external input current is associated with a V1 complex cell response.

In their early stages, these proposed systems present strong assumptions about the time encoding of the visual information in the visual cortex functional structure they emulate. These approaches demonstrate the efficiency of integrating such theoretical temporal

encoding into experimental frameworks, e.g., the rank coding in VanRullen and Thorpe (2001) and Masquelier and Thorpe (2007). In addition, the introduced spike latencies (or frequencies) were often derived from empirical or statistical settings, which are rarely correlated with biological observations. These implementations rely on visual scene acquisition based on conventional synchronous cameras. The precision of the spike firing is limited at least by the frame rate of the acquisition system, which is inconsistent with biological observations.

As reviewed in Panzerri et al. (2010), the precision of neural codes is indeed influenced by the stimulus dynamics. Butts et al. (2007) have demonstrated that the temporal precision of the neuronal responses in the visual thalamus is significantly greater when movies are displayed at much higher frame rates than normal. In addition, their data demonstrate that millisecond precision of spike times is required to decode spatial image details. Although Panzerri et al. (2010) also suggest that the brain might encode the visual information in a multiplexed manner at different timescales, it appears to be important that the visual information is encoded with the greatest temporal precision, as close as possible to the native visual information provided by the human retina.

5.6.3 Plan

This Task will provide the data, techniques, models and methods needed to produce realistic vision-based information (outputs of retina ganglion cells, optical flow, etc.) in real time. The research community could use this information to propose, test, and validate more realistic computational models of the brain's visual function and operation. A hardware system exploiting the advantages of event-driven computation in object classification was recently proposed in Perez-Carasco et al. (2013). A spike-based convolutional neural network performing real-time classification from high-speed input was implemented on a custom AER-based system. Recognition occurred as soon as the convolution layers received enough events, outperforming standard frame-based techniques. The use of such custom solutions demonstrates the capabilities of event-driven computation. We propose a system that increases the flexibility of such paradigms while maintaining precise timing information.

In this Task, we will feed the event-driven retina sensor's outputs directly into SpiNNaker to make use of the temporal dynamics of these sensors. This will allow us to build a neuromorphic sensory architecture capable of processing and interpreting visual information the moment it is acquired, thus maintaining the relative time between events. To reach this goal, we are using the ATIS silicon retina (Posch et al., 2011). The ATIS is sensitive to scene contrasts, and encodes scene changes and luminance as spikes with timing precision of less than 1 microsecond, using the Address Event Representation (AER) that is compatible with the SpiNNaker. This fully event-driven vision system will implement two fundamental, asynchronous vision-processing tasks to outline the importance of precise timing in visual information. It will also provide valuable information for further developments in robotics and cognitive task modelling, thus facilitating interactions with other research areas within the HBP.

The approaches will use direct output from the retina, but also spikes from a neuromorphic retina model developed by the Vision Institute that we plan to implement on SpiNNaker (Lorach et al., 2012). Currently, the model is running in real time on a conventional computer. It is based on an asynchronous convolutional methodology that matches with the SpiNNaker architecture. This will provide the HBP with a realistic real-time retina model with temporal resolutions similar to biological retinas.

The proposed approach will be organised as follows:

Interfacing the ATIS with SpiNNaker: This primary task is mandatory to allow SpiNNaker to process the retina's inputs as directly as possible from the silicon retina. Both devices are already using the same event-driven data format (AER). However, their designs do not

include their native connection. The interfacing will also be extended to allow the integration of two ATIS in order to get the basis of an event-driven stereovision system.

The event-driven computation paradigm for artificial and asynchronous retina models: This Task's objective is to implement the retina model developed in Lorach et al. (2012), in which ganglion cells' behaviour is reproduced on the SpiNNaker board. The higher temporal and spatial accuracy of the ATIS outlines the impact of precise timing; e.g., the event-driven optical flow developed in Benosman et al. (2012) and the event-driven 3D reconstruction algorithm that triangulates points from the stereovision system (Rogister et al., 2012; Carneiro et al., 2013). The implementation of these algorithms in SpiNNaker achieves the asynchronous processing of the flows of events at the rate of their arrival, using an event-driven methodology.

