



UNIVERSITÀ DEGLI STUDI
DI GENOVA



ISTITUTO ITALIANO
DI TECNOLOGIA

DITEN - DEPARTMENT OF ELECTRICAL, ELECTRONICS AND
TELECOMMUNICATION ENGINEERING AND NAVAL ARCHITECTURE

EDPR - EVENT DRIVEN PERCEPTION FOR ROBOTICS

PH.D. IN SCIENCE AND TECHNOLOGY FOR ELECTRONIC AND
TELECOMMUNICATION ENGINEERING – CYCLE XXXIV (2018-2021)

Neuromorphic Computing Systems for Tactile Sensing Perception

PhD Thesis

PhD Candidate:

Ali Dabbous

Tutor:

Prof. Maurizio Valle

Dr. Chiara Bartolozzi

Coordinator of the PhD Course: Prof. Maurizio Valle

Declaration

I hereby declare that except where specific reference is made to the work of others, the contents of this dissertation are original and have not been submitted in whole for consideration for any other degree or qualification in this, or any other universities. This dissertation is my own work and contains nothing which is the outcome of work done in collaboration with others, except as specified in the text and Acknowledgements. This dissertation contains fewer than 65,000 words including bibliography, footnotes, tables and equations and has fewer than 150 figures.

Ali Dabbous

November 2021

Acknowledgements

This Ph.D. journey is a combined effort of several individuals, including the family members who supported me and friends who were always accessible through this journey.

First and foremost, I would like to express my gratitude to my marvelous thesis supervisor Prof. Maurizio Valle whose support, guidance during these three years has made this thesis possible. He gave me opportunities to improve my work through my learning path.

I am very grateful to my supervisor Dr. Chiara Bartolozzi for her encouragement and extensive support through my Ph.D. journey. Her evaluation and suggestions were essential to solving various bottlenecks in my research work innovatively. She influenced me to work hard and persist in my determination.

A special thanks to Prof. Elisabetta Chicca for the advice and support she has given during my Ph.D. journey.

A big thanks to Dr. Ali Ibrahim for his guidance and invaluable advices during the three years of my P.h.D. I have learnt a lot from him and I am especially grateful for his help and mentorship.

I would like to thank my friends and lab mates at the COSMIC lab for the lively discussion and unforgettable adventures. I am also grateful to all my friends at the EDPR lab in IIT.

Last but not least, I would also like to thank my parents, my sisters, and my fiancée for their advice and support throughout my life. This could not be achieved without them.

Ali Dabbous

Abstract

Touch sensing plays an important role in humans daily life. Tasks like exploring, grasping and manipulating objects deeply rely on it. As such, Robots and hand prosthesis endowed with the sense of touch can better and more easily manipulate objects, and physically collaborate with other agents. Towards this goal, information about touched objects and surfaces has to be inferred from raw data coming from the sensors. The orientation of edges, which is employed as a pre-processing stage in both artificial vision and touch, is a key indication for object discrimination. Inspired on the encoding of edges in human first-order tactile afferents, we developed a biologically inspired, spiking models architecture that mimics human tactile perception with computational primitives that are implementable on low-power subthreshold neuromorphic hardware. The network architecture uses three layers of Leaky Integrate and Fire neurons to distinguish different edge orientations of a bar pressed on the artificial skin of the iCub robot. We demonstrated that the network architecture can learn the appropriate connectivity through unsupervised spike-based learning, and that the number and spatial distribution of sensitive areas within receptive fields are important in edge orientation discrimination. The unconstrained and random structure of the connectivity

among layers can produce unbalanced activity in the output neurons, which are driven by a variable amount of synaptic inputs. We explored two different mechanisms of synaptic normalization (weights normalization and homeostasis), defining how this can be useful during the learning phase and inference phase. The network is successfully able to discriminate between 35 orientations of 36 (0° to 180° with 5° step increments) with homeostasis and weights normalization mechanism. Besides edge orientation discrimination, we modified the network architecture to be able to classify six different touch modalities (e.g. poke, press, grab, squeeze, push, and rolling a wheel). We demonstrated the ability of the network to learn appropriate connectivity patterns for the classification, achieving a total accuracy of 88.3 %. Furthermore, another application scenario on the tactile object shapes recognition has been considered because of its importance in robotic manipulation. We illustrated that the network architecture with 2 layers of spiking neurons was able to discriminate the tactile object shapes with accuracy 100 %, after integrating to it an array of 160 piezoresistive tactile sensors where the object shapes are applied.

Keywords: Neuromorphic, Tactile sensing, Feature Extraction, Edge Orientation, Spiking Neural Network, Touch Modality, Touch, Winner-Take-All, Receptive Fields, Skin, Synaptic Learning, Weights Normalization, Homeostasis, Spike Driven Synaptic Plasticity, Spike Timing Dependent Plasticity .

Contents

	iii
Acknowledgements	v
Abstract	vii
List of Figures	xvii
List of Tables	xxix
1 Introduction	1
1.1 Motivation	1
1.2 Objectives	3
1.3 Contributions	5
1.4 List of Publications	8
1.5 Organization of the Thesis Document	9
2 Literature Review	11
2.1 Introduction	11

2.2	Human Sense of Touch	12
2.2.1	Introduction	12
2.2.2	Mechanoreceptors	13
2.2.3	Tactile Pathway and Information Processing	17
2.3	Artificial Sense of Touch	18
2.3.1	Introduction	18
2.3.2	Transduction Mechanisms	20
2.3.3	Tactile Sensors Implementation in Robotics	24
2.4	Feature Extraction and Edge Orientation Selectivity	26
2.4.1	Feature Extraction and Tactile Perception Overview	27
2.4.1.1	Traditional Machine Learning	27
2.4.1.2	Spiking Neural Networks and Neuromorphic Approaches	29
2.4.2	Edge Orientation Selectivity Overview	31
2.5	Discussion	33
3	Neuromorphic Approach for Tactile Perception	37
3.1	Introduction	37
3.2	Biologically Inspired Neuromorphic Approach for Tactile Perception	39
3.2.1	Touch Afferents: First-Layer Neurons	40
3.2.2	Receptive Fields: Second-Layer Neurons	42
3.2.3	Decoder Layer: Third-Layer Neurons	43
3.2.4	Winner-Take-All	45
3.3	Neuromorphic Architecture for Edge Orientation Selectivity	45

3.3.1	Sensors and Dataset	45
3.3.2	Network Architecture	48
3.4	Neuromorphic Architecture for Touch Modalities Classification	50
3.4.1	Sensors and Dataset	50
3.4.2	Network Architecture	52
3.5	Neuromorphic Architecture for Tactile Object Shape Classification	54
3.5.1	Sensors and Dataset	54
3.5.2	Network Architecture	54
3.6	Discussion	56
4	Receptive Fields	57
4.1	Introduction	57
4.2	Edge Orientation Discrimination with Designed Receptive Fields	60
4.3	Receptive Field Structure	63
4.3.1	Uniform Receptive Fields	65
4.3.2	Random Receptive Fields	66
4.3.3	Random with Subfields Receptive Fields	67
4.4	Receptive Fields Assessments Using Mutual Information Theory	67
4.4.1	Mutual Information Theory	68
4.4.2	Connected Sensors Study	69
4.4.3	Edge Length Study	71
4.4.4	Center position Study	73
4.5	Discussion	79

5 Spike Based Learning with Synaptic Normalization	81
5.1 Introduction	81
5.2 Spike Timing Dependent Plasticity (STDP)	84
5.3 Spike Driven Synaptic Plasticity (SDSP)	86
5.4 Unsupervised Learning and Synaptic Normalization Mechanisms for Edge Orientation Selectivity	89
5.4.1 Baseline Network	91
5.4.2 Weights Normalization Network	95
5.4.3 Homeostasis Network	100
5.5 Touch Modality Classification Using Supervised STDP Learning	104
5.6 Object Contact Shape Classification using Supervised-STDP	107
5.6.1 Software Implementation	107
5.6.2 Hardware Implementation	112
5.7 Discussion	113
6 Conclusion and Future Work	117
6.1 Conclusion	117
6.2 Future Work	121
Bibliography	125
Appendix A SDSP Learning Rule	141
A.1 SDSP Learning Rule Code	141

Appendix B Object Contact Shape Classification Codes	145
B.1 Brian2 Codes for Software Implementation	145
B.1.1 Dataset Splitting	145
B.1.2 STDP Rule and Training	146
B.1.3 Testing	152
B.2 Hardware Implementation Codes	154
B.2.1 Raspberry Pi Code	154
Appendix C Touch Modality Classification Codes	159
C.1 Brian2 Codes for Software Implementation	159
C.1.1 Training	159
C.1.2 Testing	163

List of Figures

2.1	Schematic representation of mechanoreceptors in human skin.	15
3.1	SA-I afferents: layer one encodes analog pressure in spike trains and consists of 160 LIF neurons; (A) An example of a vertical bar pressed on the skin patch that consists of 160 capacitive sensitive elements (Taxels) distributed over 16 triangles. (B) Analog output of the 160 taxels when pressed with the gray vertical (0°) oriented bar, where the gray shadow represents the non pressed taxels. (C) The output pressure of the taxels is used as input to LIF neurons; (top) Output spikes of neurons after pressing the skin with (0°) bar; (bottom) firing rate of one neuron. . . .	43
3.2	System setup for edge orientation processing	46
3.3	Skin patch from iCub robot.	47

- 3.4 Spiking neural network for edge orientation selectivity; Network architecture comprising skin patch and three layers of LIF neurons. Layer one neurons convert the analog capacity value into spike trains (SA-I afferents); Layer two gathers input from multiple layer one neurons and having spatially distributed, overlapping, receptive fields; Layer three neurons receive input from layer two, each neuron of layer three is selective to a specific input orientation. Recurrent connectivity by neurons of a global inhibitory neuron ensures that a single layer three neuron is active, implementing a form of WTA competition. 48
- 3.5 Randomly generated connectivity matrix between layer one and layer two (first 20 neurons on layer one); each neuron in layer one (rows) is connected to only one neuron in layer two (cols). 50
- 3.6 Six different touch modalities. 51
- 3.7 Spiking Neural Network for touch modalities classification; (Left) example of two from six different touch modalities applied on a sensor array of 160 piezo-resistive sensors. (Middle) 3 layers of LIF neurons; input layer consists of 160 neurons, hidden layer gathering the 160 inputs from input layer, and output layer consists of 6 neurons which is equal to the number of classes to be classified (touch modalities). (Right) lateral inhibitory layer that consists of 6 inhibitory neurons to increase the accuracy performance of the network. 53

- 3.8 Spiking Neural Network for object shapes classification. (Left) sensory array of 160 piezo-resistive sensors. (Middle) 2 layers of LIF neurons. layer one consists of 160 neurons and layer two consists of 11 neurons (one per object shape), in which the connection between the two layer were all to all connection through weighted synapses. (Right) WTA structure composed by global inhibitory neuron. 55
- 4.1 Spiking neural network for edge orientation with designed receptive fields: (A) Skin patch from the iCub robot and layer one, which encodes the analog signal into spikes. (B) First order tactile neurons (layer two) gathering the 160 inputs from layer one, organised in receptive fields. (C) Layer three discriminates edge orientation and includes a global inhibitory neuron that implements the WTA network. 61
- 4.2 Firing rates of neurons in layer three for different stimulus orientations; with (red) and without (green) WTA. 62
- 4.3 Uniform structure of receptive fields. 64
- 4.4 Random structure of receptive fields. 65
- 4.5 Random structure with subfields of receptive fields. 66
- 4.6 Example of the connected sensors study experiment with the random receptive fields structure; (left) 16 connected sensors to each receptive fields (each color represents the mechanoreceptors (sensors) of one receptive fields). (right) 2 connected sensors for each receptive fields. . . 70

4.7	Mutual Information for each receptive fields' structure, as a function of connected sensing elements (mean and std shaded).	71
4.8	Edge length experiment. 36 orientation ranging from 0 ° to 180 ° with 5 ° step increments applied on the skin patch with changing bar from 11 cm to 1 cm with step 1 cm.	72
4.9	Minimum detectable angle as a function of bar length (mean and std shaded).	73
4.10	Center position study (first experiment); 90 ° oriented bar pressed on the skin with 15 different center position.	74
4.11	mutual information as a function of 4 different bar orientations for the three different receptive fields structures (uniform, random with subfields, and random); each orientation applied on the skin with 15 different center positions.	75
4.12	Center position study (second experiment); 36 orientations ranging from 0 ° to 180 ° with 5 ° step increments applied on the skin in 12 trials, in each trial we change the center position.	76
4.13	Mean mutual information and std (shaded) as a function of center position for the three different receptive fields structures (uniform, random with subfields, and random).	77

-
- 4.14 Center position study (third experiment); (left) 36 orientations ranging from 0° to 180° with 5° step increments applied with bar changing from 11 cm to 5 cm with step 1 cm. (Right) same as in the left part, whereas the center position of the 36 orientation is different. 78
- 4.15 Mutual information as a function of edge length and center position for the random structure of the receptive fields. 78
- 5.1 Toy example of two neurons connected via SDSP synapse. (A) Pre-synaptic neuron spike arrives at time t_k . (B) Post-synaptic membrane potential. (C) Post-synaptic calcium variable. (D) Synaptic weights variation as a function of time. If post-synaptic calcium variable and membrane potential meet the requirements at the moment of the pre-synaptic spike arrival, the synaptic weights potentiated by value A_+ . . . 87

- 5.2 SDSP Learning rule with Baseline Network. (A) Baseline network with all to all connection between layer two neurons and layer three neurons endowed with SDSP synaptic learning rule to build the required connectivity through potentiation or depression of the synaptic weights and to learn an appropriate connectivity patterns for the classification. (B) Synaptic weights at the initial and final state of the simulation; (left) The synaptic weights between layer two and layer three were initialized using uniform random distribution such that all the weights were depressed (red boxes); (right) The synaptic weights at the end of the learning, each small box represents the synaptic weights of one neuron in layer two (x-axis) connected to one neuron in layer three (y-axis), where most layer two neurons to layer three neurons connecting weights were still depressed (red) and some potentiated (green). 92
- 5.3 Spiking responses of the three layer of the neuromorphic network architecture. (A) Examples of different bars pressed on the skin at different orientations (0° , 15° , 75° , 105° , 165°). (B) Raster plot of layer one neurons (SA-1 afferents) as a function of time for 36 orientations (0° to 180° with step 5° increments) (left), raster plot of layer two neuron (receptive fields) (middle), and (right) raster plots of layer three neurons as a function of time for 36 orientations with baseline network (colored spikes trains match the orientations given in part (A) and the black spike trains for the remaining orientations). 93

5.4 Network performance for the baseline networks based on mutual information. (A) Mutual information as a function of epochs with constant pressure dataset; in each epoch 36 orientations ranging from 0° to 180° with step 5° increments, randomly repeated, and all with constant pressure. (B) The benchmark of baseline network when varying pressure stimuli dataset were used. 94

- 5.5 SDSP learning rule with weights normalization network. (A) Schematic representation of the weights normalization mechanism between layer two neurons and layer three neurons; synaptic weights connecting all layer two neurons to a single neuron in layer three were normalized by dividing the weights by the summation of the activated synapses (i.e., how many weights exceeds the normalization threshold). (B) Synaptic weights at the initial and final state of the simulation; (left) The synaptic weights between the two layers were initialized using same synapses random distribution as the previous example; (right) The synaptic weights at the end of the learning, each row represents the synaptic weights connecting all neurons in layer two to one neuron in layer-three state: depressed (red) or potentiated (green), where the weights normalization mechanism help the learning procedure in a balance of activity between different neurons and converging the synaptic weights to stable connectivity patterns. (C) Number of orientation detected by weights normalization network for 10 different trials, by changing the weights initialization random values in each trial. 97
- 5.6 Raster plot of layer three neurons as a function of time for 36 (ranging from 0° to 180° with 5° step increments) orientations with weights normalization network (colored spikes trains match the orientations given in figure 5.3-A and the black spike trains for the remaining orientations. 98

- 5.7 Network performance comparison for the two different networks (Baseline, Weights Normalization) based on mutual information. (A) Mutual information as a function of epochs with constant pressure dataset; in each epoch 36 orientations ranging from 0° to 180° with step 5° increments, randomly repeated, and all with constant pressure. (B) The benchmark of the two networks (baseline and weights normalization networks) when varying pressure stimuli dataset were used; the weights normalization network Network outperformed the baseline network by detecting more stimulus orientations. 99
- 5.8 Raster plot of layer three neurons as a function of time for 36 (ranging from 0° to 180° with 5° step increments) orientations with Homeostasis network (colored spikes trains match the orientations given in figure 5.3-A and the black spike trains for the remaining orientations. 102
- 5.9 Network performance comparison for the three different networks (Baseline, Weights Normalization, and Homeostasis) based on mutual information. (A) Mutual information as a function of epochs with constant pressure dataset; in each epoch 36 orientations ranging from 0° to 180° with step 5° increments, randomly repeated, and all with constant pressure. (B) The benchmark of the three networks when varying pressure stimuli dataset were used; the homeostasis Network outperformed the weights normalization and baseline network by detecting more stimulus orientations. 102

- 5.10 Accuracy of the three networks as a function of stimuli with different intervals of times; the duration of stimuli changes from $2ms$ to $20ms$ with a step increment of $2ms$, both Homeostasis and weights normalization networks outperformed the baseline network in detecting a different number of orientations with different time interval. 103
- 5.11 STDP learning phase; (Left) variation of synaptic weights connecting all neurons in hidden layer to one neuron in output layer for each presynaptic spike as a function of time. (Right) summation of synapses weights at the end of learning (green for potentiated synapses and orange for depressed synapses) connecting all neurons in hidden layer to each neuron in output layer(x-axis). Orange bars are multiplied by -1 for better presentation. 105
- 5.12 Confusion matrix for touch modalities classification. 106
- 5.13 Variation of the synaptic weight for each presynaptic spike as a function of time. Each box represents the synaptic weights between all neurons in layer one to one neuron in layer two. 109
- 5.14 Output spikes of layer one and two:(top) 11 different object shapes; (middle) each object shape corresponds to the activation of multiple neurons in layer one; (bottom)Output spikes of neurons in layer two after learning, such as for each object shape only one neuron in layer two is firing. 110
- 5.15 Network accuracy as a function of number of samples. 111

5.16 Real time object shape classification setup. 113

List of Tables

- 2.1 Characteristics of human mechanoreceptors. 16

- 5.1 SNN computational cost 112

Chapter 1

Introduction

1.1 Motivation

Recent technological advances have shifted the role of robots from simple industrial tools in controlled environments to intelligent devices that can assist humans and interact with the environment [1]. Several applications have emerged of robots' simulation of human behavior such as humanoids, medical robots, artificial limbs, and social robots. This shift creates new challenges that must be addressed to ensure safe and efficient interaction between robots and humans while providing manufacturing, entertainment, education, healthcare and healthcare services. In order to acquire autonomous learning, safe operations, and work in unpredictable environments, robots are made capable of navigating their environment through the equipment of sensory tools. One particular example of such sensory tools that play a critical role in shaping interactions with the environment is tactile perception [2]. For robots to acquire efficient

tactile perceptions, they are expected to perform complex tasks such as distinguishing between different edge orientations, differentiation between different touch modalities, grasping objects of different shapes and sizes. Therefore, robots and hand prostheses endowed with the sense of touch can better and more easily manipulate objects and physically collaborate with other agents.

Evidently, tactile sensing in biological systems has superior performance and robustness. Human mechanoreceptors are distributed along the skin with complex overlapping receptive fields [3–5]. Different from the case of visual receptive fields, a common agreement on tactile receptive fields are still missing. Studies suggested that the tactile receptive fields are similar to the visual ones but shaped differently due to the different stimuli they interface to. Vision seems to handle more natural scenes with several objects, and a heterogeneous background, while touch is mainly related to surface textures and regular repetitions. Therefore, an analysis to define how the sensitive areas (mechanoreceptors) of receptive fields are distributed on the skin is needed, in order to design artificial skin with pre-processing embedded capabilities.

A particular study in the field of neuromorphic engineering is dedicated to the emulation of the neuronal function and organization of the human nervous systems in electronic devices in the aim of advancing the efficiency and robustness of computational interactions with environments. Mimicking the communicative function of neurons' action potentials through spikes that encode and propagate information, represents an example of such emulations [6]. Neuromorphic and event-driven sensors are modeled based on biological models to perform the function of dealing with stimuli

from the environments as spikes. Besides sensors, bio-inspired computational frameworks such as spiking neural networks have also been developed [7]. By modeling the complex dynamics of spiking neurons, learning and plasticity mechanisms observed in biology can be implemented. Moreover, a neuromorphic approach of integrating the hardware sensory system along with spiking neuron models mimics the behavior of human mechanoreceptors and could perform computations much faster than conventional computers, while drawing significantly less power. There is thus potential to revolutionize the field of tactile sensing through the use of neuromorphic principles and hardware.

1.2 Objectives

Motivated by the efficiency of biological systems in processing information, we aim to study and develop a biologically inspired, spiking models architecture that mimics human tactile perception with computational primitives that are implementable on low-power subthreshold neuromorphic hardware. More specifically, the model architecture aimed to solve the problem of stimulus orientation detection, touch modalities classification, and object shape classification. The model consists of necessary sensors (e.g. capacitive sensors from iCub robot, piezoresistive sensors) and spiking neural network of Leaky Integrate and Fire neurons; the output analog signals from sensors are used as an input to the spiking neural network that process and convert them to neuromorphic spikes trains, and algorithms that interpret them through spike driven

learning mechanisms. Several objectives are listed as follows, which will enable the successful fulfillment of the project:

1. Development of neuromorphic spiking neural network for analysis of tactile information and capable of distinguishing different stimulus applied on skin patch that consists of 160 sensors. We aim to develop intelligent SNN that can be endowed with spike-driven learning capability and adaptation.
2. Exploration different topologies of the overlapping (and interleaved) receptive fields, defining how the single sensitive areas (mechanoreceptors) are distributed in the receptive fields. We aim to find optimal distribution of mechanoreceptors in receptive fields for increasing the orientation acuity by applying a methodology that based on mutual information theory.
3. Endowing the spiking neural network with biologically inspired spike driven learning rule for interpretation and distinguishing between different stimuli. We aim to examine whether the model can build the appropriate connectivity patterns through local unsupervised and supervised spike-driven learning that exploits the temporal coincidence of input spikes and neuron's activity in three different application scenarios as follow: (1) edge orientation selectivity, (2) touch modalities classification, and (3) object shape recognition.
4. Investigating different synaptic normalization mechanisms that act on all synaptic weights to change the synaptic drive of the neuron and stabilize its activity during the learning procedure and at the inference phase. We aim to solve classification

problems in SNN's with edge orientation application that allow the synaptic weights to change progressively during learning converging to stable connectivity patterns, increasing the spatial acuity of the network.

5. Validating the neuromorphic spiking neural network on different application scenarios to assess the generalization of the achieved results.

1.3 Contributions

The major contributions of this thesis are listed below:

1. Developed a neuromorphic architecture model composed of three layers of Leaky Integrate and Fire neurons that used the output of necessary sensors distributed along a artificial skin patch as an input. We demonstrated that the patch of skin and the first layer of the SNN models the output of biological SA-1 mechanoreceptors, the second layer gathers the high-dimensional input of the previous layer into a compressed representation, and the third layer neurons responsible for decoding the stimuli presented. Moreover, the architecture is based on computational primitives that have a correspondent hardware implementation using neuromorphic sub-threshold CMOS technology.
2. Proposed an approach for estimating the optimal distribution of sensitive areas in receptive fields for increasing the orientation acuity. We demonstrated that the receptive fields created by randomly selecting sensitive points perform better

than structured receptive fields with uniform distribution in discriminating small angles (down to 5 degrees with edge length equal to 11 cm), as well as the orientation discrimination, gracefully degrades with decreasing edge length (up to 60° with edge length equal to 1cm). Moreover, we show the robustness of the model to edge orientation encoding when the receptive field's density decreases, where nine sensors connected to each receptive field with random distribution were enough to encode eight different stimulus orientations applied on the skin patch.

3. Developed an unsupervised spike-driven learning algorithm (Spike Driven Synaptic Plasticity) that responds to spatio-temporal spikes patterns with temporal coincidence. As a learning rule, we implemented a biologically plausible model that exploits the temporal correlation of input spikes and neuron's activity. We demonstrated that the network endowed with SDSP learning rule can learn an appropriate connectivity pattern for edge orientation selectivity, as well as the network is able to discriminate between different orientations with an angular resolution of 5 °.
4. Deployed synaptic normalization mechanisms to achieve robust and adaptive network architecture. It was verified that the developed system is capable of adapting to changes in sensor pressure. We demonstrated that under the edge orientation classification application scenario, in the developed network with weights normalization mechanism the current magnitude of each synapse changed, depending

on how many active synapses were connected to the decoder layer neuron. This allows the synaptic weights to change progressively during the learning phase converging to stable connectivity patterns, increasing the spatial acuity of the network. In addition, the main advantage of the homeostasis mechanism over the weights normalization mechanism is that the homeostasis performs better with stimuli with varying pressure. In the constant pressure stimuli, both the weights normalization and the homeostasis networks act the same both detecting thirty-five orientations. However, in the varying pressure stimuli, the homeostasis network outperforms the weights normalization by detecting 35 orientations compared to 31 orientations detected by the weights normalization network.

5. Developed a supervised spike time dependent plasticity (STDP) that responds to spatio-temporal spikes patterns with temporal coincidence. As a learning rule, we implemented a SNN model that exploits the temporal correlation of input spikes and neuron's activity. We show that the SNN network endowed with supervised-STDP can learn an appropriate connectivity pattern for classification of touch modalities and object shapes. The proposed network achieves a total accuracy of 88.3% for classifying six different touch modalities. Moreover, the network with two layers has the ability to differentiate between 11 different object shapes with total accuracy equal to 100%.

1.4 List of Publications

This dissertation is based in part on the following publications:

- Dabbous, A., Mastella, M., Natarajan, A., Chicca, E., Valle, M., & Bartolozzi, C. (2021, May). Artificial Bio-inspired Tactile Receptive Fields for Edge Orientation Classification. In 2021 IEEE International Symposium on Circuits and Systems (ISCAS) (pp. 1-5). IEEE.
- Dabbous, A., Ibrahim, A., Valle, M., & Bartolozzi, C. Touch Modality Classification using Spiking Neural Networks and Supervised-STDP Learning. In 2021 28th IEEE International Conference on Electronics, Circuits, and Systems (ICECS) (pp. 1-4). IEEE.
- Ali Dabbous, Ali Ibrahim, Mohamad Alemeh, Maurizio Valle, & Chiara Bartolozzi. " Object Contact Shape Classification Using Neuromorphic Spiking Neural Network with STDP Learning " (accepted at International Symposium on Circuits and Systems ISCAS 2022)
- Ali Dabbous, Michele Mastella, Elisabetta Chicca, Maurizio Valle, & Chiara Bartolozzi. "Neuromorphic Spiking Neural Network With Synaptic Normalization Mechanisms For Edge Orientation Classification In Artificial Bio-Inspired Skin". (To be submitted to Nature Scientific Report)

1.5 Organization of the Thesis Document

The rest of the thesis is organized as follows:

Chapter 2 provides an overview of biological and artificial tactile sensing. A detailed overview of applications and studies conducted in the state-of-art in the field of feature extraction and edge orientation selectivity.

Chapter 3 introduces our neuromorphic approach for the recognition and classification of tactile patterns. A detailed description of our experimental setup and network architecture for each of the following applications: (1) edge orientation selectivity, (2) touch modalities classification, and (3) object shape recognition.

Chapter 4 provides analysis for estimating the optimal distribution of sensitive areas in receptive fields for increasing the orientation acuity. A detailed description of experimental procedures and obtained results are provided.

Chapter 5 illustrates the methodologies that we adopted to construct a neuromorphic spiking neural network that can learn complex patterns using only neurons, synapses, and local learning. learning procedures, synaptic normalization mechanisms, and results for solving edge orientation detection problems are provided. A detailed description of the learning and inferring procedures for solving touch modalities and object shapes classification problems, along with results obtained are also provided in this chapter.

Chapter 6 concludes this thesis, summarizes major contributions and presents possible research directions for further investigation.

Chapter 2

Literature Review

2.1 Introduction

The sense of touch plays a critical role in enabling human beings to interact with the surrounding environments. As robots move from laboratories to domestic environments, they are expected to be endowed with a similar tactile ability to perform complicated tasks such as manipulating objects with arbitrary unknown shapes. In the past decade, tremendous effort and progress have been made to mimic the sense of touch in human beings on robotic systems. In this chapter, we will provide an review of the human sense of touch (Sec. 2.2), followed by a review of the artificial sense of touch (Sec. 2.3). Finally, we will provide an overview of application and studies conducted in the state-of-art in the field of feature extraction and edge orientation selectivity (Sec. 2.4).