5.6.4 Software/Hardware Functions (Components of Task 11.3.5)

Task No:	T11.3.5	Partner:	UPMC
Function No:	11.3.5.1	Leader:	Ryad Benosman
Function Name:	Hardware implementation: Interface to connect one ATIS camera into SpiNNaker		
Use Case A:	Feed data into SpiNNaker for computational models		
Planned Start Date:	April 2014	Planned Completion Date:	May 2015
Requires Functions:	None		

Task No:	T11.3.5	Partner:	UPMC
Function No:	11.3.5.2	Leader:	Ryad Benosman
Function Name:	Hardware implementation: Interface to connect two ATIS cameras into SpiNNaker		
Use Case A:	Feed data into SpiNNaker for computational models		
Planned Start Date:	April 2014	Planned Completion Date:	May 2015
Requires Functions:	11.3.5.1		

Task No:	T11.3.5	Partner:	UPMC
Function No:	11.3.5.3	Leader:	Ryad Benosman
Function Name:	Hardware implementation: Stimulation platform		
Use Case A:	Generate input stimulus for acquisition with ATIS cameras		
Planned Start Date:	April 2014	Planned Completion Date:	May 2015
Requires Functions:	None		

Task No:	T11.3.5	Partner:	UPMC
Function No:	11.3.5.4	Leader:	Ryad Benosman
Function Name:	Hardware implementation: Database platform		
Use Case A:	Generate datasets for tests and evaluation of computational models		
Planned Start Date:	April 2014	Planned Completion Date:	May 2015
Requires Functions:	11.3.5.1, 11.3.5.3		

Task No:	T11.3.5	Partner:	UPMC
Function No:	11.3.5.5	Leader:	Ryad Benosman
Function Name:	Computational model: Visual motion		
Use Case A:	Real-time implementation on SpiNNaker of the Visual motion model		
Planned Start Date:	January 2015	Planned Completion Date:	May 2016
Requires Functions:	11.3.5.1, 11.3.5.3, 11.3.5.4		

Task No:	T11.3.5	Partner:	UPMC
Function No:	11.3.5.6	Leader:	Ryad Benosman
Function Name:	Computational model: Retina model		
Use Case A:	Real-time implementation on SpiNNaker of the Retina model		
Planned Start Date:	January 2015	Planned Completion Date:	May 2016
Requires Functions:	11.3.5.1, 11.3.5.3, 11.3.5.4		

Task No:	T11.3.5	Partner:	UPMC
Function No:	11.3.5.7	Leader:	Ryad Benosman
Function Name:	Computational model: Stereovision		
Use Case A:	Real-time implementation on SpiNNaker of the Stereovision model		
Planned Start Date:	January 2015	Planned Completion Date:	May 2016
Requires Functions:	11.3.5.2		

5.6.5 Scientific Key Performance Indicators for Task 11.3.5

Progress within functions 11.3.5.1, 11.3.5.2, 11.3.5.3 and 11.3.5.4 will be measured by assigning it a “status”, as per the table below.

Function	Function Name	Possible KPI statuses	Current KPI status
11.3.5.1	Hardware implementation: Interface to connect one ATIS camera into SpiNNaker	1. Electronic Design & Components Selection 2. Fabrication & Assembly 3. Software Implementation (driver, SW interface) 4. Test & Evaluations 5. Final validation (redaction of technical reports)	4
11.3.5.2	Hardware implementation: Interface to connect two ATIS cameras into SpiNNaker	1. Electronic Design & Components Selection 2. Fabrication & Assembly 3. Software Implementation (driver, SW interface) 4. Test & Evaluations 5. Final validation (redaction of technical reports)	2
11.3.5.3	Hardware implementation: Stimulation platform	1. Electronic Design & Components Selection 2. Fabrication & Assembly 3. Software Implementation (driver, SW interface) 4. Test & Evaluations 5. Final validation (redaction of technical reports)	1
11.3.5.4	Hardware implementation: Database platform	1. Electronic Design & Components Selection 2. Fabrication & Assembly 3. Software Implementation (driver, SW interface) 4. Test & Evaluations 5. Final validation (redaction of technical reports)	2
Totals			**9**

Table 8: Key Performance Indicators for T11.3.5, Step 1

Function 11.3.5.4 will provide a database. The obtained sequences will be evaluated based on the number of events per second (mean, maximum, etc.). The sequences will be clustered into a set of data classes. The capability of the developed and implemented methods to process them will be established using benchmarks defined by selecting at least five reference sequences per class. The benchmarks will be used to evaluate the performance (in terms of number of events processed per second) of the three developed models (visual motion, retina model, stereovision). The expected value is 1k events/sec for comparison with the temporal precision observed in the biological retina. This performance will be considered the scientific KPI used in functions 11.3.5.5, 11.3.5.6 and 11.3.5.7.