2.2 Human Sense of Touch

2.2.1 Introduction

What if we had all forms of perception except for "touch"? Various object exploration experiments can answer this question through, for example, soaking hands in ice for an extended period of time. In [8], the authors found that it was difficult for a group of subjects to grasp objects in a stable manner after anesthetizing the skin on their hands during the experiments. Furthermore, when the sense of touch is lost, movements become imprecise and unstable. In a separate and unique experiment conducted on astronauts at the International Space Station, the vibro-tactile signals supplied via "sense of touch" appeared to be highly predictive of direction and spatial disorientation [9].

Effectively, "sense of touch" can help evaluate the attributes of an object, including its size, shape, texture, and temperature. This evaluation can manifest itself through detecting slips and developing perceptions of the body while rolling the object between fingers without dropping it, and therefore distinguishing between "I" and "I am not". Sense of touch also helps avoid injuries and dangers by alerting us to physical trauma situations through activating a sensation of pain. It can also act as a mediator for sharing and communicating emotions through social interactions. Therefore, without the sense of touch or an alternative type of sensing modality, there would be a wider gap between what is detected and what is perceived.

There are two primary subcategories of the human sense of touch—the main difference between the two depends on the location of sensory input. The first sub-form refers to the “cutaneous” sense which corresponds to receiving sensory input from skin receptors and is responsible for detecting pressure, temperature and pain [10]. The second sub-form is the “kinesthetic” sense which entails receiving input from sensory receptors located in the muscles, tendons, and joints; this functions to help approximate the position and orientation of limbs in space [11]. For the purpose of this thesis, we will primarily be concerned with cutaneous tactile perception. The process of human touch sensation is highly decentralized and encompasses all body areas. Triggered by contact with an object, cutaneous mechanoreceptors on the skin transform mechanical impulses to electrical action potentials (i.e., spikes) which then deliver sensory information to the Central Nervous System (CNS). Subsequently, this information is communicated to the primary somatosensory cortex by neural afferents and then projected onto the secondary somatosensory cortex for decoding. Ultimately, this information permits individuals to detect object qualities, such as shape, stiffness, texture, curvature, and orientation among other properties [12]. The remainder of this section will provide an overview of the mechanoreceptors found in the human skin (Sec. 2.2.2) followed by a section on the tactile pathway and information processing (Sec. 2.2.3).

2.2.2 Mechanoreceptors

There are different receptors positioned on the skin that mediate tactile sensing through managing responses to external stimuli. While Mechanoreceptors are responsible

for responding to mechanical stimuli, other receptors handle thermal gradients (i.e., thermoceptors) and pain triggered by mechanical, thermal or chemical stimuli (i.e., nociceptors). However, given that the latter are not as relevant for tactile sensation [3], we will mainly focus on mechanoreceptors and their role in mechanical touch. Positioned in the outmost layers of the skin, mechanoreceptors are innervated by neural afferents that transfer tactile sensory information to the primary somatosensory cortex. Mechanoreceptors convert mechanical stimuli into sequences of electrical discharges which get propagated to the central nervous system for further processing. The four principle tactile mechanoreceptors in the human skin are: Merkel's disks, Meissner's corpuscles, Ruffini endings, and Pacinian corpuscle. Each receptor is uniquely concerned with specific stimuli. While the Merkel's disks and Meissner corpuscles receptors are situated near the surface of the skin, neighboring the epidermis, the Ruffini endings and Pacinian corpuscles are rooted more deeply in the dermis and subcutaneous tissue. Additionally, the various mechanoreceptors are differentially concentrated where the Meissner's corpuscles are the most common, comprising 40 % of the mechanoreceptors in the human hand, while Merkel cells comprise 25 % and the Pacinian corpuscles comprise 10-15 % (See Figure 2.1 for illustration).

Mechanoreceptors can be categorized into two subcategories, slowly adapting (SA) and fast adapting (FA), with each having separate functionalities and receptive fields [3, 4]. The first point of distinction between the two is that SA mechanoreceptors maintain activity during the entire period that a mechanical stimulus is detected; however, FA mechanoreceptors' response is more intermittent, witnessing surges of action potentials

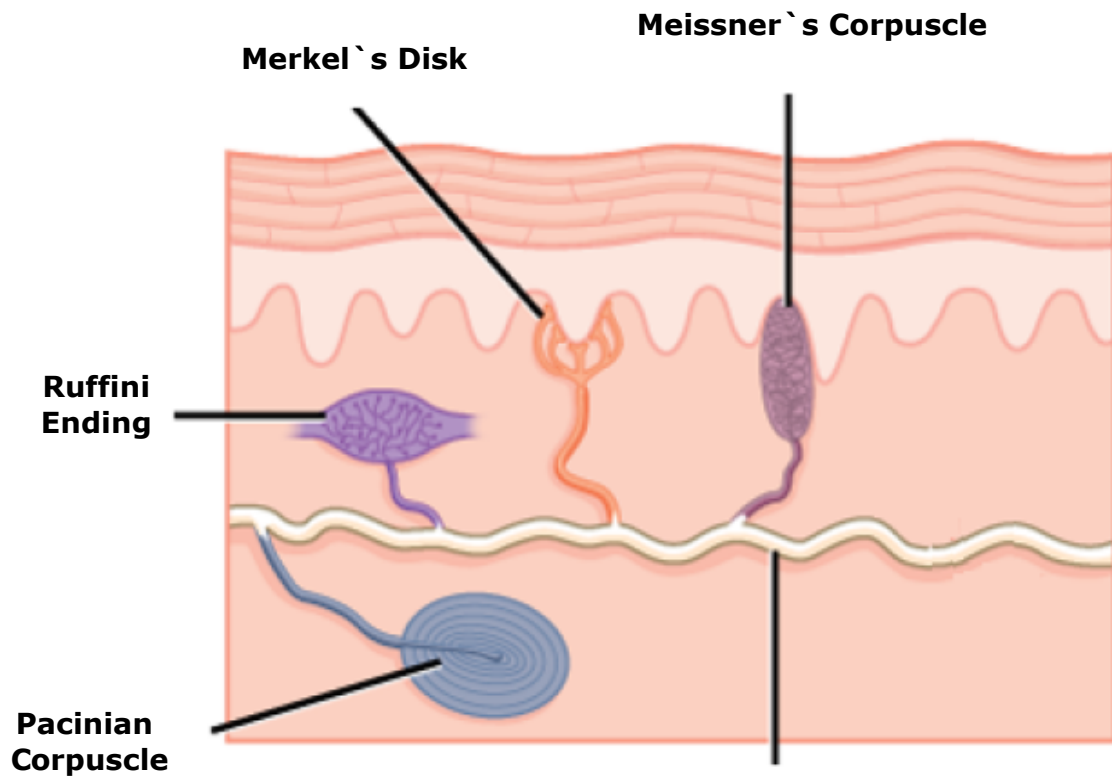


Figure 2.1 Schematic representation of mechanoreceptors in human skin.

exclusively upon the initial application of the preferred stimuli and then upon its removal [5].

In turn, the FA and SA receptors can also be categorized into two distinct subcategories according to their receptive field properties. Type I FA receptors refer to Meissner corpuscles which are located close to surface level on the skin and are especially concentrated around fingertips (about 140 units/cm²). With small receptive fields that constitute multiple maximal-sensitivity zones disseminated across a circular or oval area of 5-9 epidermal ridges, these receptors could detect relatively high-frequency dynamic skin deformations along with vibrations with low frequencies, flutter slip and motion. On the other hand, located deeper in the dermal and subdermal fibrous tissue,

Type II FA receptors correspond to Pacini corpuscles which innervate the hand in less density (of approximately 20 units/cm²) [4]. With a wider receptive field and one exclusive maximal-sensitivity zone, these receptors are activated when objects held by the hand come in contact, or lose contact, with other objects.

As for SA receptors, Type I is signified by Merkel discs which are located close to the surface level in the skin (with a density of about 70 units/cm²) and are linked to sensory nerve endings [4]. These receptors have multiple maximal-sensitivity zones, are activated by lower-frequency skin deformations, and are especially important for discriminating texture and fine form detection. Additionally, their discharge rate is identified as irregular upon the introduction of continuous stimuli. Finally, Type II, which refer to Ruffini endings, are positioned more deeply within the subcutaneous tissue and are responsible for detecting shape along with kinesthetic sense and perception of finger and joint position and movement [4]. These receptors hold one exclusive maximal-sensitivity zone and show a regular discharge rate upon sustaining stimulation.

Table 2.1 presents a summary of the characteristics of each mechanoreceptor [13].






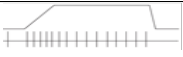


	Pacinian Corpuscle	Ruffini Corpuscle	Merkel Cell	Meissner's Corpuscle
Sensation	pressure, fast, vibration, and tickling	stretching of skin and deep pressure	fine touch and pressure	fine touch, pressure, and slow vibration
Receptive Field				
Adaptation				
Adaptation rate	Fast	Slow	Fast	Slow

Table 2.1 Characteristics of human mechanoreceptors.

2.2.3 Tactile Pathway and Information Processing

Upon contact with an object, the skin conforms and maintains the same local contour as the surface touched. This deformation is projected to a population of receptors in the skin, each responsible for a small area of the contour. The mechanoreceptors defined in Sec. 2.2.2 then respond to their respective preferred stimulus (i.e. static or dynamic stress/strain) by emitting action potentials. This signifies that no single afferent or class of afferents can provide the complete information of the contact event. These action potentials are then transferred to the brain using peripheral nerves that enter the spinal cord through the dorsal root [14].

The transfer of tactile information follows two separate pathways in the spinal cord, the medial lemniscal pathway and the spinothalamic pathway. The medial lemniscal pathway is dedicated to signals from the four cutaneous mechanoreceptive afferents that innervate the glabrous skin. The signals approach the spinal cord through the dorsal root ganglia and move up towards the ipsilateral posterior column until they synapse onto neurons in the cuneate nucleus which is located within the dorsal column nuclei of the lower brainstem. The neurons then process the tactile afferent information received from the mechanoreceptors and subsequently propagate this information to the thalamus and somatosensory cortex in the brain [15, 16]. It is along this sensory pathway that the decoding of the sensory information takes place, beginning with the mechanoreceptors, neurons of the medial lemniscal pathway, and somatosensory cortex to the brain which ultimately perceives and makes sense of the information [17, 18].

There are various tactile sub-modalities and afferent types through which stimulus information is presented. While different afferent types collaboratively convey tactile information, they differ in the ways they follow to convey this information. In their article, Jorntell et al.[18] showed that cuneate neurons uniquely respond to sensory inputs, which proposes that each neuron holds a distinct task-related combination of input features. Additionally, Pruszynski et al. [19] presented results that suggested that Type I, FA and SA, receptors are selectively sensitive to the orientation of edges. There are various differentially-distributed transduction sites corresponding to these receptors; this produces complex receptive fields that are highly sensitive to specific edge orientations. As such, the results indicate that the site of tactile information processing is not exclusively in the brain and central nervous system, but occurs along the tactile pathway during the transmission of sensory information

2.3 Artificial Sense of Touch

2.3.1 Introduction

The sense of touch is an important gateway to interact with the environment [1]. Robots and hand prosthesis endowed with the sense of touch can better and more easily manipulate objects and physically collaborate with other agents. Even though autonomous robots mainly rely on some form of visual perception to interact with the surrounding environment, there are tasks that would be impossible or too complicated without the sense of touch [20].

To acquire contact information from visual input necessitates complex 3D scene resolution which challenges the effectivity of robots in dynamic environments. In effect, there are many benefits for introducing tactile information in robotics. Case in point, tactile information can be useful as a control parameter in manipulative tasks [21–23] where the necessary information entails an estimation of the contact point, surface normal and curvature measurement, as well as slip detection [24] through measuring normal static forces. Measuring the contact forces is necessary for the ability to maintain stable grasps as it permits grasp force control [25]; in addition to a grasp force control, a manipulator displacement is required in compliant manipulators [26]. In dexterous manipulation, the magnitude and direction of force are essential to guarantee grasp stability, or what is referred to as friction cone [27]. Additionally, shear information is crucial for full grasp force and torque determination [28, 29].

The tactile sensing specific robot-environment interaction has overwhelmingly been limited to measuring static interaction forces while having neglected dynamic forces, although both forces are present in real-world interactions. In addition, the majority of sensors are developed to measure static pressure or forces which does not encompass the ability of obtaining information about the stickiness, texture, hardness, elasticity, etc. However, more recently, there have been more efforts to acknowledge the centrality of dynamic events and to develop sensors to detect stress changes [30], [31], incipient slip [32], strain changes, as well as other temporal contact events. For example, sensors with the ability to identify the shape, size, position, forces and temperature of an object can be found in [33, 31, 34], along with others than can identify surface texture [35],

[36], hardness or consistency [37], [38], and friction [39] using almost every mode of transduction ranging from resistive/piezoresistive, tunnel effect, capacitive, optical, magnetic, piezoelectric, etc. In the remainder of this section, we will present a survey of the state-of-the-art transducer technologies for artificial skin in Sec. 2.3.2 along with a review of the tactile sensors implementation in robotics (Sec. 2.3.3).

2.3.2 Transduction Mechanisms

In robotics, the most commonly adopted sensing modes are capacitive, resistive, piezoelectric, optical and magnetic. This section presents a selection of techniques, describing their main features and drawbacks.

Capacitive Sensors The first sensing mode is capacitive which involves mainly two conductive layers with deformable dielectric material separating them—when applied, this deformable dielectric is caused by pressure which subsequently transforms the structure’s capacitance, and the measurement capacity allows for pressure to be estimated. Capacitive technology is particularly appropriate for robotics (specifically large areas) because of the presence of off-the-shelf components for the readout electronics and how compatible the sensors are with flexible substrates [40]. In addition, these sensors have impressive levels of sensitivity, compactness, and hold unlimited operational bandwidth (however, some dielectric material can limit the bandwidth to relatively low frequency). On the other hand, capacitive sensors are prone to noise triggered by crosstalk noise and field interactions (particularly in mesh configurations) and necessitate highly complex

electronics for processing and filtering out the noise [20]. Thus, this may negatively result in mechanical wear and tear, hysteresis, drift of sensitivity due to temperature, and in some cases relatively complex production processes, which ultimately leads to degrading the elastomeric material in the deformable dielectric.

However, more recently, in an effort to improve mechanical figures and durability and reducing hysteresis, dielectric materials in capacity sensors have been made of a thin layer of 3D fabric glued to conductive and protective layers [41]. More reproducible responses along with less costly and less complicated fabrication can be achieved through large-scale production.

Piezoresistive Sensors Resistive sensors have a generally straightforward design which is possible to execute on flexible printed circuit board. In these sensors, the resistance varying because of forces applied to the sensor can be measured using two electrodes [42]. Compact and simple voltage divider and an off-the-shelf analog-to-digital converter are needed for the readout electronics. Additionally, due to their simple detection mechanism and measuring resistance process, piezoresistive sensors need less complex electronics, are simple to produce and integrate with other elements in circuits, and are less sensitive to noise which renders them appropriate for array and mesh configurations. On the other hand, they have a lower frequency response than other sensors (e.g., capacitive sensors) because of hysteresis [43]; the negative properties associated with this sensor is high power consumption, hysteresis and short life of material.

Optical Sensors Generally, tactile sensing through optical means can be achieved in two ways. One way entails vision where a digital camera would detect deformations on a flexible membrane surface [44]. This process builds on developments in computer vision and relies on camera systems and algorithms to reproduce surface texture in 3D [45]. However, despite its sensitivity to surface texture, this process is not appropriate for large scale implementations because of the pressing challenge that curved surfaces introduce as well as space limitations. The second process involves measuring scattered light intensity as material deforms through using photo detectors [46]. In this case, the sensor includes conformable urethane foam, and is irradiated with LED light. Photo detectors beneath the skin detect the differences in scattered light intensity that form due to deformation of the foam. While there has been a previous demonstration of the distribution of 1864 sensing elements over a humanoid robot [47], the system sampling rate was not fast enough for manipulation tasks as it was below 20 Hz. There are several benefits for optical based tactile sensors, such as their ability to not get affected by electromagnetic interference (EMI), their high spatial resolution, and their insensitivity to lower frequency electromagnetic interferences by electrical systems [48] which consequently lowers their susceptibility to environmental noises. On the other hand, associated setbacks include lower performance in strong light conditions as well as higher power consumption. A measure of local pressure results from the response of solutions, which are based on multiple layers of optical media, to light diffusion inside the layers proceeding their deformation [49].

Piezoelectric Tactile Sensors Piezoelectric materials are appropriate for dynamic force sensing primarily because of their ability to produce charges, in a fast and linear manner over a large range of stimuli, when a force is introduced to the sensor. The structure of the sensor determines its sensitivity and the ability to differentiate between transverse, longitudinal and shear forces. Polymeric material, including, for example polyvinylidene difluoride, hold extended chemical stability and flexibility. As such, they have been useful to implement tactile sensors based on an integrated device, the POSFET (piezoelectric oxide semiconductor field effect transistor [50]) in which piezoelectric material detects charges produced by the force after being situated over the gate of a complementary metal–oxide–semiconductor transistor. The POSFET permits the readout circuitry and the sensing material to integrate, which then reduces noise and wiring and optimizes resolution. However, for the polymer to be deposited and polarized over the sensing elements array, flexible integrated circuits and a specific post-processing should be developed. Moreover, while Piezoelectric tactile sensors have an impressive high-frequency response which allows them to detect high dynamics such as vibrations, they are solely able to measure dynamic forces and not static forces because of their large internal resistance [51].

Magnetic Tactile Sensors There are two primary ways to develop inductive tactile sensors which are based on magnetic transduction, as reported in the literature [52, 53]. The first approach involves the sensors measuring the variation in magnetic coupling or inductance of a coil after a force or pressure is applied. However, the second category

involves measuring the variation in the magnetic flux, whether through using the Hall effect or through magnetoresistance. While these sensors have a highly dynamic range, they are bulky, have a low spatial resolution, and low repeatability due to hysteresis which stems from their mechanical nature [53]. Additionally, the technology is restrictive in robotics due to the detected signal being altered upon the interaction between the magnetic field and metallic objects in addition to the requirement of complex electronics to process and demodulate oscillating signal amplitude.

2.3.3 Tactile Sensors Implementation in Robotics

While developing functional and robust electronic skin is complex, there have been impressive accomplishments that satisfy the previously mentioned requirements which can be taken as a building foundation for the ultimate skin technology. This section will review technologies that have appeared to be reliable in different implementations with robots. First, we will introduce SynTouch [54], a bio-inspired multimodal fingertip which contains impedance sensors used to measure deformations proceeding normal or shear forces, pressure transducers which measure vibrations and pressure when sliding over textured surfaces, as well as temperature sensors. The sensing principle is resistive due to elastomers covering a fluidic structure. The fragile transducers are protected against environmental damage due to the propagation of the force signal to a remote position. The response to forces ranges from 0.1 N to 0.3 N while the sensor spacing is under 2 mm. However, the primary drawback is their inability to cover large areas along with their high cost. Additionally, there is no straightforward relation

between applied local pressure and the sensor's response which results in an increased complexity for activating the sensing elements upon pressure. Despite this, SynTouch has been applied to various robotic hands and is successfully utilized in several tasks under machine-learning techniques, such as discrimination of objects [55], control of slip [56], in-hand manipulation [57].

In [58], the authors put forth an electronic skin that builds on hexagonal PCB modules, with every one of them relying on three distinct types of sensors and performing local pre-processing with redundant connections to a mesh network structure. Large areas of the robot's body are covered through embedding elements to an elastomer which can adapt to curved surfaces [58]. Additionally, this elastomeric layer guarantees the protection of the sensors while also regulating the transducers' sensitivity. One of the benefits of this is its ability to extend over large areas with multiple modalities, such as temperature, vibrations and acceleration (3D accelerometer), light touch and proximity (optical). Moreover, collision avoidance is achieved through proximity sensors while collision detection is achieved through the work of accelerometers [58]. As for, robot self-calibration, it is made possible through consolidating the data from accelerometers and tactile units [58].

Moreover, authors in [41] suggest an alternative flexible capacitive skin which covers large and small areas of the robot's body (e.g., fingertips). This alternative holds a triangular flexible PCB as its basic unit, with 12 capacitors as well as an off-the-shelf capacitance-to-digital converter. One of the 12 capacitors takes on the role of a reference to compensate temperature drifts. In addition, the connectivity issues for large areas

are impressively contained through the presence of up to 16 patches which serially communicate with a microcontroller to direct the signals received to a controller area network serial line. The dielectric and top layer of the capacitors are soft and flexible. Different materials are applied to fine-tune the sensitivity, hysteresis and durability of the skin as a function of the application desiderata [41]. Several robots have different versions of this solution, for example, for safe interaction with withdrawal reflexes, in human–robot interaction in cases of physical contact, for manipulation, and in learning-by-demonstration sessions. Responses from adjacent sensors are integrated to allow for stimuli localization with a higher resolution than sensor spacing since the response of the sensor is analog (super-resolution [59]). Moreover, there have been solutions implemented specifically for prosthesis [60–62].

2.4 Feature Extraction and Edge Orientation Selectivity

In this section, we provide an overview of applications and studies conducted in the state-of-art, starting with a review of feature extraction and tactile perception (Sec. 2.4.1), followed by an overview of Spiking Neural Networks and Neuromorphic Approaches (Sec. 2.4.1.2), and finally a look into previous work surrounding the edge orientation selectivity (Sec. 2.4.2).

2.4.1 Feature Extraction and Tactile Perception Overview

2.4.1.1 Traditional Machine Learning

Employing machine-learning methods for tactile data processing may facilitate the use of tactile sensing systems in many domains, such as robotics, industrial automation, and biomedical devices. Such methods are important for enabling intelligent tasks to be implemented in response to the requirements of the application domains. Pattern recognition algorithms are frequently useful in certain tasks where material and texture classification are targeted [63]. Thus, many researches have recently directed their efforts towards the classification/recognition of materials' textures, patterns, and shapes [64, 65]. Drimus et al. [66] applied k-nearest neighbor classifier and dynamic time warping for the classification of rigid and deformable objects. They equipped a robotic gripper with the tactile sensors which they then used to palpate and squeeze the test objects; the emerging patterns were subsequently classified using k-nearest neighbor classifier and dynamic time warping. The issue around estimating roughness through the k-nearest neighbor method was addressed in Oddo et al. [67]. Moreover, Decherchi et al. [65] compared between the support-vector machine (SVM), regularized least square (RLS) and regularized extreme learning machine (ELM) to categorize materials from raw sensor data. Furthermore, Pezzementi et al. [68] applied an appearance-based method for object recognition, obtaining the most informative features through utilizing a bag-of-features classification technique along with exploratory movements.

Similar approaches for object identification [69, 70], texture recognition [71] and compliance characterization [72] have also been reported in the literature.

However, the classification of gestures and touch modalities such as poking, squeezing, pushing and grabbing is addressed in fewer studies [73, 74]. Naya et al. [75] were able to recognize five touch modalities through employing the k-nearest neighbor (KNN) algorithm that supports tactile interfaces for pet-like robots . Furthermore, authors in [76] relate touch interpretation and social interaction through supervising a ‘LogicBoost’ algorithm which differentiates between eight touch modalities upon the interaction of subjects and artificial arms. In addition, machine learning methods have also been employed in material classification [77], surface-roughness classification [64], and texture and pattern recognition [78].

Neural network algorithms have also been used for recognition and classification. In [79], authors explored the potential of recurrent neural networks (RNN) models to be used in touch modality classification. Liu et al. [80] performed object shape recognition using low-resolution pressure maps. They extracted scale and position invariant features from the tactile map and used a neural network for classification of the extracted features. Furthermore, Petriu et al. [81] employed a neural network classifier to categorize the tactile patterns emerging from 3D objects. Additionally, a neural network that estimates local curvatures was reported in [82], while an algorithm that detects slippage was reported in [83]. Furthermore, Schneider et. al. [84] categorized objects using low-resolution intensity images retrieved from multiple grasping interactions through employing a bag-of-words approach and an unsupervised clustering technique.

In addition, Madry et al. [85] along with Luo et al. [86] proposed recognizing tactile patterns through employing enhanced descriptors and feature learning methods. Moreover, Liu et al [87] designed a kernel sparse coding method to achieve tactile data classification and representation. Finally, Soh et al. [88] put forth an online learning algorithm to conduct object classification through the data collected by a five-fingered iCub robotic hand.

2.4.1.2 Spiking Neural Networks and Neuromorphic Approaches

Spiking Neural Networks (SNN) and synaptic learning have recently emerged as viable techniques to solve classification problems, with SNNs being computationally efficient and able to run on low-power hardware. The temporal plasticity, ease of application in neural interface circuits, and reduced computation complexity are the key benefits of SNN. The authors of [89] suggested a power-efficient tactile texture categorization framework that learns from the neural coding of conventional tactile sensor readings using SNN. A functional neural network that can use a neuron model with a two-layer SNN and a homeostatic synaptic learning mechanism to learn and distinguish the tactile characteristics of the input from a bionic touch sensor has been implemented in [90]. In [91], authors, proposed a biologically plausible system for object recognition based on tactile form perception comprises a snn of two layers. The results show that the proposed system gives good performance in recognising objects based on tactile form perception. In addition, there have also been studies addressing neuromorphic approaches to tactile perception. Pearson et al.'s [92] work was able to facilitate the

active exploration and navigation of a robot in its environment through replicating a neuromorphic vibrotactile sensory system that resembles a rat's whiskers. This system was able to build a model of large networks of leaky integrate-and-fire neurons which process vibrotactile information through the execution of hardware processing architecture [93, 94]. Moreover, Kim et al. [95, 96] experimented with modeling mechanoreceptors in glabrous skin for robotic and prosthetic applications. Furthermore, Bologna et al. [97, 98] investigated braille patterns decoding through primary and secondary neurons in a two-layer network; they then transformed the outputs of a force sensor array to spikes through employing a leaky integrate-and-fire neuron model [97–99]. In addition, Spigler et al. [100] and Lee et al. [101] applied an Izhikevich neuron model to define surface properties and incipient tactile stimulus. Moreover, Rongala et al. [102] converted tactile array outputs to spikes through employing an Izhikevich neuron model and revealed that the classification of ten naturalistic textures can be possible through the statistics of inter spike intervals and the Victor Purpura (VP) distance between spike trains—hence showing how feasible it is to utilize neuromorphic techniques to recognize texture. While the findings revealed that spike timing is crucial for texture classification, computation of the metrics cannot be implemented in real-time as they necessitate offline processing for the entire time window. However, Friedl et al. [103] suggested a solution to this problem through an alternative spiking neural network architecture. This network involved the neural engineering framework (NEF) [104] which employs a heterogeneous population of neurons to transform sensor data to spike patterns. Nonlinear frequency information was then obtained through

employing bandpass filters, ensuring the convolution of this activity through a hidden layer. Thus, a neurally-implemented support vector machine was employed to classify the high-dimensional feature representation. However, while it was possible to simulate the suggested network on neuromorphic hardware such as SpiNNaker [105], it was not executed and tested on any neuromorphic hardware platforms and hence remained only software based. Moreover, another limitation is that its task-specific architecture can have the potential to restrict its application in various of pattern recognition tasks.

2.4.2 Edge Orientation Selectivity Overview

Artificial tactile sensing has been gaining an increasing appeal, as it has been inspired by the computational efficiency of biology through, for example, encoding tactile information as a series of neuromorphic spike trains to mimic the mechanoreceptor's response characteristics. So far, neuromorphic tactile sensing has demonstrated its potential for being useful in developing neural interfaces that could provide tactile feedback to amputees by relying on upper limb neuroprosthesis [106, 107]. Edge orientation detection is tactile discrimination application that helps during object manipulation. More precisely, edge orientation detection could be seen as the basis for contour , as it helps in detecting the shape of an object, rendering it as one of the most central feature extractions in detecting the shape of an object. For instance, edge orientation-selective neurons have been observed in the first-order tactile afferents of human fingertips [19, 108] and are often used in artificial vision and touch as part of a pre-processing stage. In addition, evidence shows that tactile feature extraction can

already take place at afferent stages [109, 110] complemented with central information processing in the sensory cortex [111]. In Pruszynski et al. (2014) [19], the authors presented a network composed of two layers for edge orientation selectivity; the first consists of neurons with overlapping receptive fields (each neuron possesses a distinct distribution of highly sensitive zones on the skin), while the second is able to spot the temporal coincident activation between these neurons and decode the input stimulus orientation. In another work of Pruszynski and Hay [112], they aimed to show that synaptic integration across the complex signals from the first-order neuronal population could possibly underlie the human ability to accurately and rapidly process the orientation of edges moving across the fingertip. First, they derived spiking models of human first-order tactile neurons that fit and predict responses to moving edges with high accuracy. They used the model neurons in simulating the peripheral neuronal population that innervates a fingertip. By employing machine learning to train classifiers performing synaptic integration across the neuronal population activity, they were able to show that synaptic integration across first-order neurons can process edge orientations with high acuity and speed. In another study [108] authors recorded spike-trains generated by the overlapping receptive fields of FA-I and SA-I neurons and reported that humans could distinguish the edges spanning the entire fingertip with of edge orientation resolution. To differentiate between various edge orientations at different locations on simulated mechanoreceptor skin patches, [113] proposed 3 layers spiking neural model starting with first-order neurons to encode input stimulus into spike trains and ending with cortical neurons to decode edge orientation.

In robots, several solutions have been employed to establish fine orientation detection (up to 5°) on artificial skin, ranging from an AI-based vector regression method with offline learning [114] to a more neuromorphic approach with spike-based classification. In [115], authors implemented a model of spike-based neuromorphic encoding of tactile stimuli, emulating the discrimination properties of cuneate nucleus neurons based on pathways with differential delay lines. These strategies allowed the system to correctly perform a dynamic touch protocol of edge orientation recognition ridges from 0° to 40° , with a step of 5° . [116] conducted static edge perception experiments by tapping the stimuli with tactile sensors to demonstrate passive tactile perception for contour following specifically, the edge detection was applied only on right angles. In [117], the authors aimed to estimate the boundary edge orientation while rotating a piezoresistive tactile sensor which was attached to a robotic hand-over an object through suggesting a model based spatiotemporal correlation matching method. Their findings revealed support for the abilities of the suggested method to efficiently leverage spatial and temporal information by obtaining accurate orientation estimates ($\pm 1.67^\circ$ error for edges oriented from 10° to 90° , with a step of 5°) in spite of a low-resolution sensor (169 mm^2 , 4×4 resolution).

2.5 Discussion

In this chapter, we presented the ways that biological systems acquire and process somatosensory data, developed tactile sensors based on various techniques, algorithms,

and applications used in literature for interpretation and distinguishing tactile stimuli. In tactile sensing, mechanical pressures and distortions are picked up by mechanoreceptors by emitting action potentials, enabling the perception of features such as vibration, texture, friction, softness, etc. This action potential are then transferred to the brain using nerves that enter the spinal cord through the dorsal root. Mimicking every aspect of the biological tactile system is probably impractical due to engineering constraints, the understanding of tactile sensing in humans nevertheless provides useful insights on the organization and behavior of organisms in dynamically changing environments. The performance of biological tactile sensing helps to define the specifications when designing robotic artificial tactile sensors and electronic skins.

Researchers focused on designing tactile sensors that mimic the behavior of mechanoreceptors based on the various techniques. Besides tactile sensors, spiking models are considered one of the tools that can be used to mimic the response of mechanoreceptors found in human skin. While it may still interesting to investigate transducer technology for specialized applications, the focus is shifting towards technologies for large-scale implementation, interpretation, and distinguishing algorithms by integrating sensory systems into neuromorphic models which can learn and decode input stimuli.

The latest generation of neural networks incorporates temporal features for computation. These networks are known as spiking neural networks (SNN), where networks of neurons and their rich dynamic behavior are simulated artificially. Due to their biological plausibility, SNNs are useful for modeling information processes in the brain and investigating various learning mechanisms.

Like most analog and digital computing chips today, neuromorphic devices are built from silicon transistors. Researchers exploit the fact that when operating in the sub-threshold region, transistors exhibit many functional similarities with neurons. In particular, the transistor current is exponentially dependent on its terminal voltages, much like how the membrane potential of a neuron is exponentially related to the active populations of voltage-sensitive ionic channels [118]. Utilizing this property, electronic models of conductance-based neurons and synapses can be implemented in silicon to achieve biologically plausible computation primitives such as logarithmic functions, inhibition, thresholding, and winner-take-all selection.

Previous works presented in this chapter for solving classification and recognition problems, specifically, for solving edge orientation selectivity lack the possibility to be embedded on robots or neuromorphic chips, due to the need for offline learning and the presence of structures not easily transferable in silicon. Moreover, online learning is an important key for interpretation and distinguishing different input stimuli which are still missing in some previous work. In such situations, where energy and space are major constraints, a hardware implementation with online learning and low power devices is usually preferred. Such a methodology enables the system to perform end-to-end computation from the sensors to the processing and classification, consuming low power. Therefore, in this thesis, we targeted the development of an edge orientation architecture based on the event-driven acquisition and unsupervised spike-driven learning that can be implemented on low-power subthreshold neuromorphic hardware. Such as, the architecture can be used for solving other classification and

recognition problems (touch modalities and object shapes classification). Moreover, the unconstrained and random structure of the connectivity among layers can produce unbalanced activity in the output neurons, that are driven by a different number of synaptic inputs. This hypothesis is one of our main constraints through learning and inferring the input stimuli.

The rest of the thesis will explore how mechanisms in biological tactile sensing can be used to improve tactile systems, in a particular designing compact and efficient sensing devices that can locally pre-process the tactile signal.

Chapter 3

Neuromorphic Approach for Tactile Perception

3.1 Introduction

Biological information-processing systems operate on completely different principles than traditional computing systems. While computers are constructed from fast and high-precision hardware with high power dissipation, brains are composed of slow asynchronous neural components which use a combination of both analog and digital representations [119]. By emulating the neural organization and function of nervous system in electronic devices, neuromorphic systems aim to enable artificial systems that can operate with human-like intelligence at power efficiencies close to biological levels [120]. Moreover, they may be helpful in providing some insight into computational mechanisms of the brain. Neuromorphic sensory systems use a new form of

asynchronous output representation which carries timing information similar to spikes in the nervous systems [119].

In robotics, in order to mimic the tactile capability of biological mechanoreceptors, several types of tactile sensors have been developed based on various techniques (e.g., capacitive, piezoresistive, optical, magnetic, binary, and piezoelectric) [121, 61, 122]. Most of these sensors require circuits that sample the environment at a fixed rate and generate discrete data, regardless of the device being in contact with a stimulus. This requirement stresses the trade-off between accuracy and speed, which can lead to a slow or inaccurate reaction in dangerous situations (for example when operating near humans). On the other hand, biological sensors react to changes in the sensory signal, rather than sampling the sensory signal at fixed time intervals. This characteristic is known as event or data driven sensing. As it conveys data only at change, it reduces redundancy (no samples for constant signal), while it increases the sampling rate for fast changes. While neuromorphic circuits are being developed to encode the continuous analog signal from physical transducers into spikes [123, 124], sigma-delta conversion [125] or spiking neuron models can be used to convert the signals generated by front-end clocked artificial tactile sensors to neuromorphic spikes [126].

Inspired by the computational efficiency of biology, neuromorphic tactile sensing has been gaining an increasing appeal, using spiking neurons and artificial sensors to mimic the mechanoreceptor's response and the processing associated with the somatosensory nervous system [123, 102].

The rest of the chapter is organized as follow: neuromorphic approach for tactile perception was demonstrated in Sec. 3.2, followed by a section describing the sensors and network architecture used for edge orientation selectivity (Sec. 3.3), then the sensors and network architecture used for touch modalities classification (Sec. 3.4), finally a section demonstrating the sensors and network architecture used for object shape classification (Sec. 3.5).

3.2 Biologically Inspired Neuromorphic Approach for Tactile Perception

Neuromorphic systems can emulate spike-based neuronal mechanisms that are observed in natural effectors and senses, and thus have a potential application scenario in neuro-robotics and neuro-prostheses. In this section, we present a neuromorphic model for tactile perception analysis that relies on event-driven acquisition, based on computational primitives that are implementable on low-power mixed-mode sub-threshold neuromorphic hardware. The neuromorphic model comprises a sensory array and three layers of spiking neural network. The role and the implementation of each layer of the proposed spiking neural network was introduced in this section as follows: First, conversion from analog signal into neuromorphic spikes provided in Sec. 3.2.1. Second, the role of layer two (receptive field layer) demonstrated in Sec. 3.2.2. Third, modeling of layer three (decoder layer) as coincidence detectors

was provided in Sec. 3.2.3. Finally, to increase the selectivity of the network, the implementation of WTA was provided in section Sec. 3.2.4.

3.2.1 Touch Afferents: First-Layer Neurons

The first layer of neuromorphic models is usually designed to convert the pressure data produced from the sensory array into neuromorphic spikes. The mechanism used to recreate this behaviour is highly inspired by the studies on the sensors present on human skin, composed by two different parts: the mechanoreceptors and the afferents. The first ones are mechanical structures that release ionic currents when pressed. Different mechanoreceptors that encode touch are present on the human glabrous skin: Merkel disks, Meissner's corpuscles, Pacinian corpuscles and Ruffini's corpuscles [127]. Afferent neurons are instead nervous cells that convert ionic currents into spikes. Several studies argue in favor of a joint contribution between SA-I and RA-I [128]. A study [113] proposes instead that the way SA-I are modeled at the moment does not reflect the real behaviour of biological equivalents. The more faithful model, called in the mentioned work Dynamic SA-I, behaves like a hybrid between SA-I and RA-I. The advantage of such behaviour, according to the paper, is the relaxation of the trade-off typical of SA-I and RA-I where SA-I are resistant to noise, but RA-I are more accurate in the stimulus depiction.

In literature, spiking neuron models can be used to convert the signal generated by front-end clocked artificial tactile sensors to neuromorphic spikes [129, 130]. There are many models of spiking neurons (e.g. Hodgkin-Huxley, Leaky Integrate and Fire,

and Izhikevich). The fathers of the spiking neurons are the conductance-based neuron models, such as the well-known electrical model defined by Hodgkin & Huxley [131]. Hodgkin & Huxley modeled the electro-chemical information transmission of natural neurons with electrical circuits consisting of capacitors and resistor. In the class of spiking neurons defined by differential equations, the two-dimensional Izhikevich neuron model [130] is a good compromise between biophysical plausibility and computational cost. Derived from the Hodgkin-Huxley neuron model are Integrate-and-Fire (I&F) neuron models that are much more computationally tractable. An important I&F neuron type is the Leaky-Integrate-and-Fire (LIF) neuron [132]. Compared to the Hodgkin-Huxley model, the most important simplification in the LIF neuron implies that the shape of the action potentials is neglected, and every spike is considered as a uniform event defined only by the time of its appearance.

The variation of membrane potential v_{mem} for layer one neurons in LIF model at time t can be written as:

$$\frac{dv_{mem_i}}{dt} = \frac{v_{rest} - v_{mem_i}}{\tau_{mem_1}} + \frac{I_i(t)}{C_{mem_1}}$$

$$\text{if } v_{mem_i}(t) > v_{th}. \quad \text{then } S_{1_i}(t) \leftarrow 1; \quad v_{mem_i}(t) \leftarrow v_{reset}$$

In this equation, $I_i(t)$ represents the input analog pressure measured by the sensors and C_{mem_1} is the membrane capacitance. When applying an input current, the membrane potential v_{mem_i} of the layer one neurons starts to increase until reaching a fixed threshold v_{th} . At this point, a spike occurs and the membrane voltage is reset to its

resting value v_{reset} . S_{1_i} represents the spike train of layer one neurons. It is a function whose value is 1 when the neuron fires a spike at time t and 0 otherwise. The neuron is leaky since the summed contributions to the membrane potential decay with a characteristic time constant τ_{mem_1} .

Figure 3.1 represents the encoding of pressure signals into neuromorphic spikes procedure when a vertical bar pressed on artificial skin from the iCub robot. The constant pressing of a bar over the skin generates a sustained constant in the sensors, as shown in Figure 3.1-B. Depending on the applied stimulus, different sensors are activated at the same time. For example, a vertical bar placed in the middle of the skin results in a response only in central sensors. Each stimulus generates different analog signals that are fed as a current into layer one neurons. This process is similar to what happens in the human skin, where the Merkel disks convert stable pressure into analog ion currents that stimulate afferents neurons. In this thesis, afferents neurons (layer one neurons) are modeled using LIF neurons. LIF respond with tonic spiking to a sustained current [133], in which the firing rate of the neurons are proportional to the input pressure. This is similar to typical SA-I afferents, which have a sustained response for constant pressure stimuli [134].

All the simulation and models are implemented using Brian2 simulator [135].

3.2.2 Receptive Fields: Second-Layer Neurons

Once the tactile analog stimulus has been converted into spikes, the signal needs to be funneled into a further layer: the receptive field layer. In this part, the dimensionality

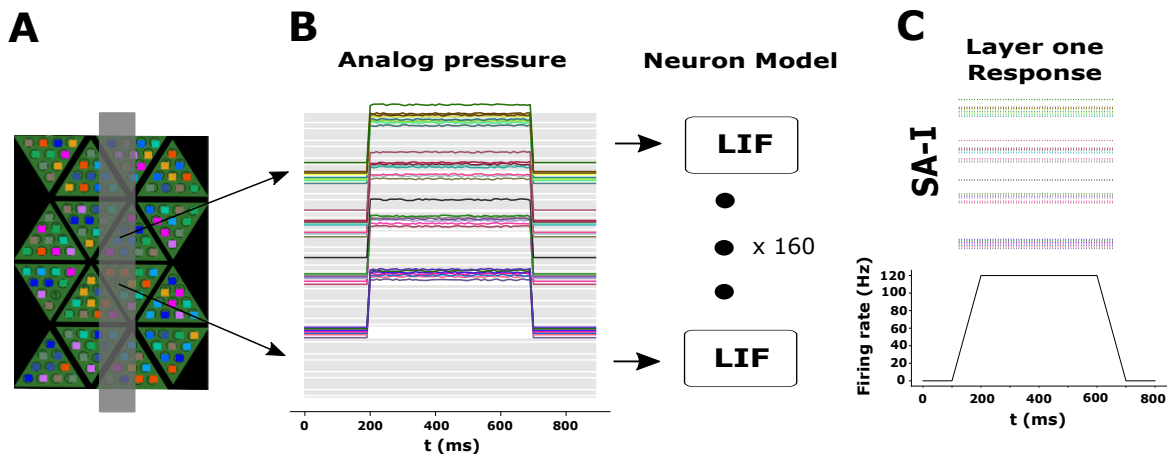


Figure 3.1 SA-I afferents: layer one encodes analog pressure in spike trains and consists of 160 LIF neurons; (A) An example of a vertical bar pressed on the skin patch that consists of 160 capacitive sensitive elements (Taxels) distributed over 16 triangles. (B) Analog output of the 160 taxels when pressed with the gray vertical (0°) oriented bar, where the gray shadow represents the non pressed taxels. (C) The output pressure of the taxels is used as input to LIF neurons; (top) Output spikes of neurons after pressing the skin with (0°) bar; (bottom) firing rate of one neuron.

of the stimulus gets compressed from high number of neurons into smaller elements. This allows efficient transmission across the acquisition path, with limited information loss. In this thesis, the second layer represents the receptive fields layer, in which each neuron in the second layer represents one receptive field.

3.2.3 Decoder Layer: Third-Layer Neurons

Johansson et al. claim the design of the somatosensory pathways could enable rapid classification of tactile stimuli by temporal-to-spatial conversion at the level of second-order neurons [3]. Based on the argument of rapid classification, they propose that cuneate neurons with a function of coincidence detection could be one possibility. Following this hypothesis, in this thesis, we modeled layer three neurons (decoder neurons) as coincidence detectors. The neurons in layer three are modeled using

LIF model, in which the input to layer three neurons was given by the summation of coincidence activation of the previous layer, such that the differential equations determining the evolution of their dynamics is:

$$\frac{dv_{3_i}}{dt} = \frac{v_{rest} - v_{3_i}}{\tau_3} + \frac{I_{ex}(t) + I_{inh}(t)}{C_{m_3}}$$

In this equation, v_{3_i} represents the membrane potential of layer three neurons, τ_{m_3} is the membrane time constant and C_{m_3} is the membrane capacitance. I_{inh} represents the inhibitory current arriving from the global inhibitory neuron, and I_{ex} is the current supplied from layer two neurons which defined as:

$$I_{ex}(t) = \sum_{j=1}^n \left(w_{3_{ji}} \sum_k S_{2_j}(t - t_k) \right)$$

S_{2_j} represents the spatio-temporal output spikes of the j_{th} neuron in layer two. t_k is the time in which the neurons in layer two fire a spike.

Therefore, the coincident activation of several layer two neurons signals the presence of a specific stimulus pressed on the simulated skin. This coincidence is decoded using the layer three neurons. Specifically, the stimulus is decoded based on the temporal coincidence activation of layer two neurons using competitive neurons, controlled by a global inhibition WTA mechanism [136]. The layer three neuron that spikes with the highest activity is then defined as the winner, while the others are considered not-significant.

3.2.4 Winner-Take-All

The winner-take-all (WTA) computation is an intrinsic property of recurrent networks, which abound in cortex. In this thesis two kinds of WTA competition were studied and implemented. The first kind, synapses with hard-wired connectivity are used to realize a competitive WTA network with layer three neurons. The WTA structure, composed by a global inhibitory neuron: all neurons of the third layer sends their excitation through excitatory synapses on the global inhibitory neuron; the inhibitory neuron, in turn, stimulates the inhibitory synapses of the all excitatory neurons of the third layer, such that only the neuron receiving the highest input can be active [136]. The second kind, the WTA structure, composed of fast lateral inhibitory layer consist of inhibitory neurons. Every neuron in the decoder layer is connected to one neuron in the inhibitory layer, where it receives excitation from the neuron in the decoder layer and in turn inhibits all the other neurons. Through lateral inhibitory connections, a firing neuron tends to either prevent the other neurons from firing or reduce their firing rate [137].

3.3 Neuromorphic Architecture for Edge Orientation Selectivity

3.3.1 Sensors and Dataset

In this study, the system is composed of a skin patch from the iCub robot [138], a ZynQ7020 board, and a laptop as shown in figure 3.2.

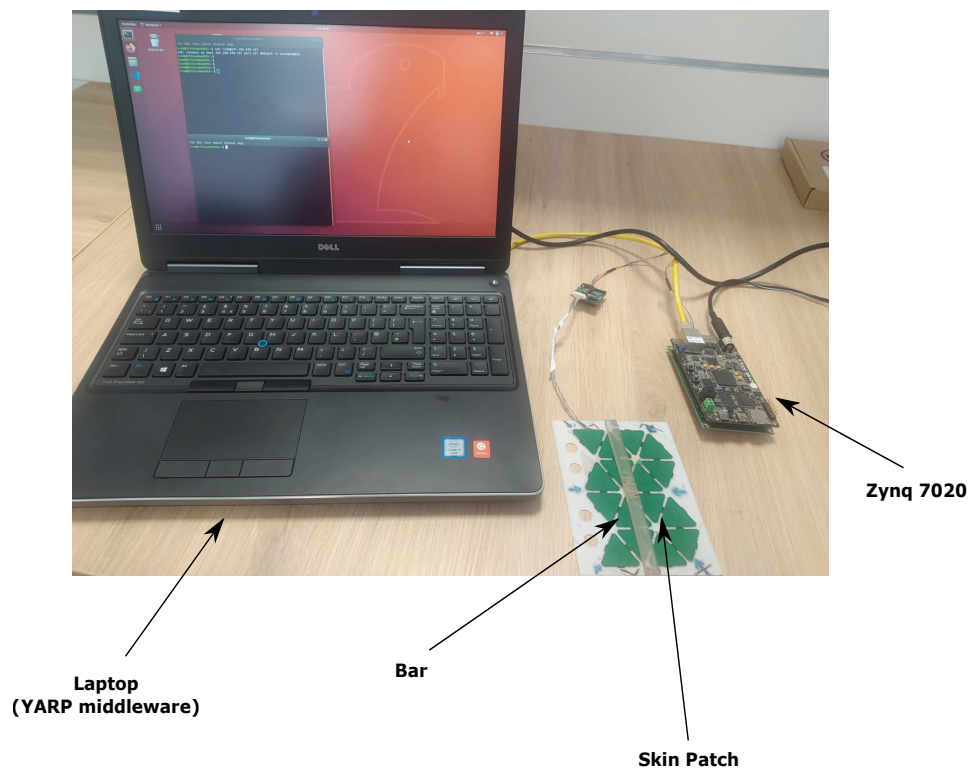


Figure 3.2 System setup for edge orientation processing

The skin patch, with dimensions (11×7.5 cm), comprises 160 capacitive tactile sensing elements (taxels) distributed along 16 triangles (10 taxels each) as shown in figure 3.3.

During the contact, the capacitance value is acquired and digitised by an off-the-shelf ADC7147 and sent via the I2C bus to the Xilinx ZynQ 7020 device. The Xilinx ZynQ 7020 device uses programmable logic to acquire, deserialise, and associate a timestamp to the samples (samples refer to the clock-based sampling of the capacitive value). The data stream is then written to memory through DMA and made available to any process for computation. In our setup, this is realised by a software grabber module that reads data from DMA and sends to Yet Another Robot Platform (YARP) middleware [139]

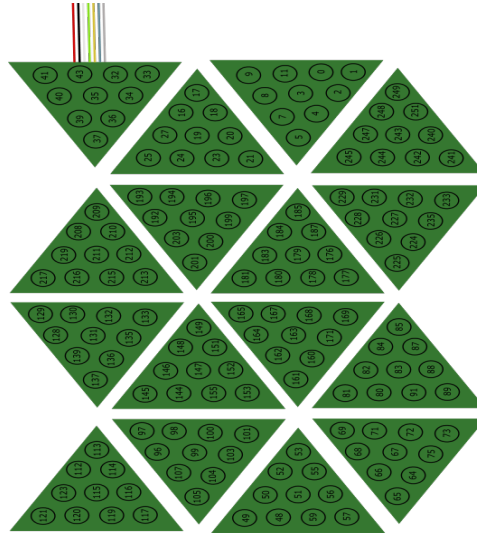


Figure 3.3 Skin patch from iCub robot.

through Ethernet, where the stream of samples is available for high level software processing.

The dataset collected for this work was obtained by manually pressing the bar on the iCub's skin at 8 different orientations (0° , $\pm 22.5^\circ$, $\pm 45^\circ$, $\pm 67.5^\circ$, 90°). We augmented the dataset, increasing the number of possible orientations, by mean of an ad-hoc Python library that computes the superposition of the bar with the taxels for a given orientation and outputs the corresponding measured activation of the taxels. The user can generate all kind of stimuli, changing bar width, length and orientation. Because a simulated press happened, a different approach has been followed: given that the real dataset didn't have any information about the relation between the sensors output and the pressure applied, we assumed in the network that the relation between the pressure of the bar and the sensor's response is linear, applying at the latter a multiplicative coefficient. In this study, the minimum angle variation considered was 5° , with a maximum angle of 180° .

3.3.2 Network Architecture

Inspired by the finding of [19], in this study, we present a neuromorphic model for edge orientation selectivity that relies on event-driven acquisition and unsupervised spike-driven learning, based on computational primitives that are implementable on low-power mixed-mode sub-threshold neuromorphic hardware. In this study, we used the capacitive skin of the iCub robot [138] as front-end sensors, connected to a three-layers network of LIF neurons, that gives rise to edge orientation selectivity, as shown in figure 3.4. Layer one consists of 160 LIF neurons that converts the analog values produced by the pressure-sensitive capacitors into spike trains. The analog pressure value is used as a current injected in the neuron's membrane whereby the output

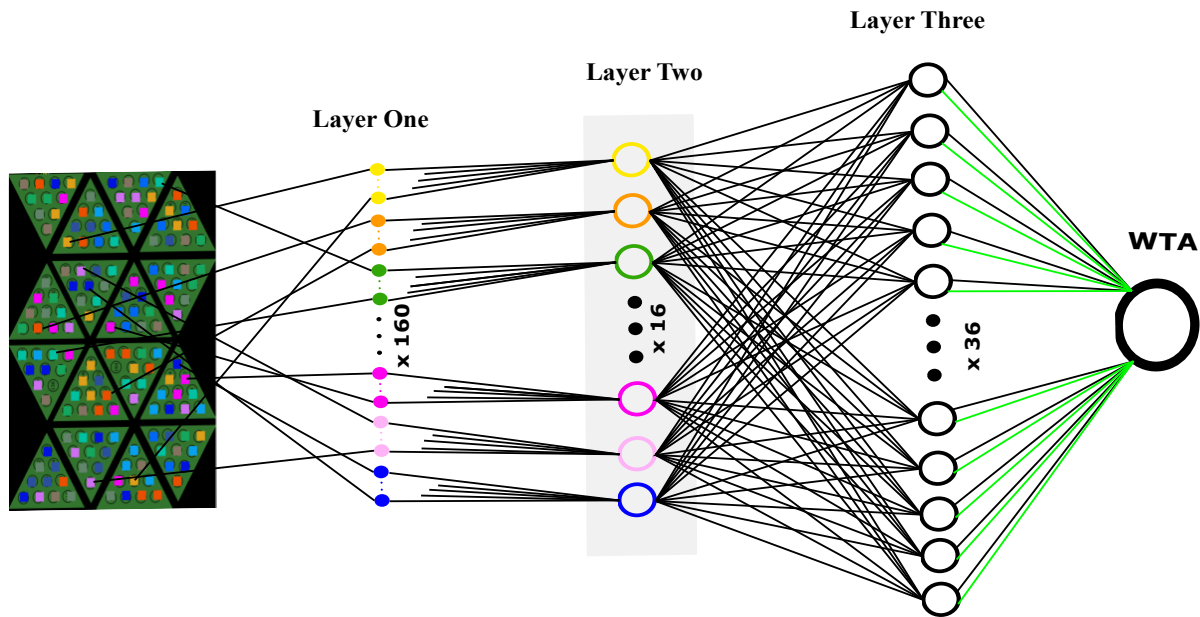


Figure 3.4 Spiking neural network for edge orientation selectivity; Network architecture comprising skin patch and three layers of LIF neurons. Layer one neurons convert the analog capacity value into spike trains (SA-I afferents); Layer two gathers input from multiple layer one neurons and having spatially distributed, overlapping, receptive fields; Layer three neurons receive input from layer two, each neuron of layer three is selective to a specific input orientation. Recurrent connectivity by neurons of a global inhibitory neuron ensures that a single layer three neuron is active, implementing a form of WTA competition.

firing of the neurons is proportional to the pressure, mimicking the sustained response of SA-I types of mechanoreceptors. Layer two consists of 16 neurons that integrate the activity of multiple mechanoreceptors from layer one. As it is possible to obtain higher orientation acuity when using receptive fields with random (and interleaved) sensitive points rather than when they are positioned uniformly [108], we implemented the structure of receptive fields based on a matrix of connectivity between layer one neurons and layer two neurons generated randomly, such that every neuron in layer one is connected to only one neuron in layer two, as shown in figure 3.5. In order to design equally likely connectivity patterns between layer one and layer two, we generated the matrix with two constraints: (1) every 10 neurons in layer one are connected to one neuron in layer two and (2) each neuron in layer one is connected only to one neuron in layer two. Layer three consists of 36 LIF neurons (each for one orientation) and decodes different orientations using the temporal coincidence activation of layer two neurons. For better selectivity, layer three employs WTA structure, composed of a global inhibitory neuron.

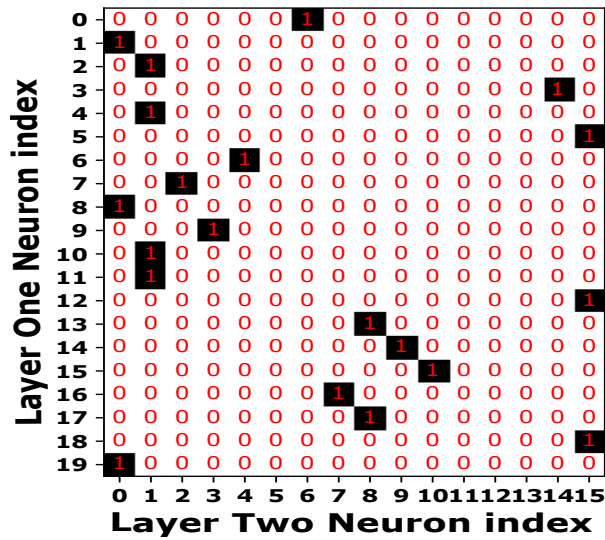


Figure 3.5 Randomly generated connectivity matrix between layer one and layer two (first 20 neurons on layer one); each neuron in layer one (rows) is connected to only one neuron in layer two (cols).