Function	Function Name	Scientific KPI	KPI model type	Target Values
11.3.5.5	Computational Model: Visual motion	Number of events processed per second	More is better	M32: 1,000 events/sec
11.3.5.6	Computational Model: Retina Model	Number of events processed per second	More is better	M32: 1,000 events/sec
11.3.5.7	Computation Model: Stereovision	Number of events processed per second	More is better	M32: 1,000 events/sec

Table 9: Key Performance Indicators for T11.3.5, Step 2

11.3.5 References

(Adelson and Bergen, 1985) Adelson, E. and Bergen, J. (1985). Spatiotemporal energy models for the perception of motion. Journal of the Optical Society of America A, 2, 284–299.

(Benosman et al., 2012) Ryad Benosman, Sio-Hoi Ieng, Charles Clercq, Ciara Bartolozzi, Mandyam Srinivasan, (2012) Asynchronous frameless event-based optical flow, Neural Networks. Volume 27, March, Pages 32–37

(Benosman et al., 2011) R. Benosman, S. H. Ieng, P. Rogister, and C. Posch, 'Asynchronous Event-Based Hebbian Epipolar Geometry,' IEEE Transaction on Neural Networks, vol. 22, pp. 1723–1734, 2011.

(Bohte et al., 2002) Bohte, S.M., La Poutre, H., Kok, J.N.: Unsupervised clustering with spiking neurons by sparse temporal coding and Multi-Layer RBF Networks. IEEE Transactions on Neural Networks 13(2), 426–435 (2002)

(Butts et al., 2007) Butts, D.A. et al. (2007) Temporal precision in the neural code and the timescales of natural vision. Nature 449, 92–95

(Carneiro et al., 2013) João Carneiro, Sio-Hoi Ieng, Christoph Posch, Ryad Benosman, "Asynchronous Event-Based 3D Reconstruction From Neuromorphic Retinas". Neural Networks, 15th march 2013. DOI:10.1016/j.neunet.2013.03.006

(Escobar et al., 2009) Maria-Jose Escobar, Guillaume S. Masson, Thierry Vieville and Pierre Kornprobst, Action Recognition Using a Bio-Inspired Feedforward Spiking Network, Int. Journal ComputVis, 82: 284-301, 2009

(Kayser et al., 2003) Kayser, C. et al. (2003) Temporal correlations of orientations in natural scenes. Neurocomputing 52, 117–123

(Kerr et al., 2011) Dermot Kerr, Martin McGinnity, Sonya Coleman, Qingxiang Wu and Marine Clogenson, Spiking hierarchical neural network for corner detection, IJCCI (NCTA) 2011: 230-235, 2011.

(Lorach et al., 2012) Henri Lorach, Serge Picaud, Sio-Hoi Ieng, Jose Sahel, Ryad Benosman. Artificial retina: the multichannel processing and temporal dynamic of the biological system achieved with an asynchronous model, Journal of Neural Engineering (2012).

(Rogister et al., 2012) Rogister, P., Benosman, R., Ieng, S. H., Lichtsteiner, P., & Delbruck, T. (2012). Asynchronous event-based binocular stereo matching. Neural Networks and Learning Systems, IEEE Transactions on, 23(2), 347-353.

(Masquelier and Thorpe. 2007) Masquelier, T., & Thorpe, S. J. (2007). Unsupervised learning of visual features through spike timing dependent plasticity. PLoS Computational Biology, 3(2), e31.

(Meftah et al, 2012) Boudjelal Meftah, Olivier Lezoray, Soni Chaturvedi, Aleefia A. Khurshid, and Abdelkader Benyettou, Image processing with Spiking Neuron Networks, X.-S. Yang (Ed.): Artif. Intell., Evol. Comput. and Metaheuristics, SCI 427, pp. 525–544. Springer-Verlag Berlin Heidelberg 2012.

(Panzeri et al., 2010) Panzeri, S., Brunel, N., Logothetis, N. K., & Kayser, C. (2010). Sensory neural codes using multiplexed temporal scales. Trends in neurosciences, 33(3), 111-120.