3.4 Neuromorphic Architecture for Touch Modalities Classification

3.4.1 Sensors and Dataset

In this study, the tactile sensing system is composed of a piezoresistive tactile sensor array, an electronic interface, and a graphical user interface. The tactile sensor array is based on the Force Sensing Resistor (FSR) MS9723 composed of a 16×10 piezoresistive sensor. It converts the applied pressure into electric resistance variation. When the force applied on the sensor increases, the resistance decreases inducing an increase in the output current. The tactile array has 10 columns and 16 rows, making a total of 160 force sensing nodes (8 mm X 8 mm).

The electronic interface is based on the snowboard system and is a resistive version of modern capacitive multi-touch sensing technology. The Snowboard is developed by Kitronyx and can be connected to any resistive matrix sensor. When connecting the MS9723 to snowboard, all forces applied to each node can be detected. The snowboard has a USB interface facilitating the communication with the computer.

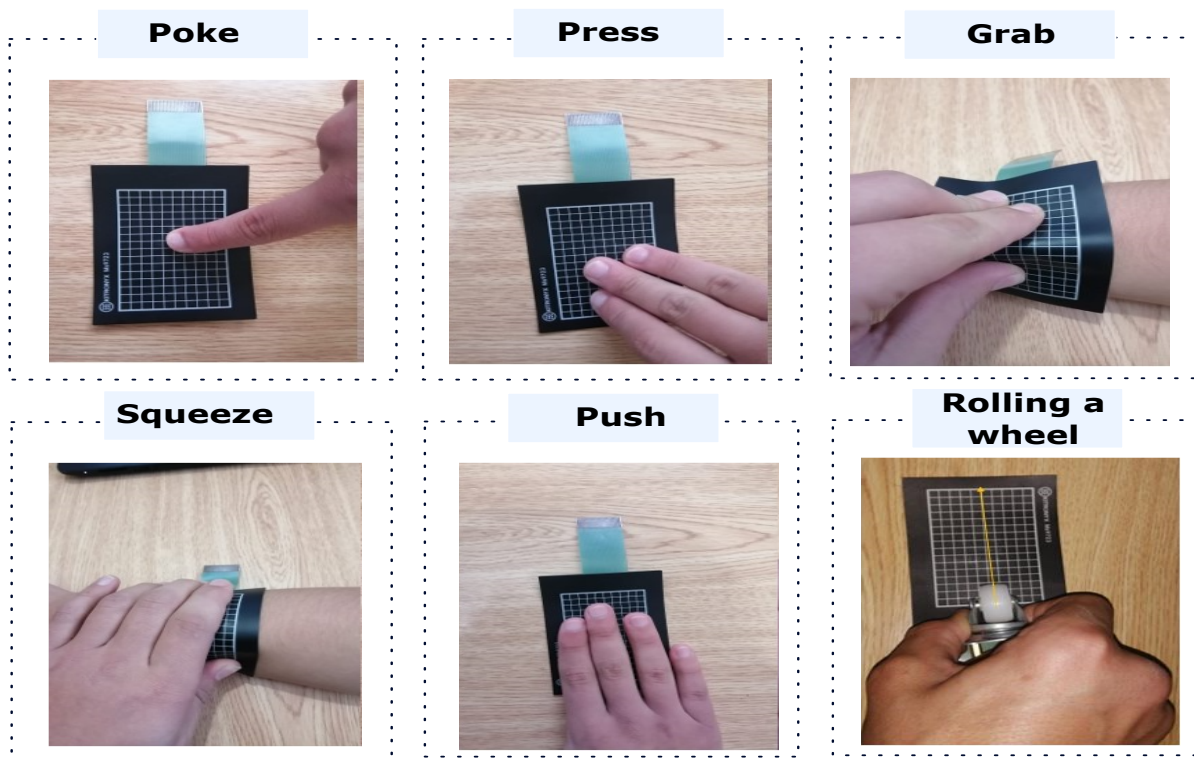


Figure 3.6 Six different touch modalities.

The graphical user interface (GUI) is based on the Snowforce 3 software to visualize 2D force map data and data logging. This software is specially designed to work with Kitronyx data acquisition devices including the snowboard. This GUI has been used to visualize and test the system, and to perform data acquisition for the touch modality dataset.

The dataset collected for this study is composed of six different touch modalities: “poke, press, grab, squeeze, push, and rolling a wheel” as shown in figure 3.6. It has been collected from three male participants with average age of 20-year-old. Each participant has been asked to apply the touch on the sensor array forming the modality with an interval of time between 1 and 3 seconds. The participants repeated each modality many times to form a dataset of size of 40X6.

3.4.2 Network Architecture

In Sec. 3.3.2 the neural architecture that gives rise to edge orientation selectivity is based on a network composed of three layers of LIF neurons. In this study, we modified the neural architecture to be able to classify different touch modalities. The network for touch modalities classification shares the same structure of the network architecture proposed for edge orientation selectivity, which include an input layer, hidden layer, and output layer as shown in figure 3.7. The input layer consists of 160 LIF neurons responsible for the conversion of analog pressure into neuromorphic spikes. When human interacts with the sensor array that consists of 160 piezoresistive sensors mentioned in the previous section (Sec. 3.4.1) with different touch modalities, these sensors produce an output analog signal that defined the activation of each sensor as a function of time. The hidden layer shares the same role and number of neurons of the receptive field layer in Sec. 3.3.2, as well as the connectivity with the previous layer. The output layer consists of 6 LIF neurons (one neuron for each modality) that decode different modalities by receiving input spikes from the previous layer and using the

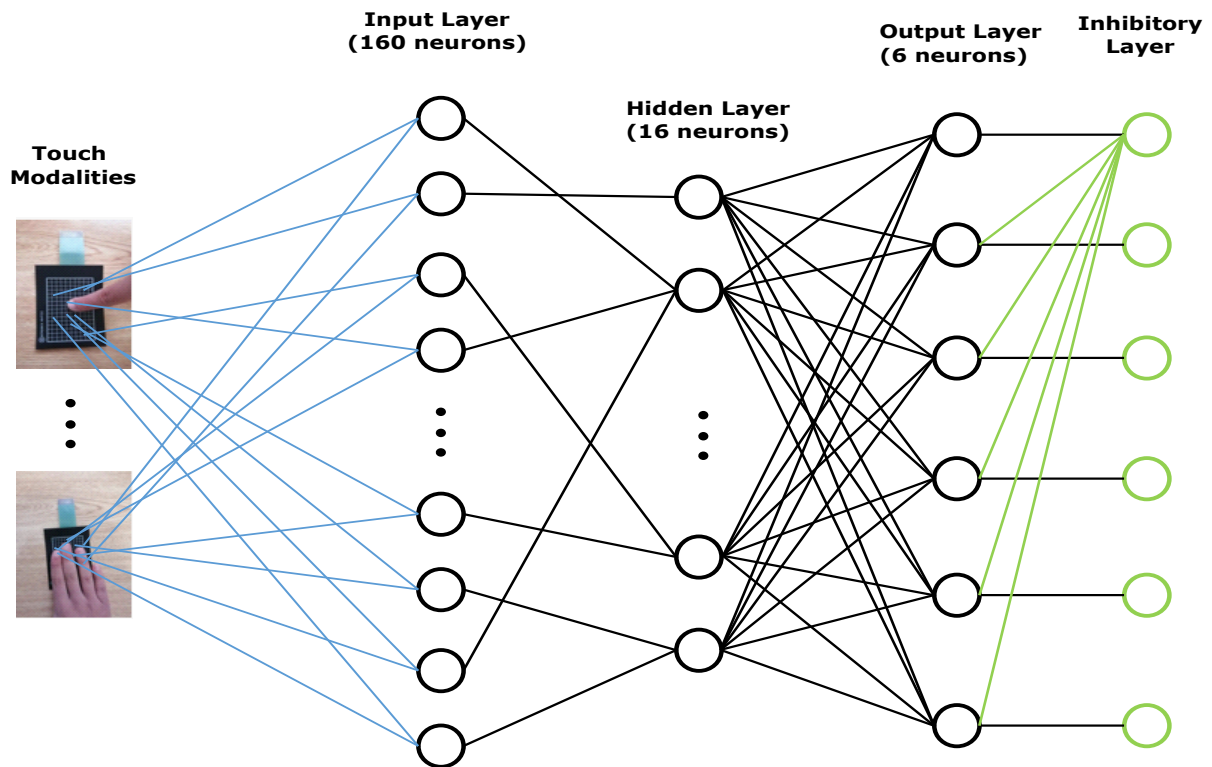


Figure 3.7 Spiking Neural Network for touch modalities classification; (Left) example of two from six different touch modalities applied on a sensor array of 160 piezo-resistive sensors. (Middle) 3 layers of LIF neurons; input layer consists of 160 neurons, hidden layer gathering the 160 inputs from input layer, and output layer consists of 6 neurons which is equal to the number of classes to be classified (touch modalities). (Right) lateral inhibitory layer that consists of 6 inhibitory neurons to increase the accuracy performance of the network.

temporal activation of hidden layer neurons. Neurons of hidden layer and output layer are connected all-to-all through weighted synapses (W). To achieve better selectivity, and also to increase the classification accuracy of the network during the testing phase, a fast lateral inhibitory layer composed of 6 inhibitory neurons has been added to the network. Every neuron in the output layer is connected to one neuron in inhibitory layer, where it receives excitation from the neuron in the output layer (black synapses between output layer and inhibitory layer in figure 3.7) and in turn inhibits all the other neurons (green synapses in figure 3.7).

3.5 Neuromorphic Architecture for Tactile Object Shape Classification

3.5.1 Sensors and Dataset

In this study, the tactile sensory system is similar to the one used in the touch modality classification experiment (Sec. 3.4.1).

The dataset collected for this study is composed of eleven different object shapes which are: Bottle Cap, Eraser, Gas Lashes, Highlighter Cap, Key, Marble, Rock, Shaped Screw Driver, Spray Cover, Tape, and Wood. Each object has been applied on the artificial skin for multiple trials, in which each trial was saved in a separate dataset. The total number of the collected datasets is equal to 440 where it was divided into 80% for learning and 20% for testing.

3.5.2 Network Architecture

In this study, we adopted and modified the feed-forward spiking neural network shown in Sec. 3.3.2 to be able to discriminate between different object shapes. The SNN comprises two layers of spiking neurons that represents the input (layer one) and output (layer two) layers attached to an artificial skin patch consisting of 160 piezo-resistive tactile sensors as shown in figure 3.8. Layer one represents the encoding layer and consists of 160 neurons, in which each neuron in layer one is connected to one sensor in the sensory array as one to one connection. The output analog signals

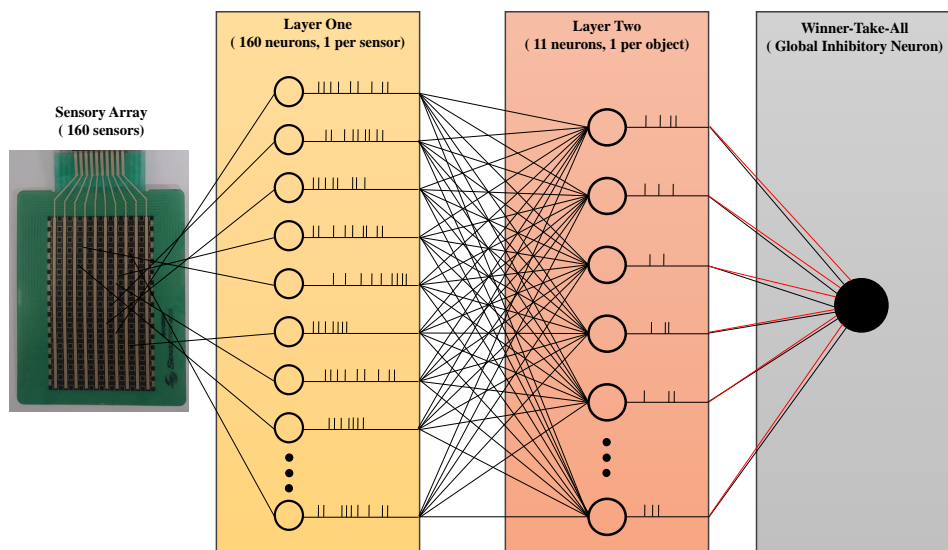


Figure 3.8 Spiking Neural Network for object shapes classification. (Left) sensory array of 160 piezo-resistive sensors. (Middle) 2 layers of LIF neurons. layer one consists of 160 neurons and layer two consists of 11 neurons (one per object shape), in which the connection between the two layer were all to all connection through weighted synapses. (Right) WTA structure composed by global inhibitory neuron.

produced by the activated sensors are used as input current to layer one neurons. Layer two (classifier layer) consists of 11 neurons that decode the different object shapes presented to the network by detecting the temporal coincidence between activated neurons in layer one. The proposed network with only two layers SNN could be considered as one of the simplest networks for solving classification problems since it only relies on the input and output layers. Moreover, replacing the used sensors with event driven sensors may dramatically decrease the complexity of the network since these sensors could fed directly the output layer.

3.6 Discussion

In this chapter, we developed a biologically spiking network composed of three layers of LIF neurons to discriminate different edge orientations of a bar, as well as different touch modalities applied on the artificial skin. Two different sensory arrays were used in this chapter, the skin patch from the iCub robot for edge orientation discrimination and the FSR sensory array for touch modalities classification. These two neuromorphic architectures were developed based on competitive primitives (e.g LIF neuron model, synapses, and WTA) that can be implemented on low power mixed mode sub-threshold neuromorphic hardware. The network is capable of analysing the analog pressure values arriving from the sensory array upon contact with stimulus and convert them into neuromorphic spikes by means of layer one neurons. Such that, the neurons output firing rate is proportional to the pressure, mimicking the sustained response of SA-I types of mechanoreceptor. In addition, the activity of each group of multiple mechanoreceptors converges to one receptive fields that sends its excitatory current to the decoder layer that decodes the input stimulus based on temporal coincidence activation of those receptive fields.

For a different application scenario in which we aim to discriminate between different object shapes, we adopted and modified the spiking neural network. The network for object shapes recognition is composed only of two layers (input and output layer). In this way we decreased the complexity of the network as well as the computational cost.

Chapter 4

Receptive Fields

4.1 Introduction

Visual and tactile sensory processing both involve neural mechanisms that extract high-level geometric features of a stimulus, such as the orientation of an edge, by integrating information from many low-level inputs [111, 140–143]. Although geometric feature extraction is generally attributed to neural processing in the cerebral cortex [144, 145], there is growing evidence in the visual system that feature extraction begins very early in the processing pathway [146], even at the level of first-order (that is, bipolar) neurons in the retina [147]. Authors in [19] demonstrated that feature extraction also begins very early in the tactile processing pathway, at the distal arborization of first-order tactile neurons. First-order neurons in the tactile system have distal axons that branch in the skin and form many transduction sites, yielding complex receptive fields with many highly sensitive zones [148, 149].

Despite the concept of receptive field is commonly present in nervous systems (like visual receptive fields [150]), a common agreement on tactile receptive fields is still missing. Previous observations that first-order tactile neuron's highly sensitive zones are non-uniformly distributed within its receptive field [109, 151] motivated two key predictions. First, the intensity of a neuron's response would signal edge orientation because its firing rate would increase with the degree of spatial coincidence between the neuron's highly sensitive zones and local tissue deformations caused by an edge moving across the skin [152, 151]. That is, for a given neuron, some edge orientations show more spatial coincidence than others, and therefore yield stronger responses. Second, the temporal structure of a neuron's response would signal the orientation of an edge moving across its receptive field. That is, the temporal structure of the evoked action potentials is defined by the sequential stimulation of the neuron's highly sensitive zones, which in turn depends on edge orientation. These two predictions were confirmed in the study conducted by authors in [19], where they presented a model of receptive fields in which for each edge orientation subset of neurons activated at the same time, and studying the spatial coincidence between neurons.

Moreover, some early studies suggested that the tactile receptive fields are similar to the visual ones but shaped differently due to the different stimuli they interface with: vision seems to be handling more natural scenes with several objects and a heterogeneous background, while touch is mainly related to surface textures and regular repetitions. Touch stimuli seem in general sparser than visual stimuli [153]. Some other more recent studies [154, 108] suggest that this sparsity greatly influences the mechanism

behind receptive fields in touch. The sparsity of the signal could have led, according to the studies, to a semi-random distribution of the receptive field. The authors also suggest that an explanation behind semi-random distributions in the receptive field comes from compressive sampling theory, in which high sampling accuracy can be obtained using random bases [108]. In [108] they modeled a virtual patch of skin with known biological constraints to show how, under a simple coincidence-coding scheme [155, 156], the presence of heterogeneous receptive fields with many subfields influences edge orientation resolution as a function of edge length and orientation. The virtual patch was innervated by synthetic units (i.e., first-order tactile neurons) with innervation density [4] and receptive field size [157] based on the known human physiology. Each unit's receptive field was actually composed of receptor elements. They compared two versions of the model. One where units had unique subfields by virtue of being connected to a random (2–64) number of receptors placed randomly in the units' nominally circular receptive field. And, as a comparison, another model with all units' having receptive fields with uniform sensitivity by virtue of being connected to one receptor element whose receptive zone corresponded to the unit's receptive field boundary. Results in [108] show that (1) the model with subfields performed at levels slightly better than our human participants – showing discrimination thresholds 1.3° for the infinite length edge to 13.1° for the 1 mm long edge and (2) the model with subfields always outperformed the model with a uniform receptive field.

In this thesis, within the edge orientation application scenario we conducted three different studies: (1) we analyzed the response of the model for edge orientation

selectivity to a manually designed set of receptive fields inspired by the finding of [19] (Sec. 4.2), (2) we investigated different receptive fields topologies with three different connectivity patterns (Sec. 4.3), and (3) we verified the hypothesis that receptive fields with random (and interleaved) sensitive points can offer higher orientation acuity than receptive fields where sensitive points are positioned uniformly [108] (Sec. 4.4).

4.2 Edge Orientation Discrimination with Designed Receptive Fields

Inspired by the finding of [19] we designed a manual structure of overlapping receptive fields in order to test the ability of our model architecture provided in chapter 3 Sec. 3.3.2 in discriminating between different edge orientations.

The dataset collected in this experiment consists of 8 different bar orientations manually pressed on the skin patch of the iCub robot (0° , $\pm 22.5^\circ$, $\pm 45^\circ$, $\pm 67.5^\circ$, 90°) using a bar of ($11\text{cm} \times 9\text{mm}$) as shown in figure 4.1-A. Moreover, five overlapping receptive fields were designed as follows: (1) for each of the 8 different bar orientations we chose three active sensors during pressing, in which these active sensors are connected to layer one neurons (colored mechanoreceptors in figure 4.1-A) as one-to-one connection, such that the total number of neurons in layer one was equal to the number of activated sensors for the whole bar orientations. (2) The three active neurons at each orientation are connected to three specific neurons in layer two (receptive fields) as one-to-one connections, causing them to fire as shown in figure 4.1-B, such that for each orientation

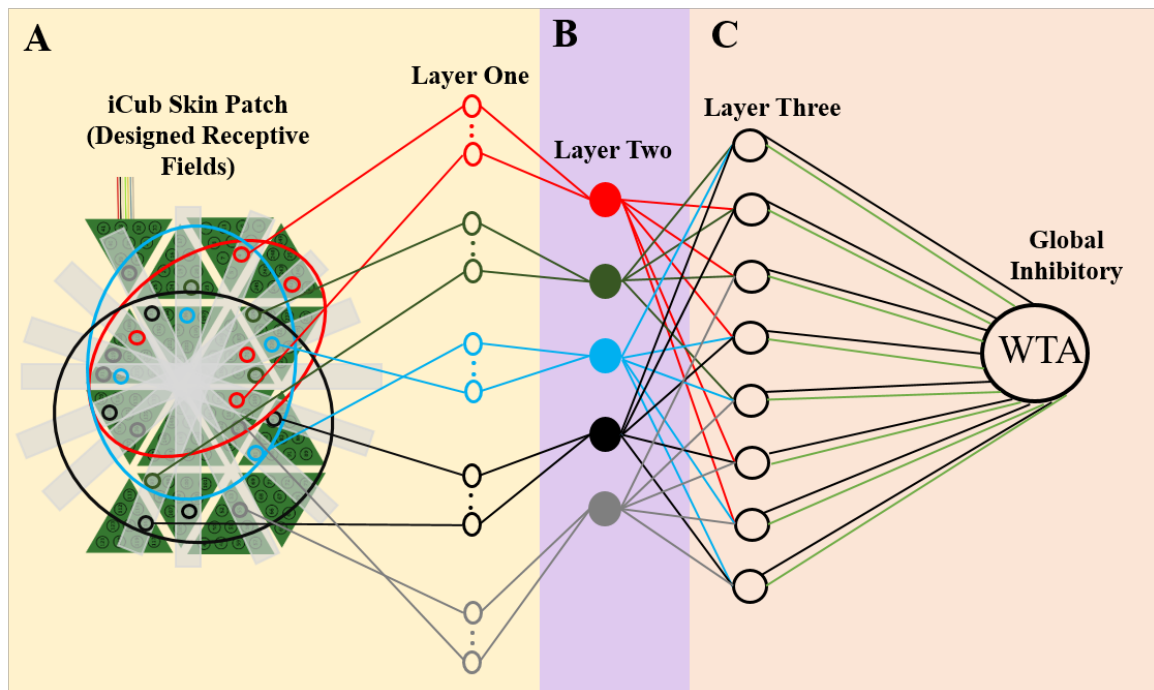


Figure 4.1 Spiking neural network for edge orientation with designed receptive fields: (A) Skin patch from the iCub robot and layer one, which encodes the analog signal into spikes. (B) First order tactile neurons (layer two) gathering the 160 inputs from layer one, organised in receptive fields. (C) Layer three discriminates edge orientation and includes a global inhibitory neuron that implements the WTA network.

three receptive fields are activated and firing simultaneously. The combination of the activated neurons in layer two at each orientation was connected to a single neuron in layer three as shown in figure 4.1-C. The coincident spiking activity of the three neurons of layer two increases the membrane potential of the neuron in layer three, causing it to spike. As a result, each neuron in the third layer is selective to a given orientation, increasing its firing rate for orientations close to its preferred orientation and showing maximum firing rate in response to the one for which it is tuned. Figure 4.2 represents the firing rate for the 8 neurons at layer three at each of the 8 different orientation (box per orientation in figure 4.2). For each of the different orientations,

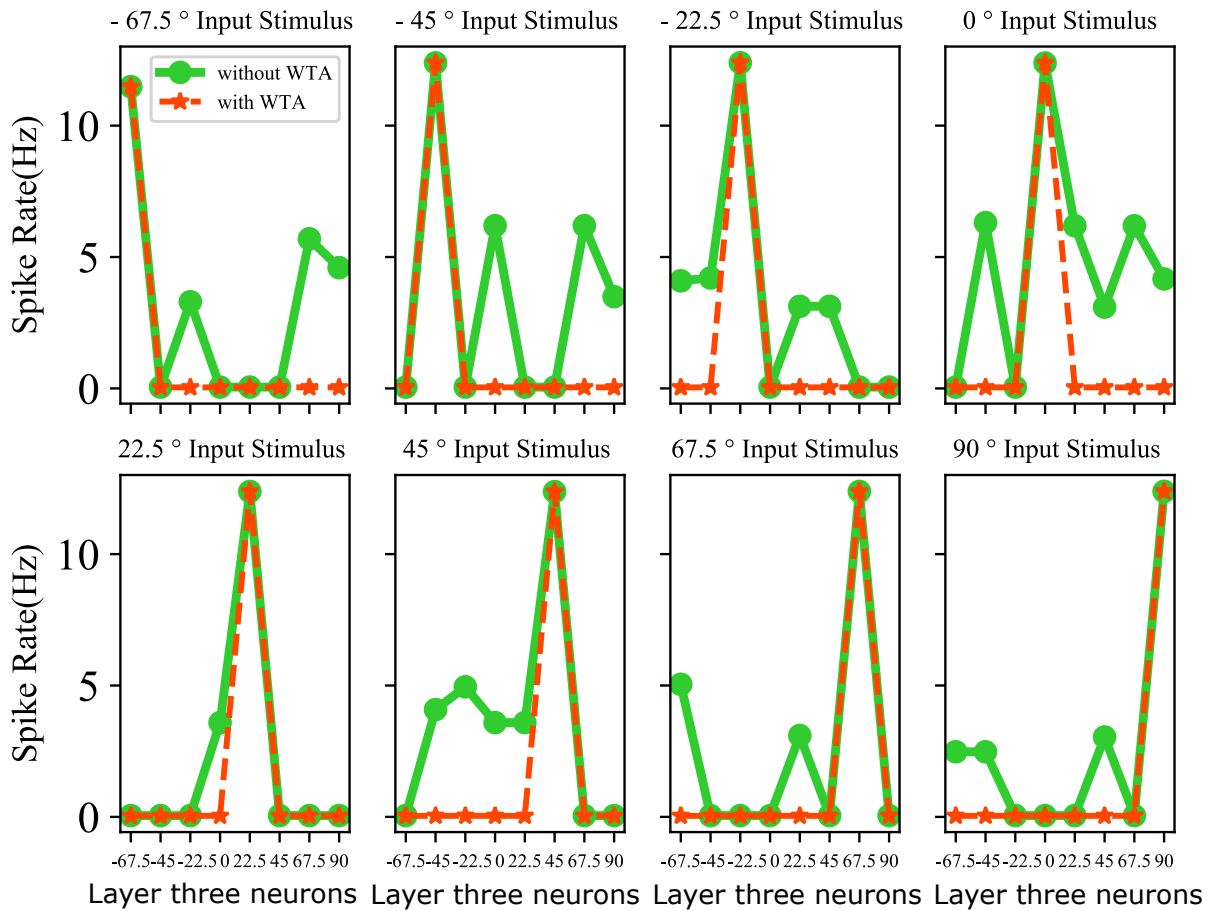


Figure 4.2 Firing rates of neurons in layer three for different stimulus orientations; with (red) and without (green) WTA.

there is only one neuron from layer three that has a maximum firing rate as shown in figure 4.2. Green curves in figure 4.2 show that some neurons fire several spikes also for orientations that do not have their exact combination of receptive fields but a similar one. To increase orientation selectivity, we implemented WTA competition using a global inhibitory neuron as shown in figure 4.1. Orange curves in figure 4.2 shows that with WTA only the winning neuron in layer three is firing and has a maximum firing rate.

Nevertheless, the network was able to detect eight different peaks through applying eight distinct input stimuli with each having a different orientation. This confirms that the model with manual structure of overlapping receptive fields can discriminate between several distinct edge orientations applied on artificial skin.

4.3 Receptive Field Structure

In section 4.2 we analyzed the response of the model to a manually designed set of receptive fields inspired by the finding of [19]. In this section, we aim to find the ability of our model to signal edge orientation with three different structures of receptive fields inspired by the findings of [108]. This gives the ability to increase the number of orientations to be detected by the model, as well as the randomization and generalization of the model proposed in chapter 3 Sec. 3.3.2 for edge orientation selectivity. In this study, The three structures share the same number of neurons in layer one (160 neurons which is equal to the number of sensors in the skin patch) and the number of receptive fields which is equal to 12 (12 neurons in layer two). The three different structures of receptive fields were implemented using a matrix of connectivity between layer one and layer two. The matrix composed of 160 rows in which each row represents one neuron in layer one, and 12 columns in which each column represents one receptive field (one neuron in layer two). Based on the two biological constraints we built the layer one layer two connectivity matrix W_1 . The first constraint is the non-negative regularization in W_1 that simulates the fact that first-order tactile neurons

can only be excited when their transduction sites are stimulated [154]. The second constraint is the convergence from layer one to layer two that simulates the many-to-one convergence from mechanoreceptors in the skin to first-order tactile neurons traveling in the nerve [154]. These constraints result in a matrix of zeros and ones (1 connected and 0 not connected), in which in each row there is single 1 which means that each neuron in layer one is connected only to one neuron in layer two. The three structures of receptive fields differ in the way of connecting the neurons in layer one to the neurons in layer two as mentioned in the following sections (Sec. 4.3.1, Sec. 4.3.3, and Sec. 4.3.2).

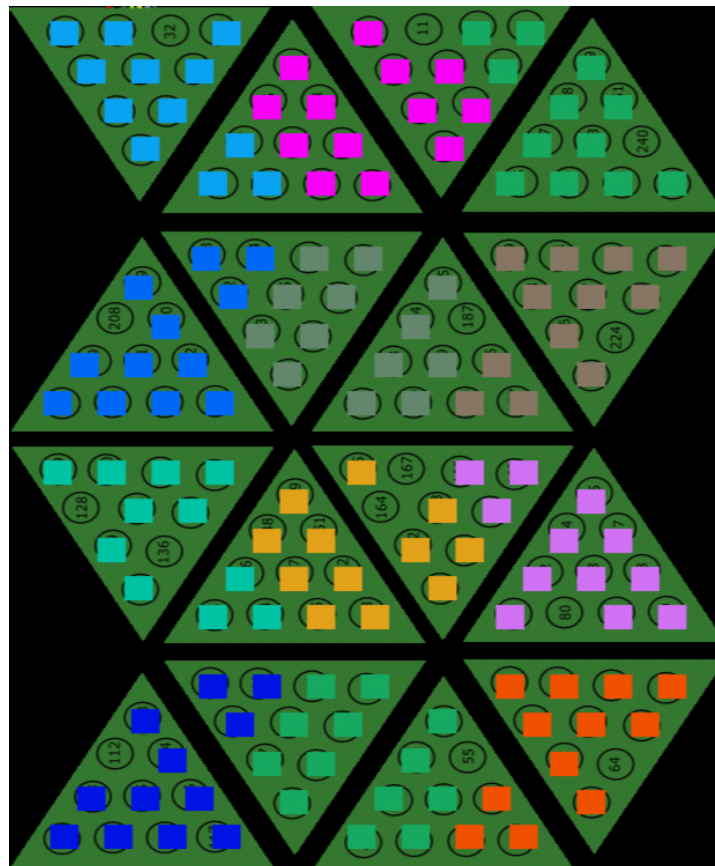


Figure 4.3 Uniform structure of receptive fields.

4.3.1 Uniform Receptive Fields

In the Uniform structure of receptive fields, the skin is divided in different and homogeneous receptive fields grouping the sensitive elements into regions. For example, we chose 12 different centers uniformly on the skin (one center per receptive field), then the 12 adjacent mechanoreceptors to these centers belong to a specific receptive field with constant area as shown in figure 4.3.

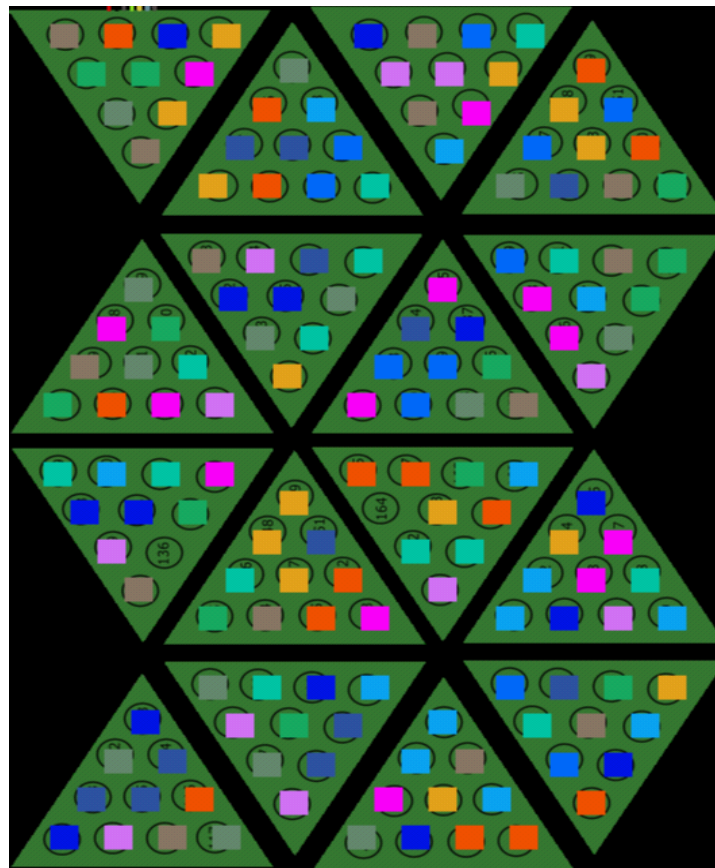


Figure 4.4 Random structure of receptive fields.

4.3.2 Random Receptive Fields

In the random structure of receptive fields, the sensitive elements on the whole skin are randomly associated to different receptive fields. For example, each receptive field consists of 12 mechanoreceptors connected randomly, the position of the mechanoreceptors changes at every simulation. This kind of connectivity forms randomly generated overlapping interleaved receptive fields. Figure 4.4 shows an example of random receptive fields structure on the skin patch of the iCub robot, where each color represents one receptive field (e.g. the dark blue small boxes represents the mechanoreceptors of the dark blue receptive field).

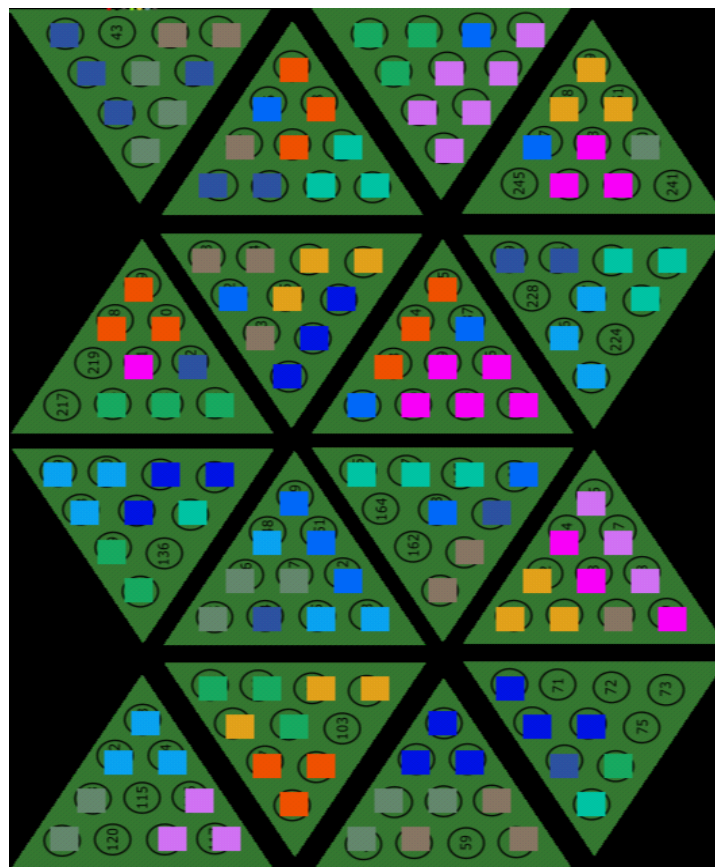


Figure 4.5 Random structure with subfields of receptive fields.

4.3.3 Random with Subfields Receptive Fields

In the random with subfields structure of receptive fields, the receptive field is composed of small clusters of adjacent sensing elements, such that for each receptive field, there are 4 highly sensitive points (taxel) randomly placed on the skin. The four highly sensitive points and two of their neighbors represent the mechanoreceptors of that receptive field as shown in figure 4.5. The random with subfields structure is different from the random structure where each taxel is independent of its neighbors.

4.4 Receptive Fields Assessments Using Mutual Information Theory

To verify the hypothesis that receptive fields with random (and interleaved) sensitive points can offer higher orientation acuity than receptive fields where sensitive points are positioned uniformly [108], we conducted three different experiments at the level of receptive fields layer (layer two) and we measured the performance acuity using mutual information theory. The three different experiments are as follow: (1) Receptive fields assessments as a function of connected sensors to each receptive fields (Sec. 4.4.2), (2) Receptive fields assessments as a function of edge length (Sec. 4.4.3), and (3) receptive fields assessments as a function of center position (Sec. 4.4.4).

4.4.1 Mutual Information Theory

When neurons respond to an input stimulus, they encode its characteristic in their spiking activity. This means that the input stimulus variables (e.g. like the pressure intensity or the angle of the bar) is converted into the output variable in the neuron (e.g. the spike count, the interspike interval or the time of the first spike). The value of the output variable is influenced up to a certain degree by the input variable. Estimating how much the output variable is expressing the input variable is the main aim of Information Theory [158].

Specifically, mutual information is the measure of the reduction of uncertainty that the output variable causes on the input one. The uncertainty (or entropy) can be expressed as:

$$H(X) = \sum_{x \in X} p(x) \log_2 \left(\frac{1}{p(x)} \right)$$

where $p(x)$ is the probability density function of the value x in the variable X . When the output variable Y is measured, the outcome can depend on the input variable X . This relation can be expressed as

$$H(X|Y) = \sum_{x \in X; y \in Y} p(x, y) \log_2 \left(\frac{1}{p(x, y)} \right)$$

where $p(x, y)$ represents the joint probability density function between the value x of X and y of Y . By computing how much uncertainty the output variable reduces with respect the input variable, the mutual information (or MI) can be estimated. The

resulting formula is

$$MI(X, Y) = H(X) - H(X|Y) = \sum_{x \in X; y \in Y} p(x, y) \log_2 \left(\frac{p(x, y)}{p(x)p(y)} \right)$$

This concept has been used in this work for assessments and to characterize the system at the receptive field layer (layer two). In the following three experiments, the activity of the receptive fields' layer encodes the orientation of the bar through a spatial code defined at each temporal window (the time during which a stimulus is presented). In said time, the neurons that have a spike activity higher than an arbitrary threshold (defined as half the spike rate of the most active neuron) are considered as '1' while the other ones as '0'. To assess the quality of the neural code generated by said layer we used mutual information. For each trial we counted the times a given spatial code appeared in relation to a bar's orientation. This results in a joint probability table of size $R \times S$, where R represents the spatial code responses and S the stimulus orientation. Therefore, using mutual information (MI), we computed how much information about the input orientation the system can encode.

4.4.2 Connected Sensors Study

We firstly recreated the experiment where a bar is pressed manually on the skin at eight different orientations ($0^\circ, \pm 22.5^\circ, \pm 45^\circ, \pm 67.5^\circ, 90^\circ$), but changing the configuration of the receptive fields according to the three proposed topologies (Uniform, Random, and Random with subfields).

Random Receptive Fields Structure

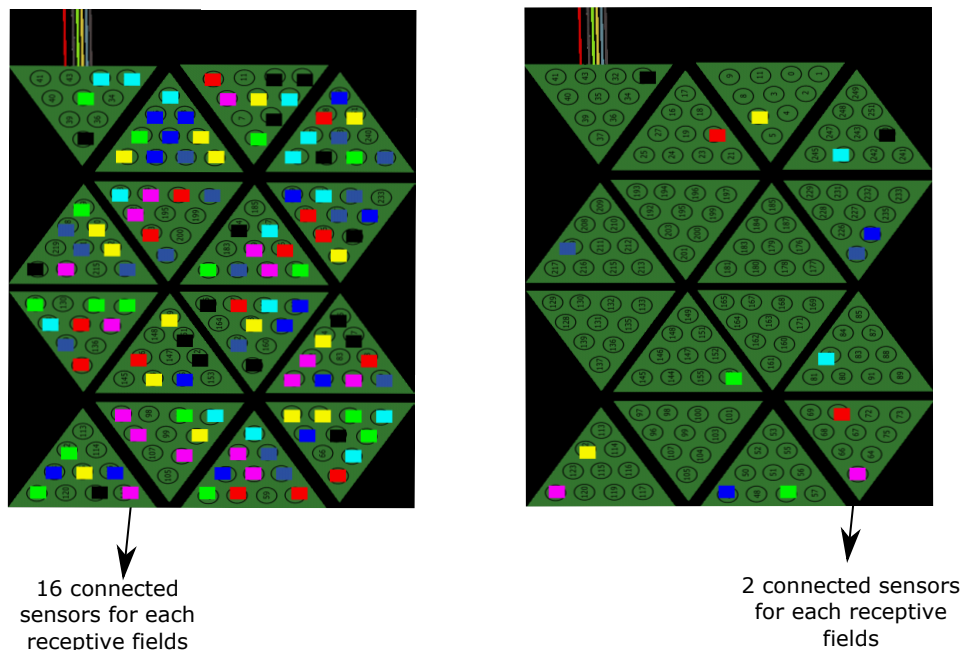


Figure 4.6 Example of the connected sensors study experiment with the random receptive fields structure; (left) 16 connected sensors to each receptive fields (each color represents the mechanoreceptors (sensors) of one receptive fields). (right) 2 connected sensors for each receptive fields.

This was repeated multiple times, while decreasing the number of taxels impinging to each receptive field. Starting with full connectivity of connected sensor (mechanoreceptors) to each receptive field, then decreasing the number of mechanoreceptors until we reach the minimum connectivity in which each receptive field comprising 2 mechanoreceptors. Figure 4.6 shows two examples of receptive fields structure with two different number of mechanoreceptors impinging to each receptive field (to the left, 16 connected sensors to each receptive field, where each color represents the mechanoreceptors (sensors) of one receptive field, and to the right 2 connected sensors for each receptive field with the random structure). Since the mechanoreceptors are randomly connected to the receptive fields in two of the structure mentioned in the

previous section, we repeated the simulation for 150 trials; in each trial the mechanoreceptors are randomized in different ways based on the matrix of connectivity. Figure 4.7 shows the mutual information computed for every different topology and for the number of taxels per receptive field. The taxels' number variation is meant to estimate the robustness of each topology to edge orientation encoding when the receptive fields density decreases. Moreover, as a function of connected sensors, the random structure of receptive fields outperformed the subfields and uniform structure.

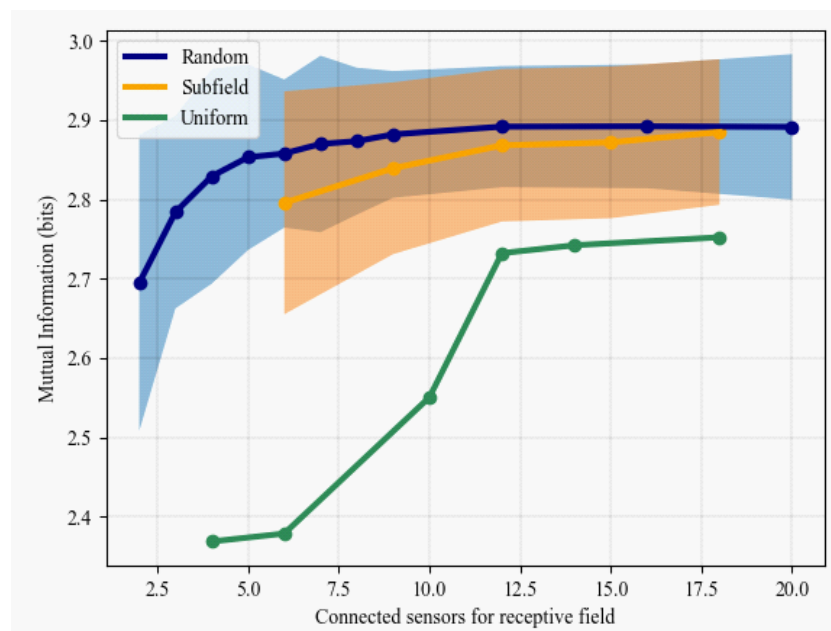


Figure 4.7 Mutual Information for each receptive fields' structure, as a function of connected sensing elements (mean and std shaded).

4.4.3 Edge Length Study

To calculate the minimum angle that the different topologies could discriminate, and to estimate robustness to the decreasing level of information about the stimulus, we

simulated an increased number of bars with different lengths and orientations. The way the taxels were distributed followed again the three different topologies.

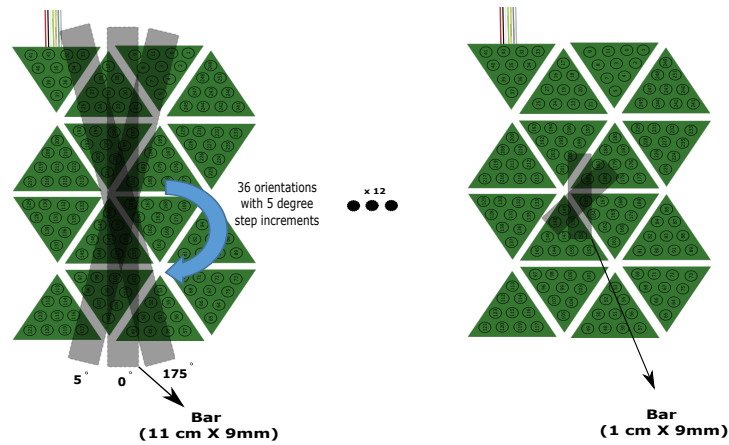


Figure 4.8 Edge length experiment. 36 orientation ranging from 0° to 180° with 5° step increments applied on the skin patch with changing bar from 11 cm to 1 cm with step 1 cm.

In this experiment, we applied 36 different angles ranging from 0° to 180° with 5° step increments with the length of the bar changing from 1cm to 11cm with a 1cm step as shown in figure 4.8. Each configuration was repeated for 150 trials. We measured the minimum angle detectable by the network dividing the maximum angle excursion and the number of detected orientations.

$$Angle(^{\circ}) = \frac{Maximum\ excursion}{Number\ of\ orientations} = \frac{180^{\circ}}{2^{MI}} \quad (4.1)$$

The results, visible in Figure 4.9, highlight that, given a fixed length, the receptive fields with random distribution seem to perform better in orientation acuity such that the receptive fields created by randomly selecting sensitive points perform better than structured receptive fields with uniform distribution in discriminating small angles

(down to 5°). As expected, the orientation discrimination, gracefully degrades with decreasing stimulus length.

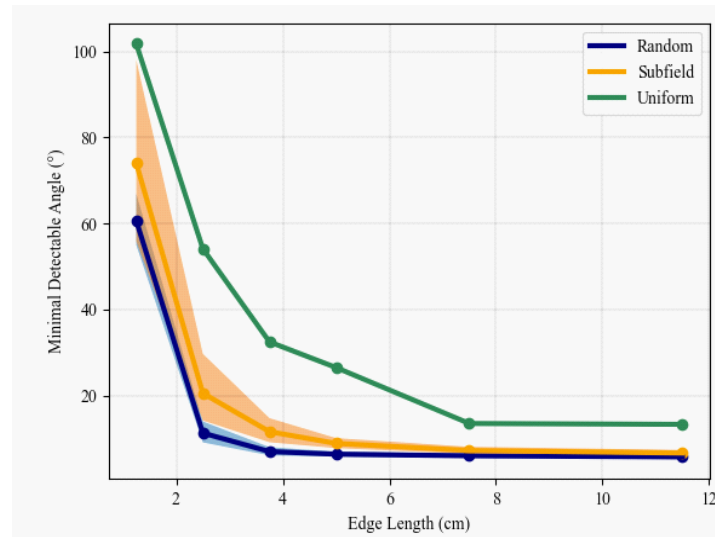


Figure 4.9 Minimum detectable angle as a function of bar length (mean and std shaded).

4.4.4 Center position Study

For generalization of the system proposed in this thesis for edge orientation selectivity, we conducted three different experiments as a function of center position of the pressed oriented bar. In the first experiment, we choose 4 different bar orientations (-45° , 0° , 45° , and 90°). Each oriented bar pressed on the skin patch with different position (e.g. the 90° bar pressed on the skin patch at 15 different position starting from the bottom of the skin and going upward by step 0.5 cm increments as shown in figure 4.10).

To illustrate, the total dataset collected for this experiment was composed of 4 different orientations with 15 different positions for each orientation (4×15). The way the taxels were distributed followed again the three different topologies. We used mutual

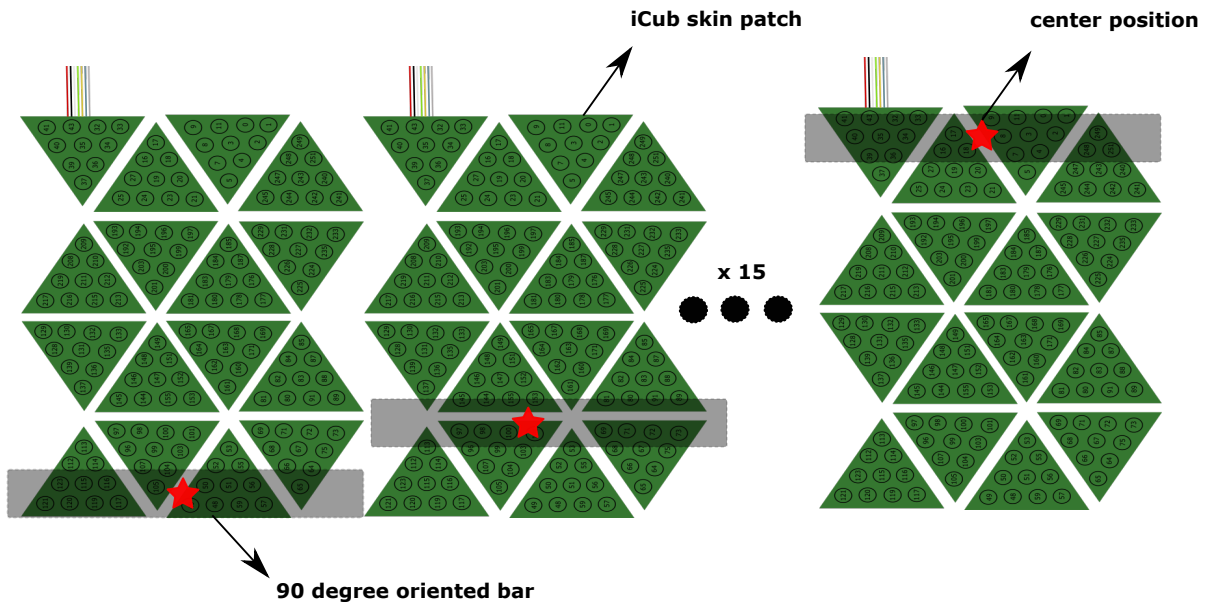


Figure 4.10 Center position study (first experiment); 90° oriented bar pressed on the skin with 15 different center position.

information in order to compute how much orientation the model encodes at each of the different center positions. The simulation was repeated 150 times. Figure 4.11 Shows the mutual information values as a function of 4 orientations. The number of centers detected by the network for each of the 4 different orientation calculated by the following equation

$$\text{number of centers} = 2^{MI} \quad (4.2)$$

(e.g., with 0° orientated bar and random structure the mutual information was equal to 4, which means that the system detected the 0° orientation at 2^4 center position as shown in figure 4.11). The results, visible in Figure 4.11, highlight that, with different centers positions the random and random with subfields structures looks similar in encoding the orientations and outperformed the uniform structure.

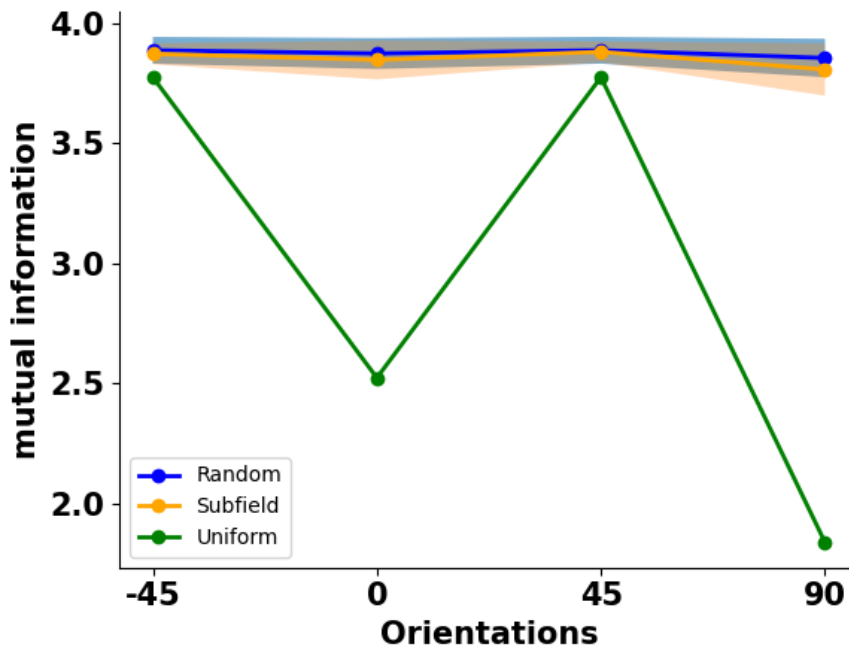


Figure 4.11 mutual information as a function of 4 different bar orientations for the three different receptive fields structures (uniform, random with subfields, and random); each orientation applied on the skin with 15 different center positions.

In the second experiment, we choose 12 centers position, at each position we simulate the 36 orientations ranging from (0 to 180 with step 5 increments) as shown in figure 4.12. As such, the total dataset collected for this experiment was equal to 12×36 .

At each position we calculated the mutual information (how much the system encodes different orientation at each of the different center positions at receptive fields layer) for the three different structures. The simulation was repeated for 150 times to calculate the mean and standard deviation. Figure 4.13 Shows the mean and standard deviation (shaded) mutual information as function of center position. The results, visible in Figure 4.13, highlight that, at the center of the skin patch, the system detected the highest number of orientations, whereas the number of orientations detected at the boundaries

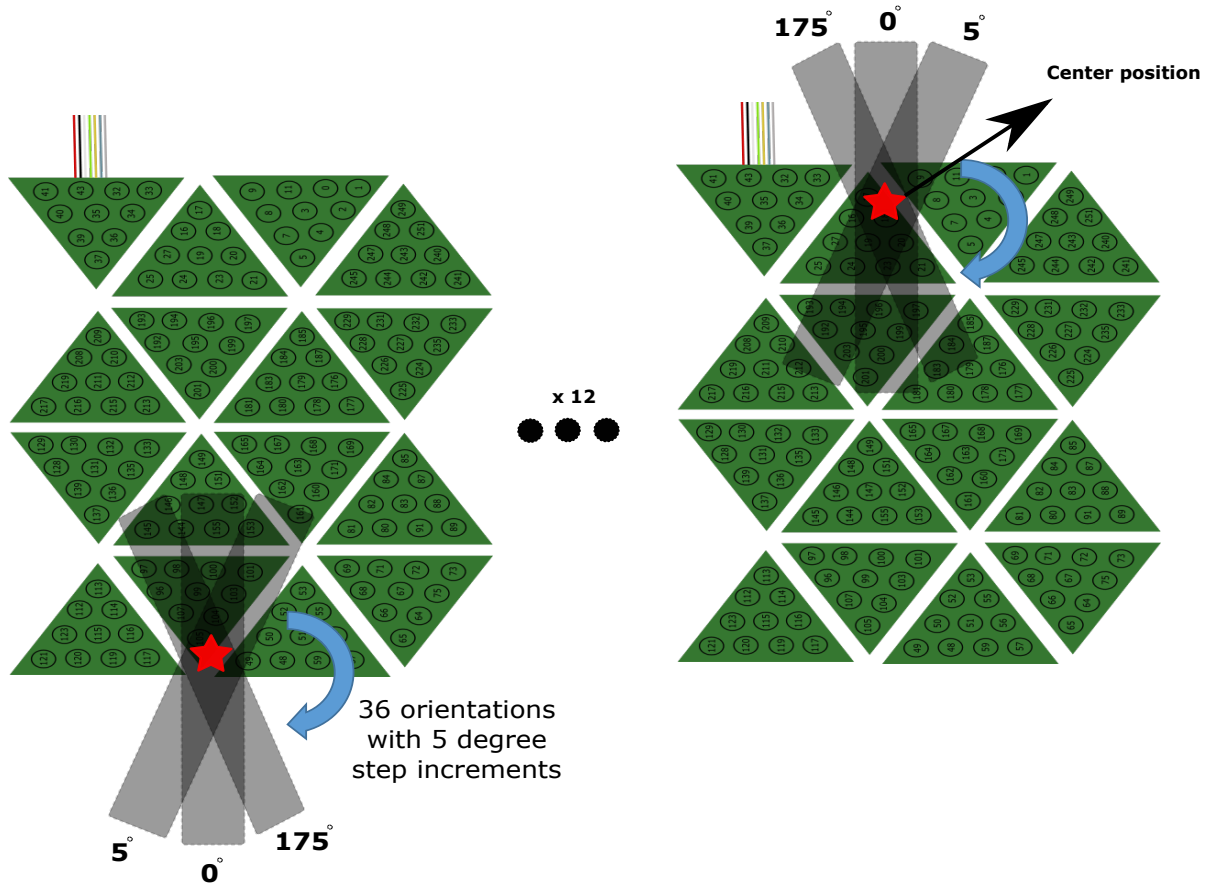


Figure 4.12 Center position study (second experiment); 36 orientations ranging from 0° to 180° with 5° step increments applied on the skin in 12 trials, in each trial we change the center position.

of the skin was the lowest. Moreover, as in the previous experiments the structure with random connectivity outperformed the random with subfields and uniform structure in detecting more orientations applied on the skin patch.