(Perez-Carasco et al., 2013) J. A. Pérez-Carrasco, B. Zhao, C. Serrano, B. Acha, T. Serrano-Gotarredona, S. Chen and B. Linares-Barranco, "Mapping from Frame-Driven to Frame-Free Event-Driven Vision Systems by Low-Rate Rate-Coding and Coincidence Processing. Application to Feed-Forward ConvNets," IEEE Trans. on Pattern Analysis and Machine Intelligence, vol. 35, No. 11, pp. 2706-2719, Nov. 2013.

(Posch, 2011) C. Posch, D. Matolin, and R. Wohlgenannt, "A qvga 143 db dynamic range frame-free pwm image sensor with lossless pixel-level video compression and time-domain cds," Solid-State Circuits, IEEE Journal of, vol. 46, no. 1, pp. 259 –275, jan. 2011

(Thorpe and Gatrais, 1998) Thorpe, S., and Gautrais, J. (1998). “Rank order coding,” in Computational Neuroscience: Trends in Research, ed. J. M. Bower (New York: Plenum Press), 113–118.

(VanRullen and Thorpe, 2001) VanRullen R, Thorpe SJ (2001) Rate coding versus temporal order coding: What the retinal ganglion cells tell the visual cortex. Neural Comput 13: 1255–1283.

(Weidenbacher and Neumann, 2008) Ulrich Weidenbacher and Heiko Neumann, The first spike counts: A model for STDP learning pose specific representations for estimating view direction, Journal of Vision 8 (6), 161-161

(Wu et al. 2008) Qingxiang Wu, T. Martin McGinnity, Liam P. Maguire, Ammar Belatreche, Brendan P. Glackin: Processing visual stimuli using hierarchical spiking neural networks. Neurocomputing 71(10-12): 2055-2068 (2008)

(Wysoski et al, 2008) Simei Gomes Wysoski, Lubica Benuskova, Nikola Kasabov. Fast and adaptive network of spiking neurons for multi-view visual pattern recognition, Neurocomputing 71 (2008) 2563–2575

5.7 T11.3.6 Implementing a Spiking Classifier Network on HiCANN

Our goal is to implement spiking neuromorphic networks for multivariate data classification that exploit the properties of large-scale neuromorphic systems, and that solve real-world computing problems. Our networks harness the massively parallel architecture of brain circuits to efficiently process high-dimensional data. Our network design takes inspiration from the olfactory system, mimicking the general blueprint of parallel feature encoding, processing and classification in feedforward networks with lateral inhibition. To achieve our overall goal, we must first achieve the following objectives:

- To develop a scalable implementation targeting the large-scale system provided by the Heidelberg group.
- To implement this solution on the Heidelberg system and test it on large multivariate classification problems.

5.7.1 Previous Work

We previously designed a framework for bio-inspired classification of multivariate data, which is based on a three-stage architecture (Figure 8). In the first stage, multivariate data are encoded into a firing-rate representation by virtual receptors (VRs, "Input" in Figure 8) with large, overlapping receptive fields. In the second stage, the firing-rate correlations between VR output channels are reduced by lateral inhibition ("Decorrelation" in Figure 8). The third stage consists of a linear classifier that is trained in a supervised fashion to perform classification ("Association" in Figure 8). Recently, we completed a spiking implementation of this network scheme in PyNN that runs on the *Spikey* hardware.

5.7.1.1 Virtual Receptors

The first challenge to developing a generic neuromorphic framework for multivariate data classification with spiking neural networks is to achieve a universally valid transformation from the domain of real numbers into the domain of spike trains. This transformation must provide bounded, positive numbers (spike rates) from the unbounded space of real numbers. We use the approach of virtual receptors (VRs) defined as linear radial basis functions with large, overlapping receptive fields. The response r of a VR located at point p (in data space coordinates) to stimulus **s** is given by: $r = 1 - (d(\mathbf{s},\mathbf{p}) - d_{\min})/(d_{\max} - d_{\min})$, where d(**s**,**p**) is the Euclidean distance between **s** and **p** in data space, and $d_{\min}$, $d_{\max}$ are the minimal and maximal distance observed across all data points **s** in the data set.