Finally in the last (third) experiments, we combined the center position and edge length in the same study, in which the experiments were as follow: (1) 5 different center positions, (2) 6 different bar lengths, and (3) 36 orientations. At each center position we applied 36 different angles (180° with 5° steps) with the length of the bar changing from 5cm to 11cm with a 1cm step as shown in figure 4.14.

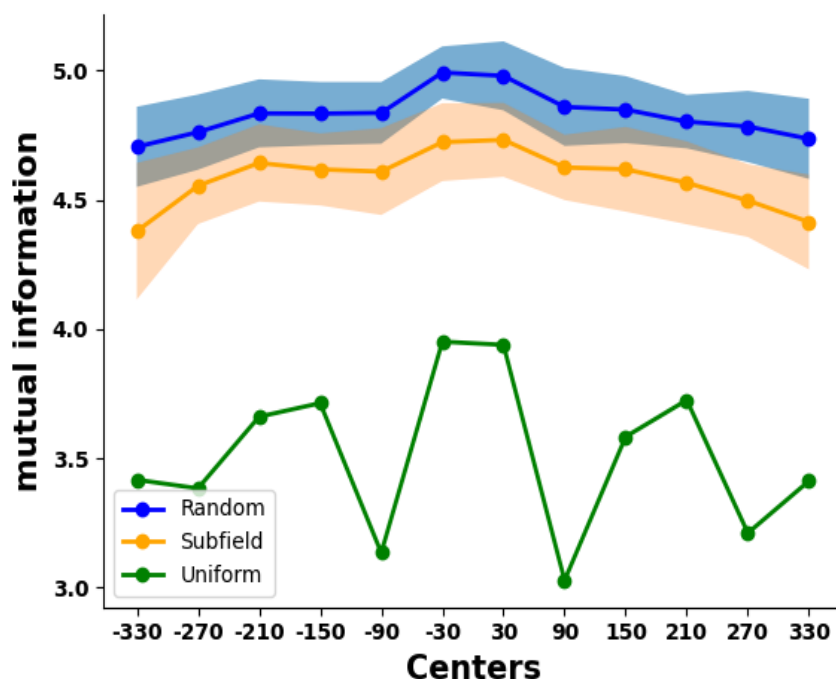


Figure 4.13 Mean mutual information and std (shaded) as a function of center position for the three different receptive fields structures (uniform, random with subfields, and random).

Each configuration was repeated for 150 trials. Figure 4.15 shows the mutual information as a function of center position and edge length for the random receptive field structure. The results, visible in Figure 4.15, highlight that, given a fixed length, the number of encoded orientation was better when the center position of the 36 orientation was at the center of the skin patch and with the longest edge bar, whereas the number of orientation decreased when the bar length decreased. Moreover, the orientation discrimination, decreases whenever the center of the applied orientations was far away from the center of the skin patch.

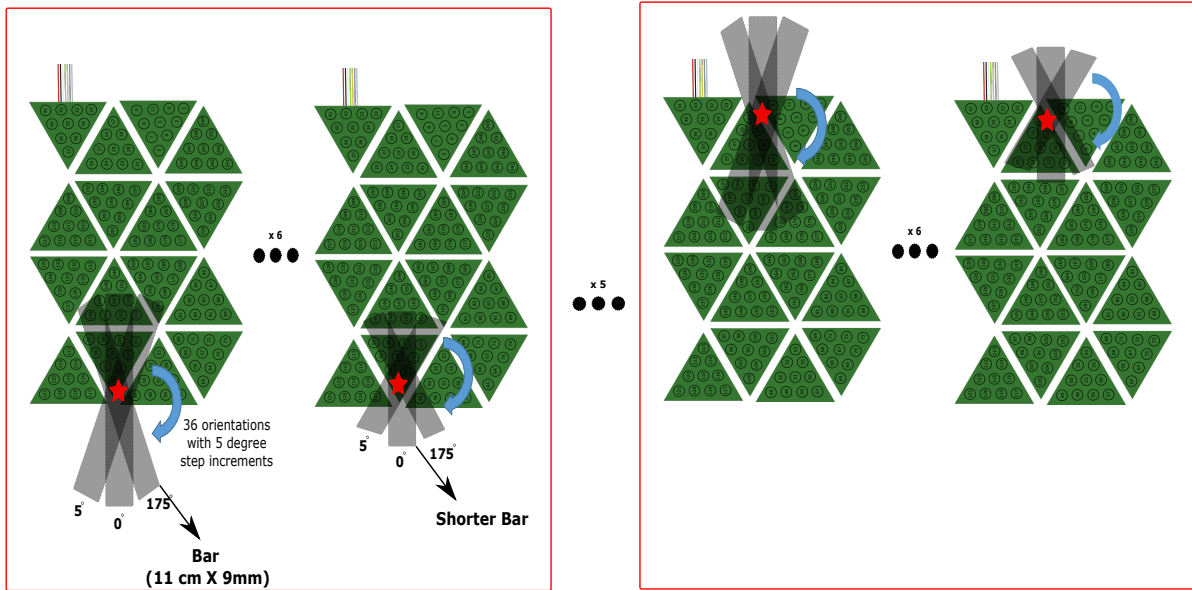


Figure 4.14 Center position study (third experiment); (left) 36 orientations ranging from 0° to 180° with 5° step increments applied with bar changing from 11 cm to 5 cm with step 1 cm. (Right) same as in the left part, whereas the center position of the 36 orientation is different.

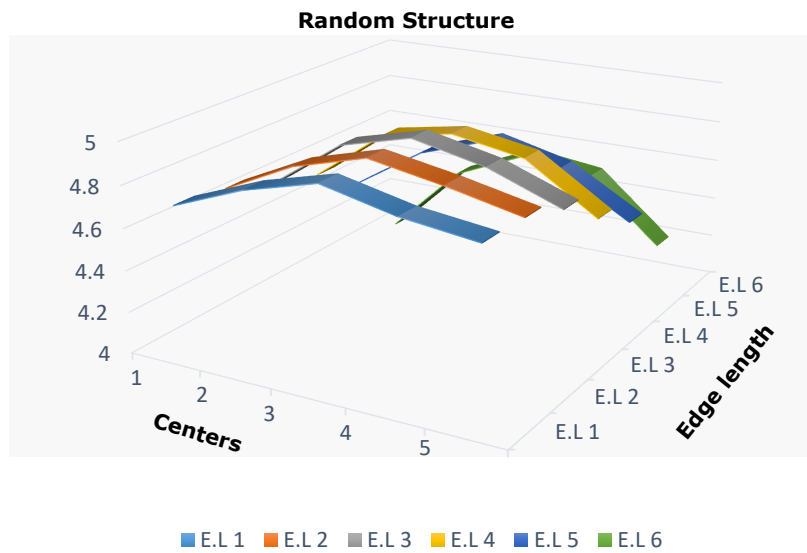


Figure 4.15 Mutual information as a function of edge length and center position for the random structure of the receptive fields.

4.5 Discussion

In the first part of this chapter the ability of the network architecture in discriminating different bar orientations pressed manually on the skin patch of iCub robot with manually designed overlapping receptive fields was studied. Thanks to the encoding strategy and the coincidence activation mechanism the network has the ability to discriminate between eight different orientations. Moreover, the WTA competition was introduced in this work, showing how when we used WTA composed of only global inhibitory neurons, the orientation selectivity can be increased. In the second part of this chapter, the response from several neurons connected to different sensors have then been collected into receptive fields to compress the information. The connection between layer one and layer two (i.e. the neurons connected to sensors and the neurons representing the receptive fields) has been studied to assess maximum information transmission. We verified the hypothesis that receptive fields with random (and interleaved) sensitive points can offer higher orientation acuity than receptive fields where sensitive points are positioned uniformly within three different studies (connected sensors, edge length, and center position). Therefore, through following biological examples [108] in which sensitive areas of the receptive fields are randomly distributed on the skin, we proposed an approach for estimating the optimal distribution of taxels in receptive fields for increasing the orientation acuity. This method can be used as a tool to minimise the connectivity and number of required taxels in future generations of artificial skin, while maintaining spatial sensitivity.

Chapter 5

Spike Based Learning with Synaptic Normalization

5.1 Introduction

Neurons in most vertebrate nervous systems communicate through action potentials. These are abrupt moments where the membrane voltage of a neuron rapidly rises and falls in a consistent trajectory. As the trajectory of the membrane voltage is independent of the amount of current that induced the response, an action potential or spike is typically considered an all or nothing event [159]. It is traditionally assumed that information is conveyed through the average spiking rate of a neuron [160]. However, recent studies have found precise temporal relationships in spike patterns evoked by sensory stimuli in visual, auditory, olfactory, and somatosensory pathways [161], suggesting that information is also encoded in the time of each spike. The

computational capabilities of individual neurons would be much higher than previously thought if temporal features are taken into consideration [162]. These key insights are the fundamental basis for the field of SNN. SNNs are synthetic networks of neurons that emulate the rich time-varying dynamics found in biology for increased realism [163, 164]. SNNs are shown to be superior to rate-based neural network models for complex tasks such as recognition of visual patterns, odors, sound properties, and tactile stimuli [165, 166]. Due to their biological credibility, SNNs are useful for modeling information processes in the brain and investigating various learning mechanisms [167, 168]. Engineering problems such as pattern recognition and real-time computation also benefit from the use of SNNs [169, 170]. The neurons in an SNN are linked together through synapses. Each higher-order neuron may receive inputs from multiple lower-order neurons through synapses of adjustable weights or efficacies. Like their biological counterparts, computational neurons models in SNNs communicate through spikes, which are discrete events in time. For SNNs with multiple layers, the ability for neurons to map specific spatiotemporal input spike patterns to temporally precise output spikes is very useful, as the spiking outputs can then be decoded by higher-order neurons sensitive to spike timing [171]. The temporally precise mapping can be achieved by adjusting the weights of the input synapses. However, the development of efficient algorithms for computing weights remains a challenge due to the complex nature of spike trains. For instance, elementary operations such as addition, subtraction, and multiplication cannot be directly performed on spike trains. Existing learning rules typically employ heuristic approaches which may produce

sub-optimal results. Hence, techniques to robustly optimize synaptic weights remain an important area of research for SNNs. Several techniques to train spiking neurons have been reported in the literature, in which they act directly on the weights of the synapses of two neurons connected with each other. Spike Timing Dependent Plasticity (STDP) and Spike Driven Synaptic Plasticity (SDSP) are the most famous learning rules, which depend on the activation patterns of spike trains between neurons, and they are implementable on low-power neuromorphic hardware devices. In this chapter, we introduce a novel bio-inspired spiking neural network endowed with spiking-based learning rule, to show how, using only neurons, synapses, and local learning, complex patterns can be learned. Moreover, we show how using synaptic normalization mechanism the unbalanced current between neurons can be solved in spiking neural networks. The organization of this chapter is as follows. Sec. 5.2 & Sec. 5.3 presents the most famous biological learning rules STDP and SDSP. Sec. 5.4 demonstrates the problem of edge orientation classification in SNN and how can be solved using unsupervised spiking-based learning rule and synaptic weights normalization. Sec. 5.5 STDP is applied to the particular application where the goal is to detect different touch modalities. In Sec. 5.6 we solved the problem of object shape classification using SNN and supervised STDP learning. Finally, Sec. 5.4.1 provides a discussion of our finding.

5.2 Spike Timing Dependent Plasticity (STDP)

Learning rules are algorithms performed in the synapses: depending on the activity of the synapses or of the neurons they are connecting. The learning rule changes the synapse's transmission capability by modifying its weight. More specifically, the way the weight changes depends on the behaviour of the neuron before the synapse (called PRE) and the neuron after the synapse (called POST). The weight of the synapse changes the amount of signal that the PRE is able to send to the POST. One of the most famous learning rule is (STDP) [172, 173, 101]. The latter is a form of learning that depends on the timing between the spikes of the PRE and POST neurons, connected by a synapse. STDP is often referred as the standard learning rule for emulating neural behaviour [174], as well as it is considered the basis of learning and information storage in the human brain. In the case where a presynaptic spike is followed closely by a postsynaptic spike, the synapse potentiates. However, when the postsynaptic spike is emitted shortly before a presynaptic spike, the synapse is depressed. Previous work by the authors [175], has successfully developed a learning algorithm that used a modified version of the conventional STDP rule, called the Cross Correlated (CC) rule. This algorithm was used to train a two-layer network and the results were benchmarked against existing supervised learning algorithms [176, 177]. Authors in [178] proposed a bio-plausible SNN model for SL based on the symmetric spike-timing-dependent plasticity (sym-STDP). They achieved good performance in the benchmark recognition task (MNIST dataset) by combining the sym-STDP rule with bio-plausible synaptic

scaling and intrinsic plasticity of the dynamic threshold. In [178] the readout synaptic weights (W) connecting hidden layer neurons to output layer neurons evolve according to a supervised rule through STDP using post-synaptic spike traces $tr_i(t)$ and post-synaptic target trace $tgt_i(t)$. The post-synaptic tr_i is updated at every post-synaptic spike and decays exponentially over a time constant as shown in equation 5.1

$$\tau_{tr} \frac{dtr_i}{dt} = -tr_i + \sum_f \delta(t - t_i^f) \quad (5.1)$$

In this equation τ_{tr} represents the time constant in ms and t_i^f the time were the post-synaptic (output layer) neurons fire a spikes. The weights updated at every pre-synaptic (hidden layer neuron) spike as shown in equation 5.2

$$w_{ij} = \alpha \cdot (tgt_i^{post} - tr_i^{post}) \delta(t - t_j^f) \quad (5.2)$$

where w_{ij} represents the synaptic weights dynamic between the j_{th} neuron in hidden layer and the i_{th} neuron in output layer, α is the learning rate, tgt_i represent the post-synaptic target trace which is constant, and $\delta(t - t_j^f)$ represents the time were the j_{th} neuron in hidden layer fire a spikes.

In this thesis, the STDP learning rule using post-synaptic spike traces and post-synaptic target traces has been used for learning the network to classify different touch modalities and object shapes applied on artificial sensory array.

5.3 Spike Driven Synaptic Plasticity (SDSP)

Initially proposed by Brader et al [179], the SDSP is counted as one of the most biologically plausible learning rules. Differently from the STDP, which depends only on the time between spikes, the synaptic weights in SDSP are dependent on two variables, as shown in figure 5.1: (1) the concentration of the slow postsynaptic calcium and (2) the state of the postsynaptic membrane potential at the time of the presynaptic spike. This rule has been already tested by Brader et al. [179] who demonstrated the ability of 2000 input neurons in classifying thousands of overlapping patterns from the MNIST dataset. They also demonstrated the ability of the network to generalize to other classification tasks with a similar or better performance than ANNs. Finally, they compared the long-term performance of STDP and SDSP and found that they were similar, but SDSP presented better generalization properties and biophysical accuracy. In addition, through presenting a VLSI network of spiking neurons and dynamic synapses, Mitra [180] showed the network's ability to perform real-time classification of complex patterns of mean firing rates. Moreover, another experiment [181] proposed an online learning digital spiking neuromorphic processor that utilized on-chip learning for the classification of 16x16 MNIST images. These findings establish that the SDSP learning rule is not only biologically realistic but also implementable on neuromorphic devices. Figure 5.1 shows the results of a Brian2 simulation where two neurons are connected in a PRE-POST link using SDSP-equipped synapses for a 500 ms stimulation period. Figure 5.1-A represents the presynaptic spike train. When input spikes arrive from the

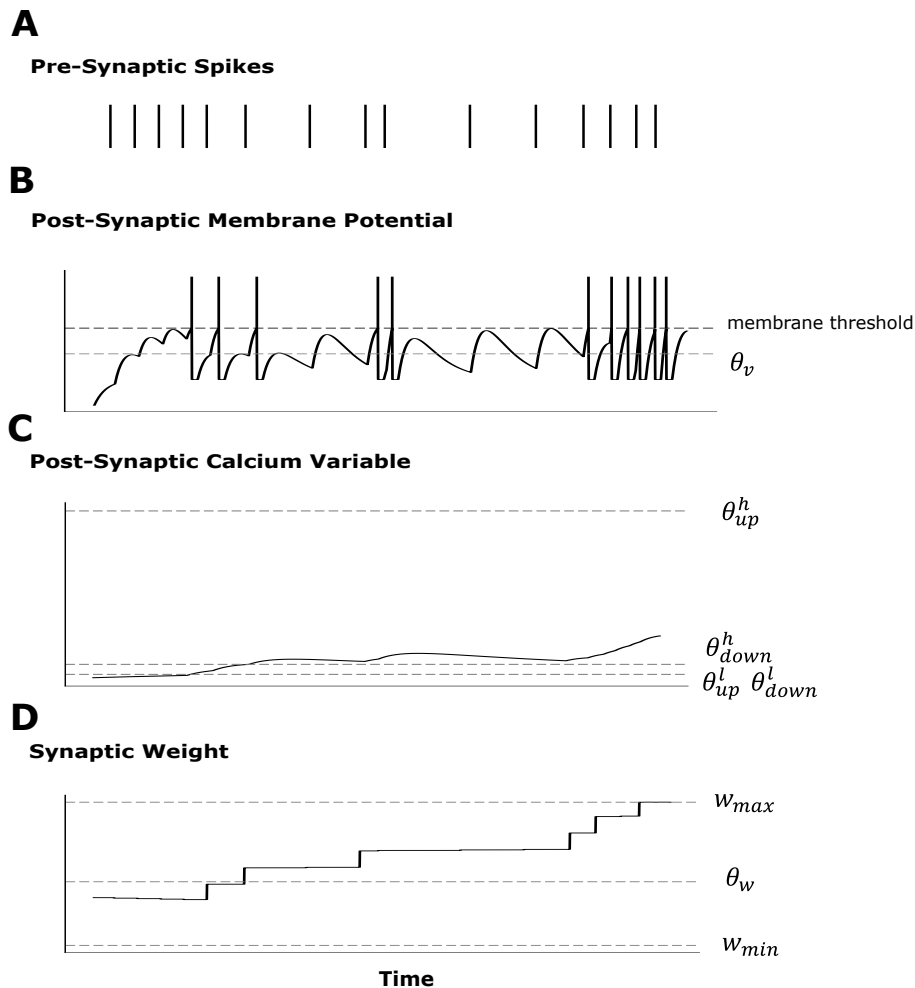


Figure 5.1 Toy example of two neurons connected via SDSP synapse. (A) Pre-synaptic neuron spike arrives at time t_k . (B) Post-synaptic membrane potential. (C) Post-synaptic calcium variable. (D) Synaptic weights variation as a function of time. If post-synaptic calcium variable and membrane potential meet the requirements at the moment of the pre-synaptic spike arrival, the synaptic weights potentiated by value A_+ .

presynaptic neuron, they increase the membrane potential of the postsynaptic neuron above the SDSP membrane threshold θ_v . This behaviour goes on until it reaches the neural membrane threshold, this triggers a spike in the post-synaptic neuron, as shown in figure 5.1-B. According to SDSP rule, every time the post-synaptic neuron emits a spike, the post-synaptic calcium variable $C(t)$, which represents calcium concentration, increases of a fixed value J_C . The calcium variable $C(t)$ is represented in mathematical

equation:

$$\frac{dC}{dt} = \frac{-C}{\tau_C} + J_C \sum_i \delta(t - t_i)$$

This variable is therefore proportional to the instantaneous neuron's spiking activity and it decays with a fixed time constant (τ_C), as shown in figure 5.1-C.

When a pre-synaptic spike arrives, the instantaneous values of $v(t_{pre})$ and $C(t_{post})$ indicates the change of the synaptic variable w_3 according to the following conditions:

$$\left\{ \begin{array}{l} w \rightarrow w + A_+ \quad \text{if } v(t_{pre}) > \theta_v \quad \text{and} \quad \theta_{up}^l < C(t_{pre}) < \theta_{up}^h \\ w_3 \rightarrow w + A_- \quad \text{if } v(t_{pre}) < \theta_v \quad \text{and} \quad \theta_{down}^l < C(t_{pre}) < \theta_{down}^h \end{array} \right.$$

In this conditions, A_+ and A_- are the learning jumps, θ_v is the voltage threshold and $\theta_{up}^l, \theta_{up}^h, \theta_{down}^l$ and θ_{down}^h indicates the thresholds for the calcium variable.

Under the mentioned conditions related with the calcium concentration and POST's membrane potential, the presynaptic spike triggers a variation in the internal weight of the synapse (W), increasing it by a value A_+ , as shown in figure 5.1-D. Despite the instantaneous adaptation, W is converging toward one of two stable states (W_{max} or W_{min}). The direction of the convergence depends on whether the current value is above or below a given threshold θ_w as shown in the following conditions:

$$\left\{ \begin{array}{l} \frac{dw}{dt} = \alpha \quad \text{if } w > \theta_w \\ \frac{dw}{dt} = -\beta \quad \text{if } w < \theta_w \end{array} \right.$$

Moreover, the synapses has efficacy J_+ if (W) is greater than θ_w , otherwise the synapse has efficacy J_- , as shown in figure 5.1-D.

5.4 Unsupervised Learning and Synaptic Normalization Mechanisms for Edge Orientation Selectivity

In chapter 3 we proposed a neuromorphic model for edge orientation selectivity, composed of artificial skin from the iCub robot and three layers of spiking neural network, based on computational primitives that are implementable on low power neuromorphic hardware devices. Moreover, A previous analysis was conducted in chapter 4 to propose a receptive field structure on the iCub skin to maximize the mutual information transmitted between layer one and layer two. The outcome of the study was that, in presence of a noiseless stimulus, higher information is delivered by the random shaped structure, while a more structured receptive field performed increasingly worse when reducing the length of the pressed bar. In this structure the different sensors present on the skin patch are assigned randomly to 16 different receptive field. The activation of one of the afferent in the receptive field activates the whole receptive field. More the active afferents in the receptive field, higher the receptive field activity. In the following parts, for reference, a random distribution has been generated and kept identical for all the simulations.

The goal of this work was to make the network learn the different orientations using temporal coincidence. For that reason, we endowed the network with local unsuper-

vised learning (SDSP) between layer two and layer three, in which the baseline network is equipped only with a WTA mechanism. Mutual Information has been used in this work for estimating the ability of the network to distinguish the different orientations. When a stimulus (i.e., a bar pressed with a specific orientation) is presented to the network the output layer response is used for estimating the MI. The amount of spikes emitted by the different neurons in the layers is calculated, after a binning in time. The neuron with the highest spike count is considered the winner and the positive number '1' is assigned to it in the vector containing all the output neurons. All the other neurons receive instead a '0'. This particular way of coding the output is necessary to overcome the drawbacks present in Information theory. Furthermore, the usage of WTA network at the output layer reduces the error generated by this simplification. As a result, a matrix composed by N time bins and M neurons is generated. This matrix is then used to create the joint probability where in each column is encoded the probability that a specific output neuron, given different input stimuli, spikes more than the others. This matrix can be defined as the joint probability between the two variables $p(X, Y)$. The information obtained by the formula has been then converted in "Number of Orientations Detected" using the concept that

$$\text{Recognised Orientation} = 2^{MI}$$

The unconstrained and random structure of the connectivity among layers can produce unbalanced activity in the output neurons, that are driven by a different number of

synaptic inputs. Two different forms of activity balancing (e.g. weights normalization and homeostasis) can be implemented at inference and learning time to create an homogeneous network that enables the system to discriminate between different bar's orientation applied on the skin patch.

In this section, three different networks are studied. The three networks share the same common structure: three layers of neurons, a random connection between layer one and layer two, and a learning connection between layer two and layer three. The difference between these three networks is that while the first network is equipped only with a WTA (Baseline Network Sec. 5.4.1), in the other two there is weight normalization (Weight Normalization Network Sec. 5.4.2) and homeostasis (Final Network Sec. 5.4.3), respectively. These two techniques are used to balance the activity of the output neurons.

5.4.1 Baseline Network

The baseline network (figure 5.2-A) was trained with multiple consecutive stimuli presentations, at randomly selected orientations in the range 0° - 180° (5° resolution). During learning, the synapses connecting layer two to layer three were potentiated or depressed depending on the reciprocal activation of layer two neurons.

Figure 5.2-B shows the synaptic weights before and after learning. The synaptic weights were initialized using an uniform random distribution (figure 5.2-B left) such that all the weights were depressed. After learning, only a small percentage of weights were potentiated (green squares in the layer two and layer three connectivity matrix

of figure 5.2-B right) and, correspondingly, several layer three neurons did not learn any of the orientations presented during the learning phase, in which all the synapse weights connecting to them were still depressed (red squares in the layer two and layer three connectivity matrix of figure 5.2-B right). Nevertheless, most likely a subgroup of output neurons learnt several orientations, leaving some output neurons non-selective to any orientation.

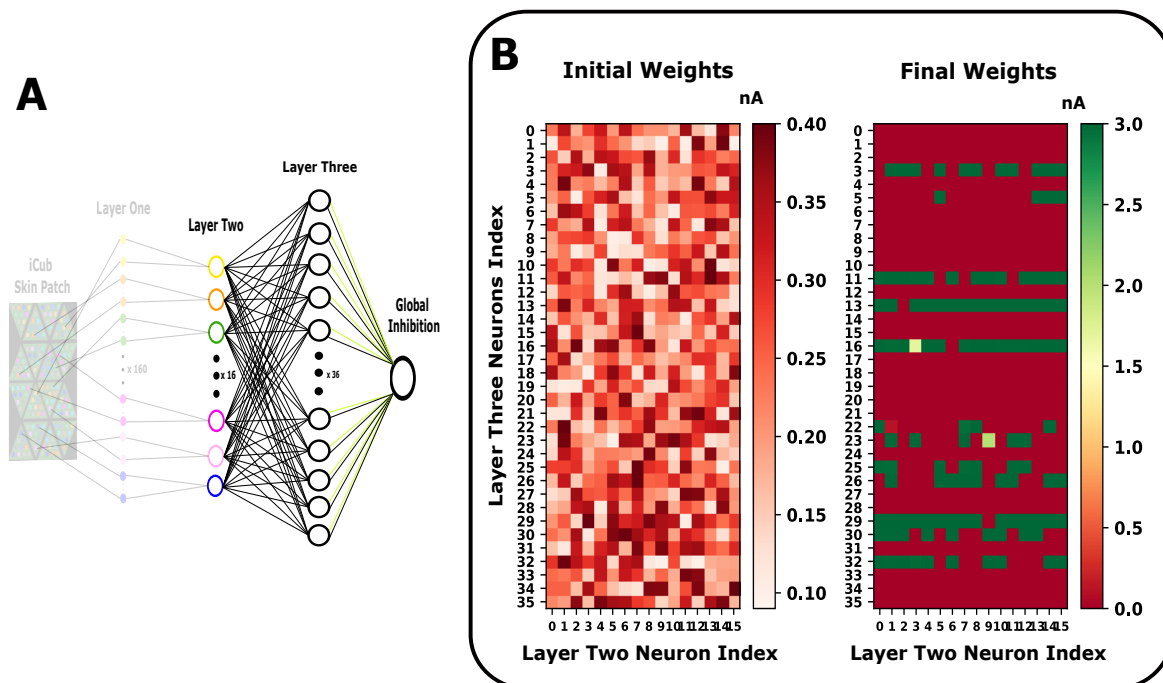


Figure 5.2 SDSP Learning rule with Baseline Network. (A) Baseline network with all to all connection between layer two neurons and layer three neurons endowed with SDSP synaptic learning rule to build the required connectivity through potentiation or depression of the synaptic weights and to learn an appropriate connectivity patterns for the classification. (B) Synaptic weights at the initial and final state of the simulation; (left) The synaptic weights between layer two and layer three were initialized using uniform random distribution such that all the weights were depressed (red boxes); (right) The synaptic weights at the end of the learning, each small box represents the synaptic weights of one neuron in layer two (x-axis) connected to one neuron in layer three (y-axis), where most layer two neurons to layer three neurons connecting weights were still depressed (red) and some potentiated (green).

To illustrate the spiking response of the third layer for different edge orientations, we fed the network with 36 orientations ranging from 0° to 180° with step 5° . Right part of Figure 5.3-B represents the spiking responses of layer three neurons with baseline network. In this case it's visible that while some neurons responded to several patterns some other neurons were unresponsive.