The VR positions (i.e., the RBF centroids) are placed in the data space using an unsupervised, self-organising approach like a neural gas [3] to ensure good coverage of the data manifold in high-dimensional space. Large overlapping receptive fields ensure full coverage of the data space without supervised tuning, though they induce correlations between VR outputs. The number n_{VR} of VRs determines whether the dimensionality n_{dim} of the original data is inflated ($n_{VR} > n_{dim}$) or reduced ($n_{VR} < n_{dim}$). In the spiking version of the model, VR output is transformed using a gamma process to obtain spike trains, which constitute the input to the network (receptor neurons, or RNs, in Figure 8).

5.7.1.2 Lateral Inhibition

In the original, rate-coded model, the vector **r** of all VR responses is transformed by lateral inhibition according to $r' = r - q \cdot C \cdot r^T/n_{VR}$, where **r'** denotes the output of projection neurons (PNs) in the firing rate model, n_{VR} the number of VRs, **C** a zero-diagonal weight matrix, and q a weight factor to adjust the overall strength of lateral inhibition. Lateral inhibition is implemented by lateral inhibitory connections (LN connections in Figure 8). Lateral inhibition effectively reduces the correlation between VRs' outputs that is due to their overlapping receptive fields [1,4].

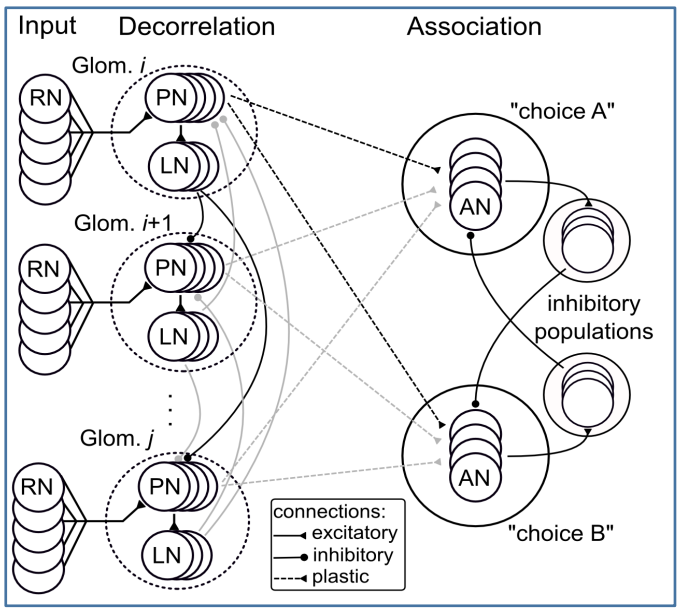

Figure 8: Network Schematic

Key for Figure 8*: RN: Receptor neurons, PN: projection neurons, AN: Association neurons.*

5.7.1.3 A Spiking Circuit for Supervised Classification

It has been shown that Bayesian inference can be implemented with a simple winner-take-all (WTA) circuit and a supervised Hebbian learning rule [5]. Our classifier stage implements these concepts in a spiking network (termed "association layer" in Figure 8). PNs in the network converge onto association neurons (ANs) in the WTA circuit. Each class label that is present in the data set is assigned a corresponding population of ANs. The classifier system is trained iteratively: in each step, a data point is presented to the network and the resulting output of the WTA circuit is evaluated off-chip. The winner population (i.e., the AN population with the highest spike count) determines the classification output of the network. The weight updates are computed using a Hebbian learning rule: only PN-AN synapses with pre- and postsynaptic activity above a fixed threshold are changed. The change is positive (potentiation) if classification was correct, or negative (depression) if incorrect. The weights are then updated on the chip and the next data point from the training set is presented.

5.7.1.4 Proof-of-Concept

We have completed and tested an implementation of the spiking classifier network on the *Spikey* neuromorphic hardware system. Figure 9 demonstrates the network functionality for Fisher's classic Iris data set (Figure 9A). The depicted spiking activity (Figure 9B) was recorded at the end of the training phase. The network achieved a classification performance comparable to that of a Naïve Bayes (NB) classifier on the VR-transformed input data (not shown). A more challenging problem was posed by classification of a subset of the 784-dimensional MNIST data set that contains 28x28 pixel grey-scale images of handwritten digits (Figure 9C) [6]. We trained the network to differentiate the digits '5' and '7'. The spiking network clearly outperformed the NB classifier (Figure 9D, cols. 'NB' vs 'hw'). Importantly, when training the Naïve Bayes classifier on the PN firing rates (i.e., on the data representation after filtering by lateral inhibition), the spiking network performs at the same level as the full spiking implementation (Figure 9D, col. NB/PN). This result indicates that lateral inhibition improves the data representation such that even a very simple linear classifier can learn the classification boundaries.