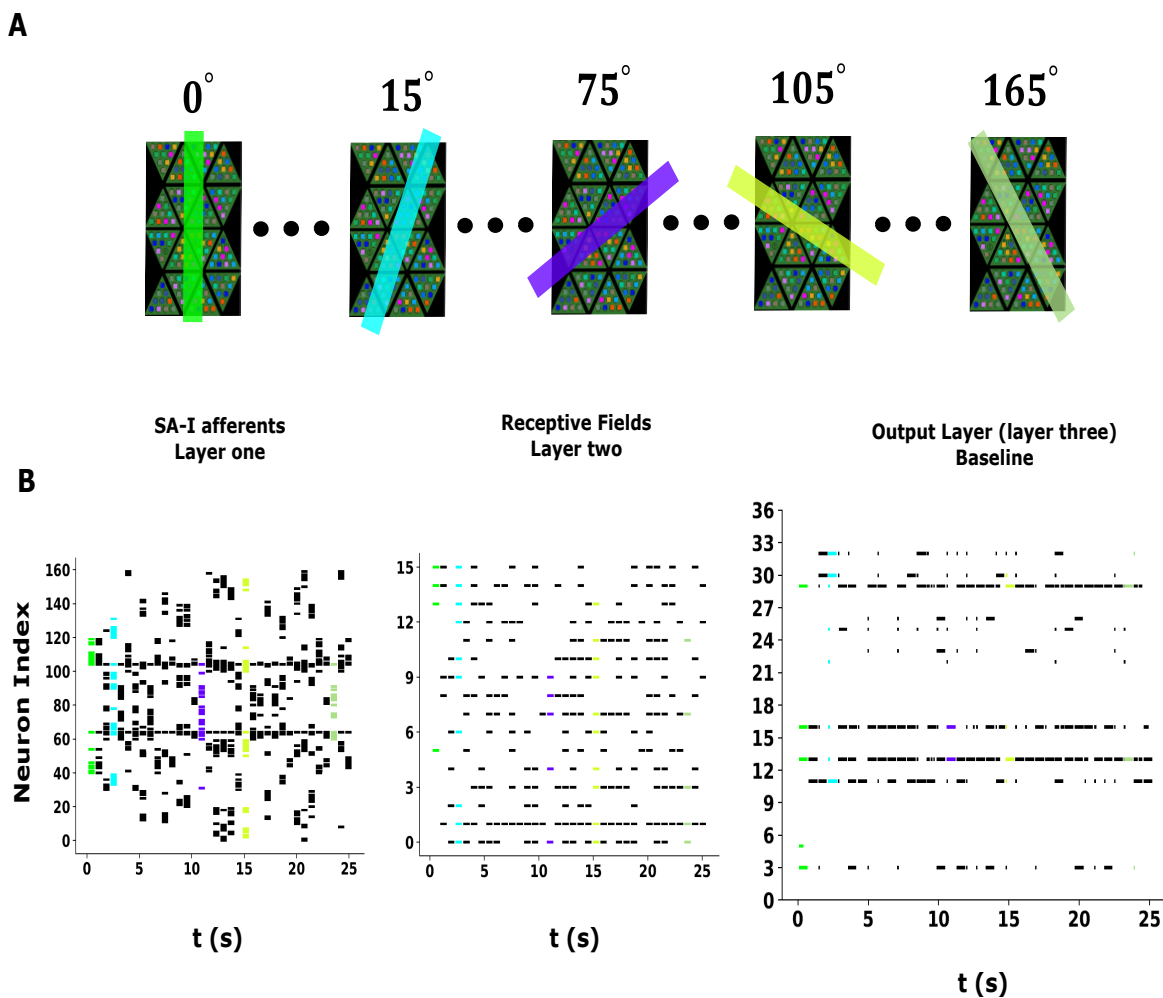


Figure 5.3 Spiking responses of the three layer of the neuromorphic network architecture. (A) Examples of different bars pressed on the skin at different orientations (0° , 15° , 75° , 105° , 165°). (B) Raster plot of layer one neurons (SA-1 afferents) as a function of time for 36 orientations (0° to 180° with step 5° increments) (left), raster plot of layer two neuron (receptive fields) (middle), and (right) raster plots of layer three neurons as a function of time for 36 orientations with baseline network (colored spikes trains match the orientations given in part (A) and the black spike trains for the remaining orientations).

For quantitative assessment, we divided the learning simulation into 10 epochs. In each epoch, we extracted the weights and tested the network on two different datasets. The first dataset was composed of 36 orientations ranging from 0° to 180° with 5° step increments, randomly repeated for many times, where in each dataset the pressure is constant. The second dataset was similar to the first except that the pressure was varying. We used mutual information to compute the number of orientations and the information the network decodes at layer three. Figure 5.4 shows the mutual information as a function of number of epochs. In each epoch, we computed the number of orientations detected using the mutual information (number of orientations = 2^{MI}). As such, using the first dataset, the baseline network was able to detect only six orientations at the end of learning, whereas, using the second data set, the network was able to detect only three orientations.

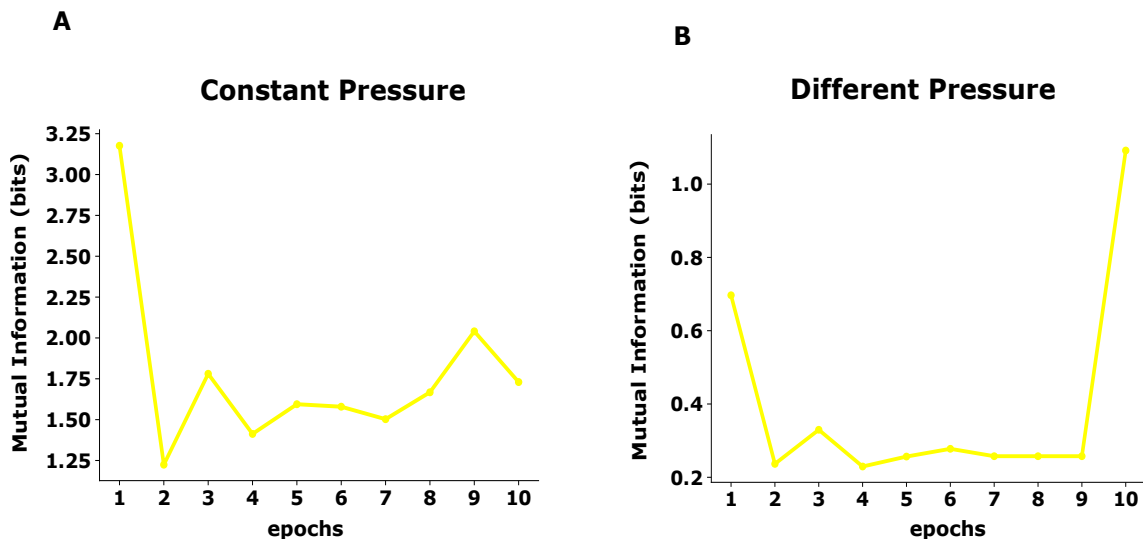


Figure 5.4 Network performance for the baseline networks based on mutual information. (A) Mutual information as a function of epochs with constant pressure dataset; in each epoch 36 orientations ranging from 0° to 180° with step 5° increments, randomly repeated, and all with constant pressure. (B) The benchmark of baseline network when varying pressure stimuli dataset were used.

Figure 5.4, shows that the baseline network doesn't recognize all the possible input orientations. This behaviour is given by the presence of different amount of excitatory currents arriving from layer two to layer three neurons at each orientation. In fact, due to the non-uniform distribution of taxels in the layer two receptive field, it can happen that an output neuron encoding a specific orientation which has just three layer two neurons connected to it will always spike weaker than an output neuron that has six layer two neurons connected to it (middle part of figure 5.3-B). This becomes even more clear if we consider that several neurons at the layer two are encoding in their spiking different orientations, making it difficult for the network to differentiate between those.

5.4.2 Weights Normalization Network

Given the problem highlighted in the baseline example, a way to reduce the unbalanced input current from layer two to layer three was tackled. Specifically, to assure that the response of different neurons competing with each other was the same, regardless of the quantity of input synapses connected to them, weight normalization was used. The latter is a technique where synapses' input is reduced according to some algorithm in the network. In the weights normalization network, we changed the current magnitude of each synapse depending on how many active synapses were connected to a layer three neuron (i.e. how many weights were potentiated).

Early studies have suggested that weights normalization could be one of the key mechanisms for classification problems in SNNs. A previous work from Mostafa et al.[182]

utilized weight normalization in a MNIST classification task, setting the normalization weights as the following: the neuron spikes when its membrane potential crosses a firing threshold which was set to 1, where the firing threshold was dynamically changed using spiking activity. In another work from Soures et al. [183], the variation in synaptic weights was considered as a function of the activity of a larger set of neurons. As they state, the normalization of synaptic weights enables the network to achieve better classification accuracy through adding global constraints to the synapse's strength. When the sum of the currents arriving from pre-synaptic neurons to post-synaptic neurons exceeds a certain threshold, the synaptic weight decreases, such that the current remains within the identified threshold.

In the weights normalization network, we introduced a component that reduces the current coming from the synapses looking only at the weights of the connection to a specific neuron. This way of implementing synapse normalization has the advantage of implementing local changes without the need of global variables. Using the same dataset as the previous example, we studied how the weights normalization network was able to discriminate between different orientations. The weight normalization was especially useful during learning: synaptic weights, driven by SDSP rule, were used to define dynamically the current that each synapse was able to conduct. When a weight between one layer two neuron and one layer three neuron exceeded a normalization threshold, then all the synapses connected to that layer three neuron were reduced in magnitude, as shown in figure 5.5-A. This resulted in a balance of activity between different layer three neurons, helping the learning procedure. As in the baseline

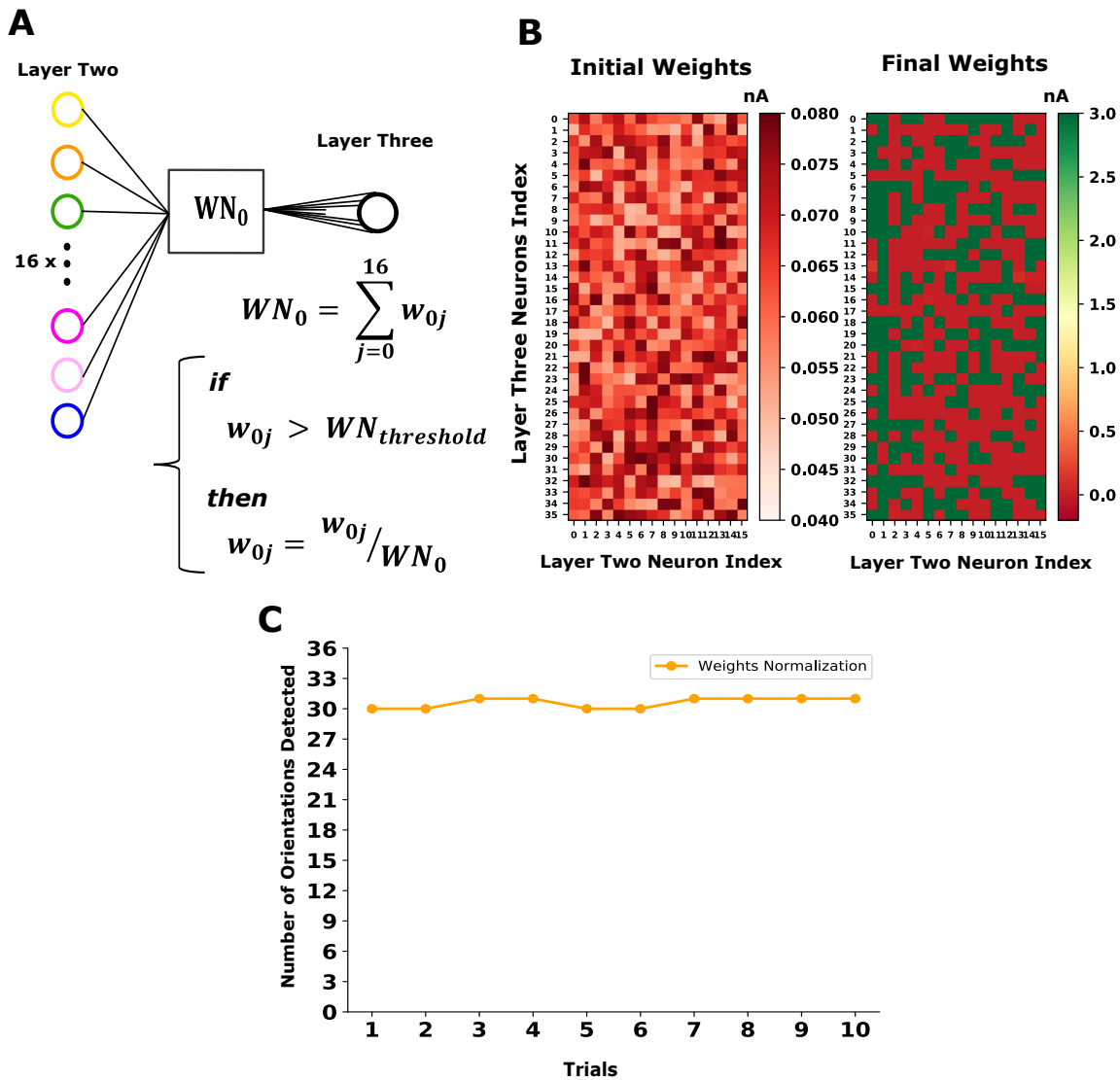


Figure 5.5 SDSP learning rule with weights normalization network. (A) Schematic representation of the weights normalization mechanism between layer two neurons and layer three neurons; synaptic weights connecting all layer two neurons to a single neuron in layer three were normalized by dividing the weights by the summation of the activated synapses (i.e., how many weights exceeds the normalization threshold). (B) Synaptic weights at the initial and final state of the simulation; (left) The synaptic weights between the two layers were initialized using same synapses random distribution as the previous example; (right) The synaptic weights at the end of the learning, each row represents the synaptic weights connecting all neurons in layer two to one neuron in layer-three state: depressed (red) or potentiated (green), where the weights normalization mechanism help the learning procedure in a balance of activity between different neurons and converging the synaptic weights to stable connectivity patterns. (C) Number of orientation detected by weights normalization network for 10 different trials, by changing the weights initialization random values in each trial.

network, the synaptic weights between the two layers were initialized using uniform random distribution (figure 5.5-B to the left) with all the weights depressed at the initial state.

After learning, synaptic weights progressively change to converge to stable connectivity patterns, as shown in figure 5.5-B to the right.

Thanks to SDSP learning rule and weights normalization mechanism, each neuron in layer three learned a specific orientation. To study the accuracy performance of the network with different initialization of the the weights, we repeated the simulation for 10 different trials, in each trials the initial weights randomized in different distributions. Figure 5.5-C shows the number of detected orientations at each trial, where the number of detected orientations looks similar for all trials.

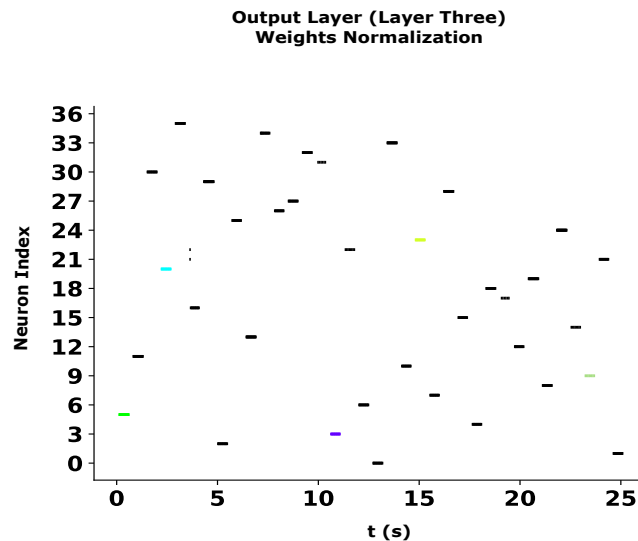


Figure 5.6 Raster plot of layer three neurons as a function of time for 36 (ranging from 0° to 180° with 5° step increments) orientations with weights normalization network (colored spikes trains match the orientations given in figure 5.3-A and the black spike trains for the remaining orientations).

Figure 5.6-B shows the raster plot of layer three neurons with weights normalization network, where for each orientation only one neuron is firing.

Figure 5.7-A shows the improved performance of the weight normalization over the baseline network, in which the number of detected orientation was equal to 35 orientations from 36 orientations (ranging from 0° to 180° with 5° step increments). Whereas Figure 5.7-B shows that the weight normalization was not able to infer all the orientations when pressure variation was present (31 from 36 orientations). Therefore, the network ability to infer can be in fact worsened by heterogeneous pressure levels, during inferring, in which the pressure heterogeneity is an important feature in a network that wants to detect real-world stimuli.

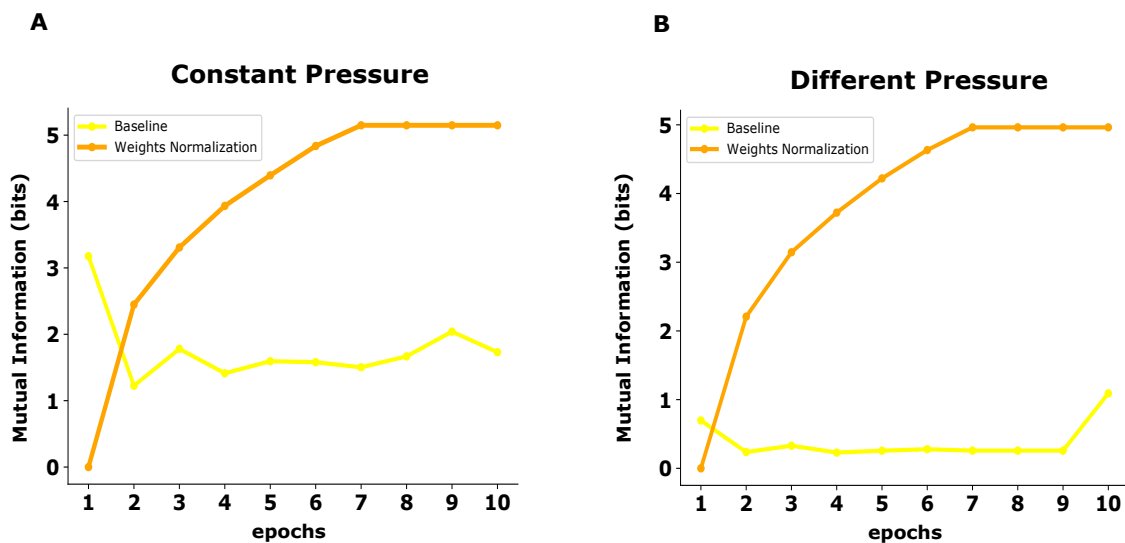


Figure 5.7 Network performance comparison for the two different networks (Baseline, Weights Normalization) based on mutual information. (A) Mutual information as a function of epochs with constant pressure dataset; in each epoch 36 orientations ranging from 0° to 180° with step 5° increments, randomly repeated, and all with constant pressure. (B) The benchmark of the two networks (baseline and weights normalization networks) when varying pressure stimuli dataset were used; the weights normalization network Network outperformed the baseline network by detecting more stimulus orientations.

5.4.3 Homeostasis Network

As an alternative to weight normalization to balance the neurons' synaptic drive and allow the network to discriminate between different orientations, we explained the use of homeostasis between layer two neurons and layer three neurons. Synaptic homeostasis is a mechanism observed in biological neural systems that acts to maintain a homogeneous response from neurons within a given operating range [184]: when neuron's spiking activity stably leaves given boundaries, the synaptic drive is scaled to restore the activity within their functional operating range. The process is usually very slow and account for drifts of activity, without interfering with signal processing.

Several different types of homeostatic mechanisms have been observed in biological neural networks, comprising both forms of synaptic plasticity mechanism and modification of intrinsic properties of the excitability of the neuron [185–187]. Global homeostatic synaptic scaling allows for the control of the network's overall stability, while complying to the need for change (or learning), to adapt to the statistic of the input signals [188]. In [189] authors present a synaptic circuit that supports both spike-based learning and global synaptic scaling homeostatic mechanism.

In our implementation, based on [189], the synaptic excitatory current, triggered by layer two neurons and fed into layer three neurons, is compared to a target current.

The homeostatic control monitors the activity of layer two neurons at each orientation and adapt the weights connecting layer two neurons to layer three neurons globally when the pressure of the pressed bar increased or decreased, so that the excitatory

current driving from layer two neuron to layer three neurons should be equal to a target current I_{target} , as in the following equation:

$$\tau_{homeo} \frac{dhomeo_i}{dt} = -homeo_i + \frac{\sum_{j=1}^n (w_{3ji} S_{2j}(t - t_k))}{I_{target}}$$

In this equation, τ_{homeo} is the homeostasis time constant, w_{ji} represents the synaptic weights connecting layer two neurons to layer three neurons and S_2 represents the spatio-temporal output spikes of layer two j_{th} neurons. T_k is the time in which the neurons in layer two fire a spike.

If the value is higher the homeostatic plasticity changes the normalization of the weights.

$$w_{3ji} = \frac{w_{3ji}}{homeo_i}$$

The network with homeostasis has the advantage to adapt when the output pressure of the artificial tactile sensors steadily changes during pressing.

As in the weights normalization network, Figure 5.8 shows that with homeostasis network, each neuron in layer three is selective to a specific orientation and firing along with respect to the input orientation.

The main advantage of homeostasis network over the weights normalization mechanism is that homeostasis performs better with stimuli with varying pressure. The results highlight that in the constant dataset both the weights normalization and the homeostasis networks act the same both detecting thirty five orientations as shown

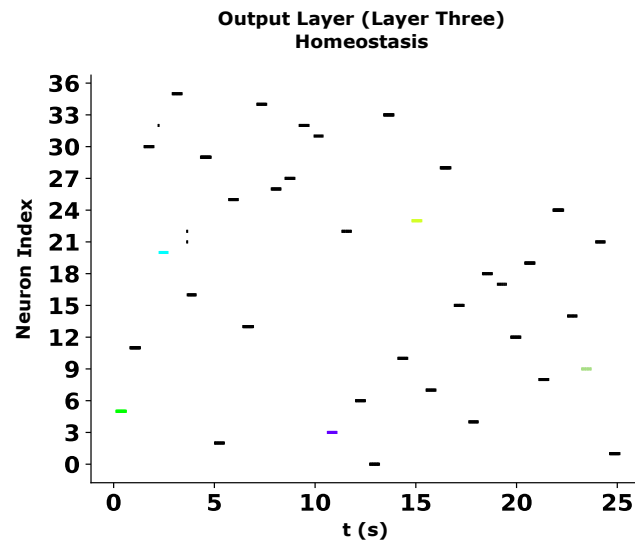


Figure 5.8 Raster plot of layer three neurons as a function of time for 36 (ranging from 0° to 180° with 5° step increments) orientations with Homeostasis network (colored spikes trains match the orientations given in figure 5.3-A and the black spike trains for the remaining orientations).

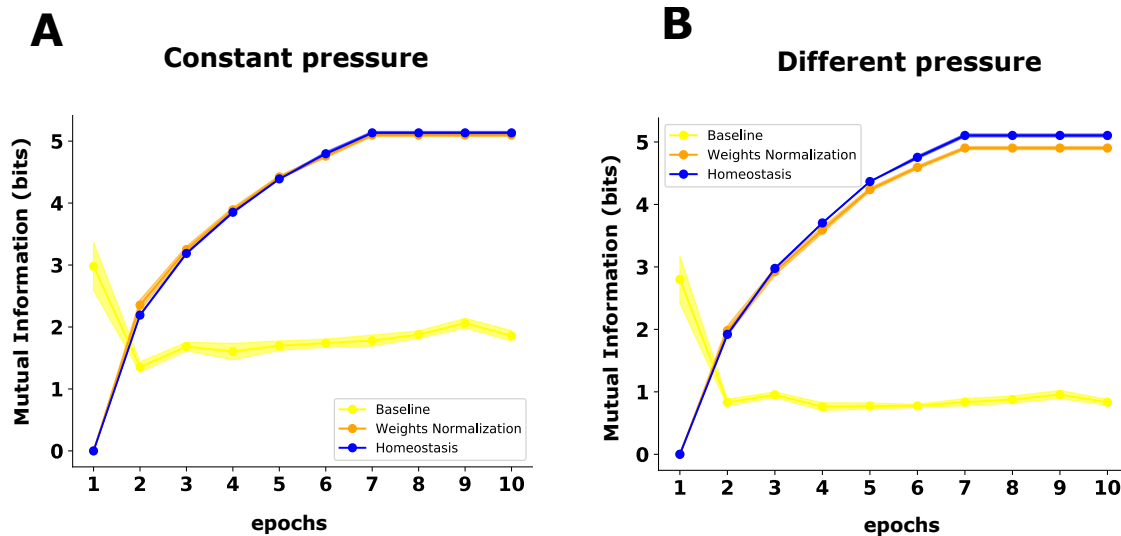


Figure 5.9 Network performance comparison for the three different networks (Baseline, Weights Normalization, and Homeostasis) based on mutual information. (A) Mutual information as a function of epochs with constant pressure dataset; in each epoch 36 orientations ranging from 0° to 180° with step 5° increments, randomly repeated, and all with constant pressure. (B) The benchmark of the three networks when varying pressure stimuli dataset were used; the homeostasis Network outperformed the weights normalization and baseline network by detecting more stimulus orientations.

in figure 5.9-A. However, in the varying pressure dataset, the homeostasis network outperformed the weights normalization by detecting 35 orientations compared to 31 orientations detected by the weights normalization network. Thanks to homeostasis control, the network was able to discriminate between 35 orientations as shown in figure 5.9.

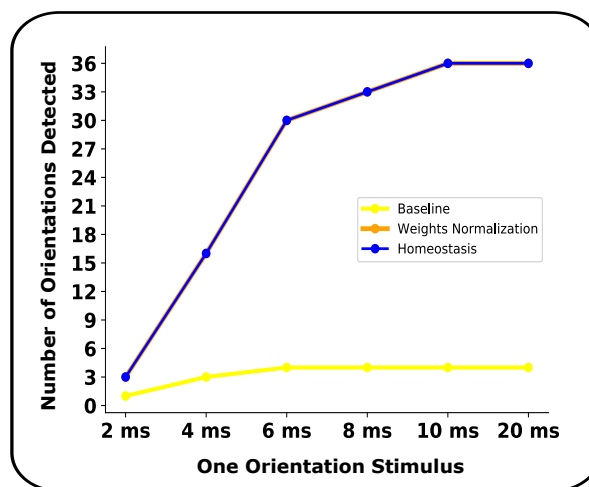


Figure 5.10 Accuracy of the three networks as a function of stimuli with different intervals of times; the duration of stimuli changes from $2ms$ to $20ms$ with a step increment of $2ms$, both Homeostasis and weights normalization networks outperformed the baseline network in detecting a different number of orientations with different time interval.

In order to demonstrate that this architecture can be used in real-time applications, we measured the accuracy of the three networks when stimuli are presented in different interval of times. The duration of stimuli presentation changes from $2ms$ to $20ms$ with a step size of $2ms$. Figure 5.10 shows that the number of detected orientations increases as a function of the duration of the stimulus presentation, the system then reaches a maximum at $10ms$, recognising all between the 36 orientations both with the weights normalization and homeostasis network.

5.5 Touch Modality Classification Using Supervised STDP

Learning

This section presents the experiments done to demonstrate the ability of the network presented in Chapter 3 Sec. 3.4 in learning different touch modalities. To achieve this goal we endowed the network with supervised STDP learning rule between hidden layer and output layer neurons with all-to-all connections. During the learning phase, the network fed the output layer with multiple consecutive stimulus presentations, randomly repeating the six touch modalities (poke, press, grab, squeeze, push, and rolling a wheel). Each stimulus presentation is encoded in the spiking activity of number of neurons of the hidden layer. Moreover, the learning dataset composed of 180 interactions (6 modalities X 30 each), repeated many times. In addition output layer neurons are fed with external stimulus spikes trains that act as teacher signals. Thus, when one neuron in the output layer spikes thanks to the combination activity of the input synapses and of the teaching signal, the synapses that are active are potentiated, those that are less active and hence have a lower probability of falling in the positive region of the STDP curve are depressed. Therefore the potentiation or depression depends on the combination between the activated neurons for each modality at hidden layer and the teaching signal.