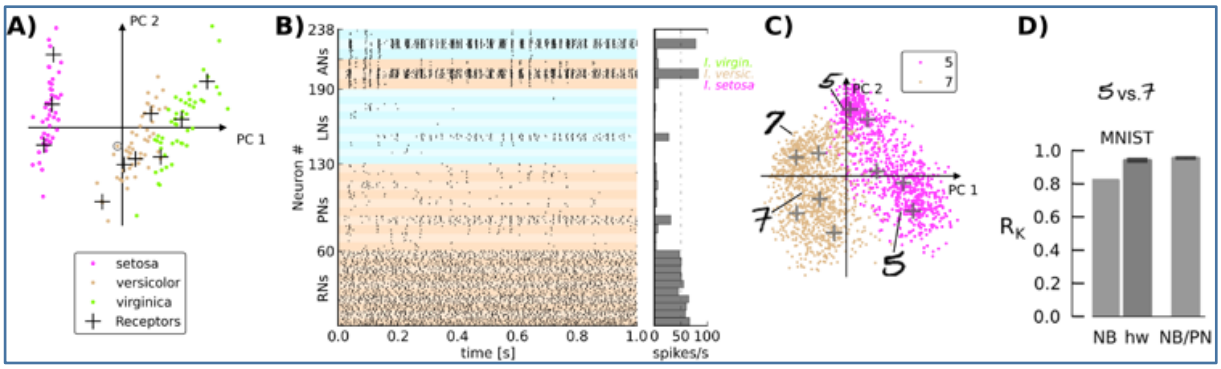

Figure 9: Results from the Proof-of-Concept Implementation.

Key to Figure 9*: A) The Iris data set with locations of VRs (2D projection of 4D-space). B) Activity (spike dot plot) in the trained network running on the Spikey hardware system in response to the data point outlined in grey in A). Orange: excitatory, blue: inhibitory populations. C) Subset of the MNIST data set and VRs (2D proj. of 768D space). D) Classifier performance on the MNIST data set (discriminating '5' against '7'). NB: Naïve Bayes. Hw: spiking network on the hardware, NB/PN: Naïve Bayes trained on the PN firing rates. RK: Gorodkin's k-category correlation coefficient.*

5.7.2 Particular Benefits of the Proposed Network for the Neuromorphic Approach

The pre-processing performed by VRs and lateral inhibition provides a representation of data space that can be easily partitioned by a linear classifier. This approach is particularly effective because VRs are placed in the data space such that they match the distribution of the data. But, while extremely beneficial, this placement alone is not sufficient for efficient linear classification – the lateral inhibition step is indispensable for good performance (see Figure 9D). The VR encoding essentially implements a soft Voronoi tesselation of the input space. While VR-receptive fields are global by design, the lateral inhibition step makes VRs "local experts" that represent the distribution of data points in their vicinity. Arithmetically, hard Voronoi tessellation is prohibitively expensive in high-dimensional feature spaces, and is practically infeasible even for data sets with moderate dimension counts. However, our lateral inhibition approach delivers this kind of tessellation basically for free. Moreover, it scales at no additional computation time cost when using larger neuronal systems to fit more VRs. Thus, lateral inhibition perfectly exploits the inherent massively parallel architecture of neuromorphic hardware systems.

Due to the limited neuron count on the *Spikey* system, we were limited to using 10 VRs to represent the data sets. Clearly, using more VRs provides more fine-grained representations of data space. Hence, increasing the number of VRs increases classification performance, as we have demonstrated in a firing rate model in [1]. Our aim in this proposal is therefore to scale the network to use more VRs, using the high neuron counts available on the large-scale neuromorphic systems developed by the HBP.

5.7.3 Proposed Approach

We will first implement a GPU-accelerated software version of the network to explore the effect of anticipated hardware-specific constraints in the large-scale system. These constraints include the switch from standard leaky IF on *Spikey* vs. AdExp IF on the large-scale system [7], potential limitations of connectivity, and bandwidth limitations for host-to-system communication. As soon as the next-generation single-chip system (*HiCANN*) and the large-scale systems are available, we will tackle implementation and benchmarking on the hardware system.

5.7.4 Description of Work

5.7.4.1 Model 1: Scaling Up Network to HiCANN-Supported Size

In this Task, we will modify our previous network to use a higher neuron count. This will build on preliminary work from a separately financed short project in eFuturesXD (EPSRC), in which we are testing different scales of the model in a GPU implementation. The immediate benefit of an increased neuron count will be the ability to use more VRs. More VRs will improve classification performance for large problems. Hence, in a first step, we will increase the number of VRs to cover the whole MNIST data set, instead of being restricted to two digits. We will explore the performance that can be achieved by this moderate upscaling of network size. A second step in the transition to using the HiCANN network will be the switch from standard IF to the AdExp IF neuron model. To identify a regime that supports stable network operation, we will use the flexibility of the GPU implementation to explore the parameter space of the AdExp model.

5.7.4.2 Model 2: Implement the Network on HiCANN

We will port the PyNN-based implementation of the network to use the HiCANN chip as it becomes available for PyNN users. Since we will already have explored the effects of scaling neuron counts to 512, and changing the neuron model to AdExp-IF in the GPU approach (Model 1), the challenge here will lie in dealing with the stochastic variability of hardware neurons. Here, we can build on our previous work, from which we know that asynchronous activity and population sizes play a crucial role in controlling variability.

5.7.4.3 Model 3: Implement the Network on the Wafer Scale

The next challenge will consist of expanding the implementation to use several tens of thousands of neurons on the wafer-scale system. The high number of available neurons will allow us to dramatically scale up the number of VRs, and thus enable us to process complex data sets. The wafer-scale system puts certain constraints on the connectivity between individual modules. Potential bottlenecks exist in connections with a large fan-in or fan-out, since these entail transmission of many spike events from or to a small group of neurons (e.g., connections from LNs to PNs, or from PNs to ANs). We will explore whether potential overloads can be remediated by reducing the respective connection probabilities without hurting network functionality. Another potential limitation is constrained external bandwidth, which may limit the number of spikes that can be monitored, and thus limit the operation of off-chip learning rules. These limitations can potentially be worked around by representative sampling of subsets of ANs and PNs.

We will challenge the network with diverse multivariate data sets with high feature counts. We expect these data sets to benefit maximally from the ability of the VR/lateral inhibition method to process data on high-dimensional manifolds. Examples include the “Human Activity Recognition Using Smartphones” (10,299 instances, 561 features); “p53 Mutants” (16,772 instances, 5,409 features); “Daily and Sports Activities”(9,120 instances, 5,625 features); and the “Dorothea” drug discovery data set (1,950 instances, 100,000 features); all of which are available from the UCI repository.

We will relate the execution time to conventional state-of-the-art machine learning approaches. By analysing the amount of time taken in the different stages of model execution (data preparation on the host machine, configuration of the hardware system, execution of the model, off-chip computation of synaptic weight updates), we will identify the major bottlenecks in overall computing time and suggest optimisations to the software architecture. Moreover, we expect to gain further insights into the nature of the kind of data sets for which the neuromorphic approach will provide the most benefits.

5.7.5 Software/Hardware Functions (Components of Task 11.3.6)

Task No:	11.3.6	Partner:	UoS
Model No:	11.3.6.1	Leader:	Thomas Nowotny
Model Name:	Mid-Scale Model (HiCANN size)		
Use Case A:	The mid-scale model runs on the IRIS and MNIST data sets (GPU based)		
Planned Start Date:	May 2014 (M8)	Planned Completion Date:	Nov 2014 (M14)
Requires:	None		

Task No:	11.3.6	Partner:	UoS
Model No:	11.3.6.2	Leader:	Thomas Nowotny
Model Name:	HiCANN mid-scale model		
Use Case A:	Mid-scale model classifies IRIS and MNIST data sets running on the HiCANN chip		
Planned Start Date:	Dec 2014 (M15)	Planned Completion Date:	Apr 2015 (M19)
Requires:	11.3.6.1, HiCANN System available for PyNN users		

Task No:	11.3.6	Partner:	UoS
Model No:	11.3.6.3	Leader:	Thomas Nowotny
Model Name:	Wafer-scale model		
Use Case A:	Large-scale model classifies IRIS and MNIST data sets running on the-wafer-scale system		
Planned Start Date:	May 2015 (M20)	Planned Completion Date:	Dec 2015 (M27)
Requires:	11.3.6.2, Wafer Scale System available for PyNN users.		

5.7.6 Scientific Key Performance Indicators for Task 11.3.6

KPI 1: Number of checked in code revisions on software management system (more is better)

KPI 2: Maximal performance achieved on the IRIS data set (more is better)

KPI 3: Number of different classifier models tested on IRIS (more is better)

5.7.7 T11.3.6 References

[1] Schmuker M, Schneider G (2007) Processing and classification of chemical data inspired by insect olfaction. *PNAS* 104:20285-20289.
[2] Schmuker M, Pfeil T, Nawrot MP (2014) A neuromorphic network for generic multivariate data classification. PNAS 111:2081-2086.
[3] Martinetz T, Schulten K (1991) A“ neural-gas” network learns topologies. In: Kohonen T, Mäkisara K, Simula O, Kangas J (eds) Artif. Neural Networks. Elsevier B.V., North-Holland, pp 397–402
[4] Pfeil T, Grübl A, Jeltsch S, et al. (2013) Six networks on a universal neuromorphic computing substrate. *Front Neurosci*. 7:11.
[5] Soltani A, Wang X-J (2010) Synaptic computation underlying probabilistic inference. *Nat Neurosci* 13:112–9.
[6] Lecun Y, Cortes C The MNIST database of handwritten digits. http://yann.lecun.com/exdb/mnist/.
[7] Brüderle D, Petrovici M A, Vogginger B, et al. (2011) A comprehensive workflow for general-purpose neural modeling with highly configurable neuromorphic hardware systems. *Biol Cybern* 104:263–96.

6. Glossary

Term	Definition	For more Information
AER	Address Event Representation. An asynchronous handshaking protocol used to transmit signals between neuromorphic systems	http://avlsi.ini.uzh.ch/classwiki/lib/exe/fetch.php?id=fall07%3Afall07&cache=cache&media=fall07:lab11.pdf
ARM Holdings	Manufacturer of processors and embedded CPU processors, based in Cambridge, UK.	http://www.arm.com
ATIS	Asynchronous time-based Image Sensor	
BiNAM	Binary Neural Associative Memory	
CA	Cell assembly	
CABot	Cell Assembly robot	
FLIF	Fatiguing Leaky Integrate and Fire. A point neural model.	http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.217.3900
FPGAs	Field Programmable Gate Arrays	http://en.wikipedia.org/wiki/Field-programmable_gate_array
IMSE	Instituto de Microelectrónica de Sevilla (Partner responsible for Task 11.3.3)	http://www.imse-cnm.csic.es
KPI	Key Performance Indicator	
LTP	Long-term potentiation. A long-lasting enhancement in signal transmission between two neurons that results from stimulating them synchronously.	http://en.wikipedia.org/wiki/Long-term_potentiation
PCSIM	Parallel neural Circuit SIMulator	http://www.lsm.tugraz.at/pcsim/
PM	Person Month	
PyNN	Simulator-independent Python programming language for building neuronal network models.	http://neuralensemble.org/PyNN/
RBF	Radial Basis Function	
SAM	Spiking associative memory	
SAN	Spiking associative network	
SpiNNaker	Spiking Neural Network Architecture. A UK-funded research project whose goal is to build neuromorphic computing systems based on many-core chips with efficient bi-directional links for asynchronous.	http://apt.cs.manchester.ac.uk/projects/SpiNNaker/
STP	Short term potentiation	http://www.ncbi.nlm.nih.gov/pubmed/9242283
STDP	Spike-timing Dependent Plasticity. A process that adjusts the strength of connections between neurons based on relative timings of inputs and output signals.	http://en.wikipedia.org/wiki/Spike-timing-dependent_plasticity
V1	First cortex layer of the primate visual cortex	http://en.wikipedia.org/wiki/Visual_cortex#Primary_visual_cortex_.28V1.29
VLSI	Very Large Scale Integration. The integration of very large numbers of transistors on a single silicon chip. VLSI devices were initially defined as chips with more than 10,000 transistors. Current systems may contain more than 2,000,000.	http://en.wikipedia.org/wiki/Very-large-scale_integration