Figure 5.11 to the left represents the synaptic weights connecting all neurons in hidden layer to one neuron in output layer, in which that neuron learned one of the touch modalities, where the green curve represents the potentiated synapses and the orange

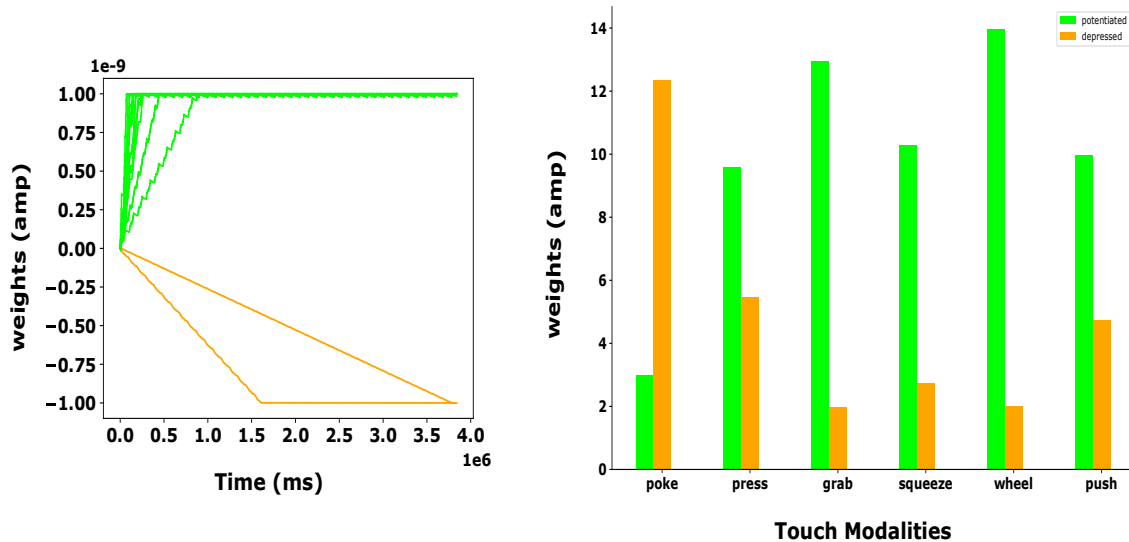


Figure 5.11 STDP learning phase; (Left) variation of synaptic weights connecting all neurons in hidden layer to one neuron in output layer for each presynaptic spike as a function of time. (Right) summation of synapses weights at the end of learning (green for potentiated synapses and orange for depressed synapses) connecting all neurons in hidden layer to each neuron in output layer(x-axis). Orange bars are multiplied by -1 for better presentation.

one represents the depressed synapses. The synaptic weights are bounded between two values w_{max} and w_{min} . After learning synaptic weights connecting hidden layer neuron with output layer neurons progressively change to converge to stable connectivity patterns. Figure 5.11 to the right represents the summation of the potentiated (green) and depressed (orange) weights connecting hidden layer neurons to each neuron in output layer (one neuron per touch modality), where each neuron has its own connectivity pattern that depends on the activated neurons in hidden layer for each modality. For example, for the poke modality, the summation of the potentiated synapse weights and the depressed weights that connected hidden layer neurons to the poke

neuron are equal to $3nA$ and $13nA$ respectively, which means that 3 synaptic weights potentiated and the others depressed.

To validate the proposed network, the synapses weight obtained after leaning the network with the 6 touch modalities have been extracted. Moreover, the learning contributions have been switched off. In the testing phase, all output neurons fired according to inputs from the hidden layer. To achieve better selectivity, and also to increase the classification accuracy of the network during the testing phase, a fast lateral inhibitory layer composed of 6 inhibitory neurons has been added to the network. The dataset used for testing is composed of 10 samples from each of the six modalities.

Figure 5.12 shows the confusion matrix after testing. The poke modality has the highest accuracy in comparison with the other touch modalities with 100% acuity. Moreover, most of the touch modalities have an accuracy equals to 90% which overcomes similar state of the art work [190]. These results indicate that the touch modalities are easily

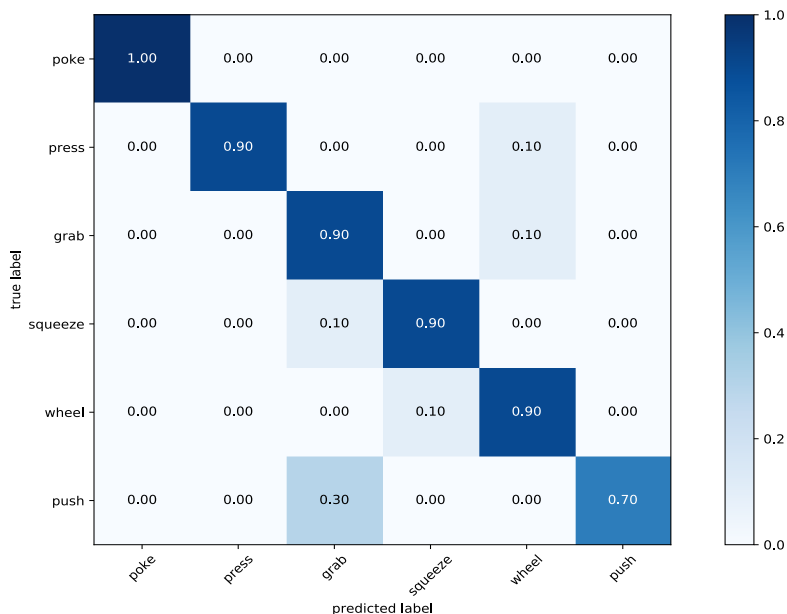


Figure 5.12 Confusion matrix for touch modalities classification.

recognized by the network. The lower accuracy was for the push modality type, where it was equal to 70%. Nevertheless, the accuracy achieved by the proposed network equals to 88.3%, whereas the accuracy presented in [190] was equal to 71%.

5.6 Object Contact Shape Classification using Supervised-STDP

one of the important applications of robots is multisensory object recognition. Recently tactile based object recognition in robotics has gained increasing interest. Inspired by the achieved results obtained with edge orientation detection and touch modalities classification, in this section, we aim to classify different objects shapes applied on sensory array using spiking neural network and synaptic learning. In the remainder of this section, we will present the software implementation including the learning and testing phase for distinguishing between different shapes in Sec. 5.6.1 along with a hardware implementation for real time classification (Sec. 5.6.2).

5.6.1 Software Implementation

Learning Phase In the previous sections we solved the problem of classifying different bar's orientation applied on an artificial skin using a three layers spiking neural network and unsupervised learning, as well as touch modalities classification using similar network architecture of three layers of spiking neurons and supervised spike based learning rule. Here, in this study, our goal was to solve the problem of object shapes

classification using a network architecture composed of spiking neurons inspired by the finding of previous experiments. Besides, the network architecture developed for object shapes classification composed of two layers. As in the previous sections, we aim to learn the connectivity between layer one neurons and layer two neurons, such that the network could distinguish between different object shapes presented as input stimuli. To achieve this aim, we endowed the network with the STDP rule between layer one and layer two neurons with all to all connection, in which the synapses weights were initialized as zeros. During the learning phase, the network fed the output layer with multiple consecutive stimulus, randomly repeating the eleven object shapes. Each object shape is encoded in the spiking activity of several neurons in layer one. The training has been repeated over 10 folds where in each fold the dataset is randomly divided into 80% for training and 20% for testing. The training accuracy achieved a 100% accuracy over the 10 runs. In addition, output layer neurons are fed with external stimulus spike trains that act as teacher signals during learning. Thus, when a neuron in the output layer spikes thanks to the combination activity of the input synapses and of the teaching signal, the synapses that are active are potentiated. Otherwise, the synapses are not active are depressed.

Figure 5.13 represents the variation of synaptic weights connecting all neurons in layer one to neurons in layer two as a function of time. Each box in Figure 5.13 shows the synaptic weights change connecting all neurons in layer one to one neuron in layer two (one neuron per object shape). The synaptic weights are bounded between two values w_{max} and w_{min} . After learning, synaptic weights connecting layer one neurons

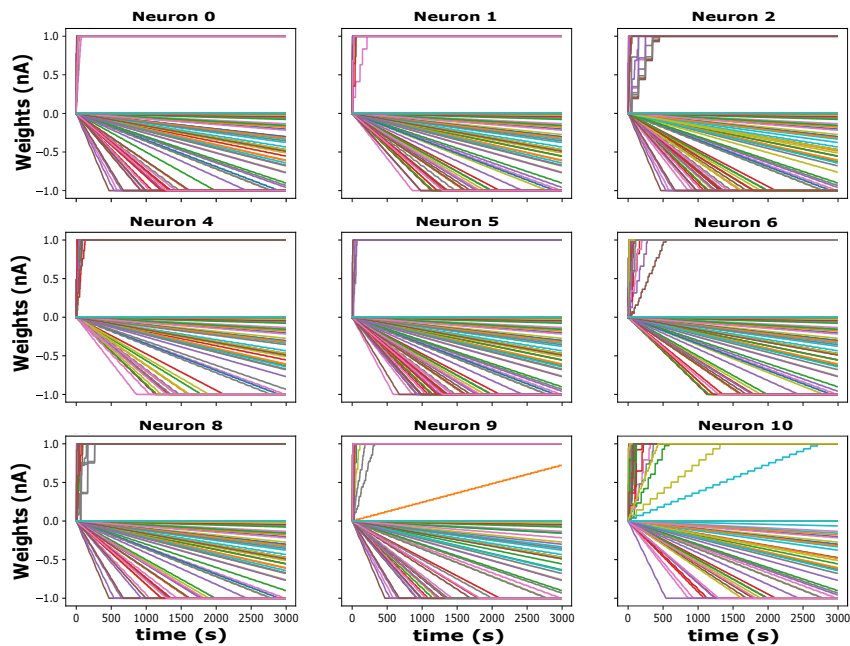


Figure 5.13 Variation of the synaptic weight for each presynaptic spike as a function of time. Each box represents the synaptic weights between all neurons in layer one to one neuron in layer two.

with layer two neurons progressively change to converge to stable connectivity patterns. Thanks to STDP learning each neuron in layer two learned specific connectivity patterns, which means that each neuron in layer two learned specific object shape.

Testing Phase To validate our proposed network, the synapses weight obtained after learning the network with 11 different object shapes have been extracted. Moreover, the learning contributions have been switched off. During the learning phase the synaptic weights were modified according to the combination activity of the input synapses and of the teaching signal. Whereas in the testing phase, the activation of neurons in layer two depend only on the synaptic current inputs arriving from layer one neurons at each object shape presented as input stimulus. Inspired by the network developed

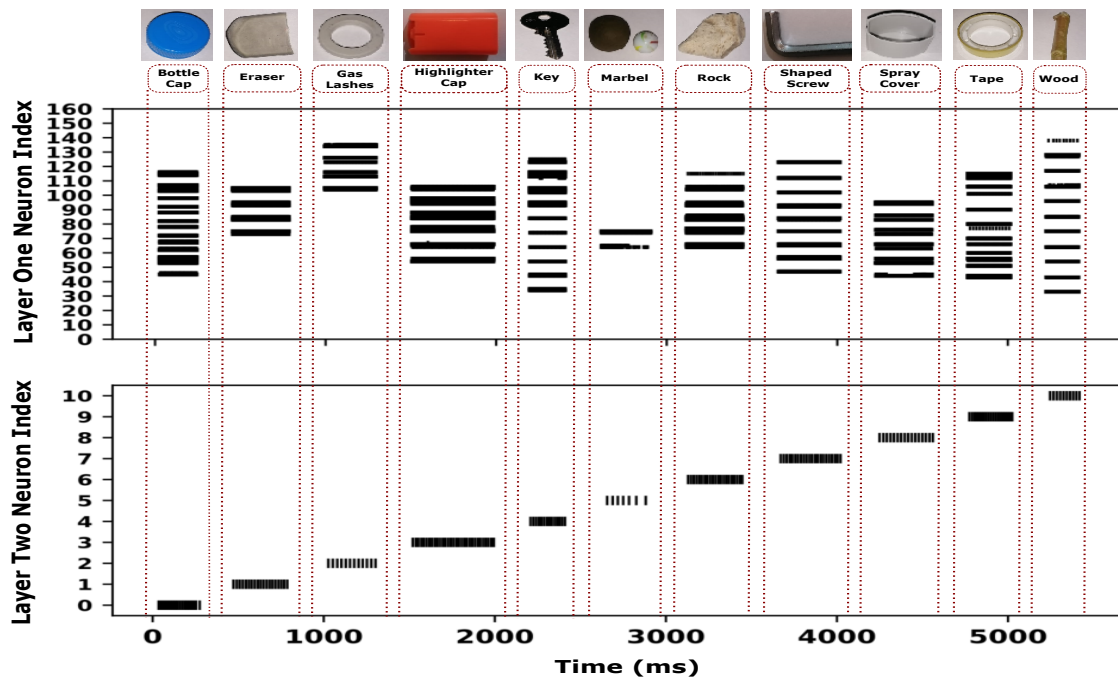


Figure 5.14 Output spikes of layer one and two:(top) 11 different object shapes; (middle) each object shape corresponds to the activation of multiple neurons in layer one; (bottom) Output spikes of neurons in layer two after learning, such as for each object shape only one neuron in layer two is firing.

for edge orientation classification, and to achieve better selectivity during testing, we implemented WTA competition composed by global inhibitory neuron. In the testing phase, the 20% of the dataset allocated to the testing have been fed to the network. Each of the 11 objects shapes was detected by the network with an accuracy of 100 %. Figure 5.14 shows the raster plots of layer one neurons for each of the 11 objects. Each object activates several neurons representing the sensors. Besides, a raster plot of the layer two after decoding the object shape is also provided. For every object shape, only one neuron in layer two is firing, giving out the predicted shape.

In order to demonstrate that this architecture can be used in real-time applications, we measured the accuracy of the networks when stimuli are presented in different

number of samples. Starting by the lowest number of samples produced by the sensors (1 sample), then increasing the number of samples by step one increments. Figure 5.15 shows that the accuracy increases when the number of samples produced by each sensors increase. Therefore, the minimum number of samples required for detecting the whole object shapes with accuracy 100 % was equal to nine samples per each sensor.

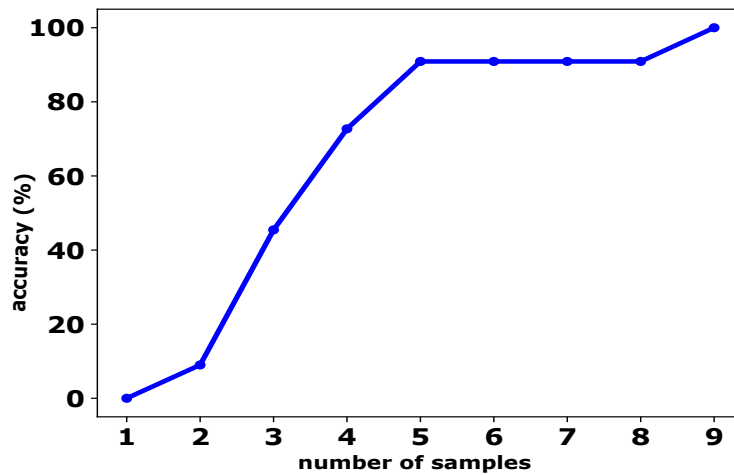


Figure 5.15 Network accuracy as a function of number of samples.

Computational Cost Counting the number of floating-point operations FLOPs is considered as a metric to analyze the efficiency of an algorithm. The number of FLOPs gives an indicator on how fast a model will perform FLOPs. In this study, we counted the FLOPs that the proposed SNN network uses in order to classify a single object shape. The total number of FLOPs required by the network can be estimated as follows:

$$TotalFlops = (\#FLOPs)(\#iterations) \quad (5.3)$$

where the number of #FLOPs denotes the number of FLOPs per iteration and #iterations represents the total number of iterations for the given time of run:

$$\#iterations = \frac{Duration}{dt} \quad (5.4)$$

In addition, we measured the CPU execution time the network required to classify single object shapes. Table 5.1 represents the FLOPs, CPU time, and accuracy for three different time steps (dt).

<i>dt</i>	<i>2ms</i>	<i>1ms</i>	<i>0.1ms</i>	<i>0.01ms</i>
<i>FLOPs</i>	10770	19128	179292	1674012
<i>CPUtime</i>	0.17s	0.1875s	0.21875s	0.421875s
<i>accuracy%</i>	80	90	100	100

Table 5.1 SNN computational cost

5.6.2 Hardware Implementation

Based on the promising results achieved with the proposed network in terms of computational cost and high classification accuracy, a validation of the network on an embedded hardware platform has been targeted. For that, since the Brian2 simulator is a python based platform, we decided to implement the SNN network on a Raspberry pi device since the codes could be easily converted. After that, the weights obtained after the learning phase have been used between layer one and layer two of the network. Note that the same interface electronics and sensors mentioned in Sec. 5.6.1 were used. Figure 5.16 shows the block diagram of the hardware setup. The application scenario was as follow: first, the object is pressed on the sensory array, in which the

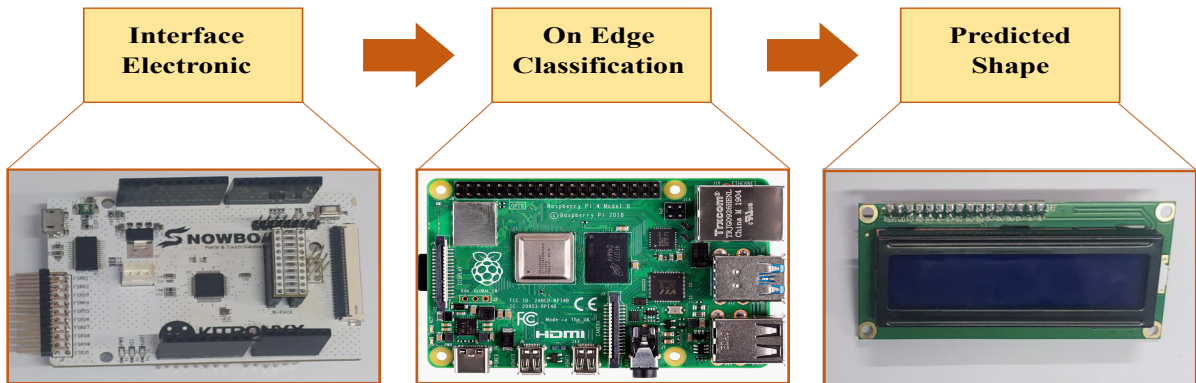


Figure 5.16 Real time object shape classification setup.

interface electronic collects the tactile stimuli. Second, the Raspberry pi with the implemented SNN network executes the inference and reads the sensory data. The communication between the Raspberry pi and the snowboard, was achieved through a UART communication. Finally, the Raspberry pi communicates the predicted shape to be displayed on a Liquid Crystal Display.

5.7 Discussion

In this chapter, a neuromorphic spiking neural network has been implemented on a neuron simulator to show how, using only neurons, synapses, and local learning, complex patterns can be learned.

In the first part of this chapter, a neuromorphic spiking neural network composed of three layers of LIF neurons endowed with unsupervised spike-driven learning was introduced. This network is used to solve the problem of distinguishing between different bar orientations applied on the artificial skin from the iCub robot. Several issues appeared during the learning and inference procedure. First, when the network

is equipped only with a WTA structure, several neurons in the output layer did not learn any of the orientations presented during the learning procedure. Besides, a subgroup of output neurons learned several orientations, leaving some output neurons non-selective to any orientation. This behavior is given by the presence of the different amounts of excitatory currents arriving from layer two to layer three neurons at each orientation and the non-uniform distribution of taxels in the layer two receptive fields, which yield to unbalanced input current from layer two to layer three neurons.

This issue was solved using weights normalization endowed between layers two and three neurons: the number of active synapses connected to each output neuron was defining the degree of normalization each synapse was getting. The higher the number of active synapses connected to the output neuron, the lower the strength of each synapse. This method allows the synaptic weights to change progressively during the learning phase converging to stable connectivity patterns, increasing the spatial acuity of the network. Nevertheless, the network with weights normalization mechanism has the ability to distinguish between 35 orientations of 36 orientations presented to the network as input stimuli with constant pressure. Whereas, the network with weights normalization was not able to infer all the orientations when pressure variation was present in the input stimuli, detecting only 31 orientations of 36 orientations.

Another problem that was tackled in this study which is the different pressure levels that a pressed bar can exert on the skin. The network ability to infer can be in fact worsened by heterogeneous pressure levels, during inferring. This issue was solved using homeostasis, where the activity of the neurons triggers a normalization of the

synapses. The mechanism can be also used to compensate for drifts in sensor behaviours, given by fatigue or aging. The main advantage of the homeostatic mechanism over the weights normalization mechanism is that the homeostasis performs better with stimuli with varying pressure. In the varying pressure stimuli, the homeostasis network outperforms the weights normalization by detecting 35 orientations compared to 31 orientations detected by the weights normalization network.

In the second part of this chapter, we present the implementation of a spiking neural network endowed with a supervised spike-based learning rule (STDP) for touch modality classification. The network has demonstrated the ability to learn appropriate connectivity patterns for the classification, achieving a total accuracy of 88.3% overcoming similar previous work.

Finally, we demonstrated that based on temporal coincidence, a spiking network with two layers of LIF neurons and endowed with supervised learning has the ability to build up the required connectivity between layers for distinguishing between different object shapes. Moreover, the network is capable to distinguish between eleven different objects shapes, achieving a total accuracy of 100 %, when a time step of 0.1 ms was used.

Inspired by the computing efficiency of SNNs we implemented the proposed network on Raspberry PI, in which it reads sensory data, execute the inference, and show the result on a Graphical Liquid Crystal Display. The execution time for classifying one object shape after pressing in real-time was equal to 0.421875s.

Chapter 6

Conclusion and Future Work

6.1 Conclusion

The core idea presented in this dissertation represents a paradigm shift in the approach towards representing and processing of tactile signals. This thesis presents a biologically inspired, spiking models architecture that mimic human tactile perception with computational primitives that are implementable on low-power subthreshold neuromorphic hardware. More specifically, the model architecture aimed to solve the problem of stimulus orientation detection, touch modalities classification, and object contact shape classification. The model consists of a necessary sensory array integrated into a spiking neural network of leaky integrate and fire neurons. Inspired by biology, I hypothesized that a spatiotemporal representation of tactile signals will be a powerful technique for the efficient encoding of contact stimuli. Following this hypothesis, we used the analog

signals measured by the sensors as input to our spiking neural network that processes and converts them into neuromorphic spikes.

In artificial sensory systems, feature extraction is an important stage to extract information from the sensory signal. Edge orientation detection could be seen as the basis for contour following making it considered as one of the most feature extraction for detecting the shape of an object. In literature, edge orientation selectivity was solved by concentrating on the biological aspect more than the computational efficiency [113]. Furthermore, the methodology used by researchers through their studies [115, 114], lack the possibility to be embedded on robots or neuromorphic chips, due to the need for offline learning and the presence of structures not easily transferable in silicon. In such situations, where energy and space are major constraints, a hardware implementation with online learning and low power devices is usually preferred. Such a methodology enables the system to perform end-to-end computation from the sensors to the processing and classification, consuming low power.

Inspired by the biological finding of Pruszynski et al. (2014) [19], we validated the developed network architecture in discriminating different edge orientations manually applied on the artificial skin from the iCub robot by designing a structure of overlapping interleaved receptive fields. The network is capable of discriminating between eight different orientations by adopting a model of overlapping and interleaved receptive fields that exploits temporal coincidence of the activation of neurons with different receptive fields. Then, following biologically examples [108] in which sensitive areas of the receptive fields are randomly distributed on the skin, we proposed an approach for

estimating the optimal distribution of sensitive areas in receptive fields for increasing the orientation acuity. We demonstrated that the receptive fields created by randomly selecting sensitive points perform better than structured receptive fields with uniform distribution in discriminating small angles (down to 5° with edge length equal to 11 cm), as well as the orientation discrimination, gracefully degrades with decreasing stimulus length (up to 60° with edge length equal to 1cm). Moreover, we show the robustness of the model to edge orientation encoding when the receptive field's density decreases, where nine sensors connected to each receptive field with random distribution were enough to encode eight different stimulus orientations applied on the skin patch. This method can be used as a tool to minimize the connectivity and number of required taxels in future generation of artificial skin, while maintaining spatial sensitivity.

Thanks to the encoding, based on temporal coincidence, the network also has the ability to build up the required connectivity to discriminate between different orientations, when endowed with spike-based unsupervised learning (Spike Driven Synaptic Plasticity (SDSP) Learning rules). The SDSP learning rule responds to Spatio-temporal spikes patterns with temporal coincidence, where it is implementable on neuromorphic subthreshold CMOS technology. Furthermore, we hypothesized that the unconstrained and random structure of the connectivity among layers can produce unbalanced activity in the output neurons, that are driven by a different number of synaptic inputs, as well as the change in pressure in the input stimulus orientation can degrade the performance during inferring. Thus, we deployed synaptic normalization mechanisms to achieve

robust and adaptive network architecture. It was verified that in the developed network with weights normalization mechanism, the current magnitude of each synapse changed, depending on how many active synapses were connected to the decoder layer neuron. This allows the synaptic weights to change progressively during learning converging to stable connectivity patterns, increasing the spatial acuity of the network. Moreover, the developed system is capable of adapting to changes in sensor pressure using a homeostasis mechanism. Nevertheless, the network is able to discriminate between different orientations with an angular resolution of 5° .

Moving on, we demonstrated that the network developed after modification has the ability to appropriate connectivity patterns for the classification of different touch modalities when endowed with a supervised STDP learning rule. The network integrated with a sensory array consisting of 160 piezo-resistive sensors achieves a total accuracy of 88.3 % in classifying six different touch modalities overcoming similar previous work.

Finally, inspired by the computational efficiency of the spiking neural network we developed a network of two-layer of spiking neurons to distinguish between different object shapes pressed on a sensory array. The network with only two layers of SNN could be considered as one of the simplest networks for solving classification problems since it only relies on the input and output layers. Moreover, replacing the used sensors with event-driven sensors may dramatically decrease the complexity of the network since these sensors could feed directly the output layer. The network with two layers has the ability to build up the required connectivity between input and output layers

when endowed with supervised STDP learning rules. The network achieved a total accuracy of 100% when a time step of 0.1 ms was used for classifying eleven object shapes. Furthermore, inspired by the computing efficiency of SNNs we implemented the proposed network on Raspberry Pi, in which it reads sensory data, executes the inference, and shows the result on a Graphical Liquid Crystal Display. The execution time for classifying one object shape after pressing in real-time was equal to 0.42s. The proposed system overcomes a similar state-of-the-art solution in [91] by increasing the number of sensors (160 tactile sensors compared to 42) and the number of objects to be recognized (eleven objects instead of 3) in [91]. Moreover, the network presented in [91] was intended for simulation on software, while the proposed network has been simulated on software then ported on a Raspberry Pi hardware device.

6.2 Future Work

Development of a robust and efficient tactile sensory system for robotic and prosthetic applications is still a great challenge. This thesis explored several biologically-inspired approaches to address some of existing challenges. However, it is far from achieving a complete solution. There is much to be explored in this domain.

The full potential of a neuromorphic tactile perception system will not be feasible without appropriate event-driven neuromorphic tactile sensors. In this thesis, we utilized a biologically neuromorphic approach in which taxels were scanned passively using a hardware setup and their values were converted into spatiotemporal spike patterns

for processing. This work, and the field of neuromorphic tactile perception in general, can greatly benefit from development and miniaturization of event-driven sensors and their readout circuitry that mimic the behaviour of human mechanoreceptors response, specifically, SA and FA types.

Algorithms to process spatio-temporal feature representations are still in their infancy. In this thesis, we proposed to utilize spiking neural networks for signal processing. However, spiking neural networks simulated on conventional computing hardware will not achieve the real-time performance and low power consumption promised by neuromorphic systems. Ideally, the proposed algorithms should be implemented on multi-purpose event based computing platforms such as the SpiNNaker [191] and Loihi [192] to leverage on the massive parallelism and computational efficiency afforded by these systems.

The multiple receptor types and their varying responses have been assumed to be a form of low level feature extraction, specifically, edge orientation selectivity. We have presented a model that mimic the biological SA-I that appears to improve the efficiency of feature representations and edge orientation selectivity. However, the model only simulates the first-order properties of the mechanoreceptor, and additional refinement is needed to reproduce the more complex characteristics of the afferent. Moreover, it would be prudent to include the simulation of Dynamic SA-I and FA-I receptors to increase biological realism and signal dimensionality. In this thesis, we proposed an approach for estimating the optimal distribution of sensitive areas in receptive fields for increasing the orientation acuity. By combining different types of mechanoreceptors

could increase the orientation acuity, besides it leads to a complex random overlapping receptive fields on the artificial skin. Reward-Modulated STDP could be the neuronal basis for reward learning, and it could be used to learn the distribution of different types of mechanoreceptors along their receptive fields, with the aim of designing compact and efficient sensing devices that can locally pre-process the tactile signal. Eventually, the effectiveness of the neuromorphic approach will only be appreciated by the solving of real-life problems.

Bibliography

- [1] Ravinder S Dahiya, Giorgio Metta, Maurizio Valle, and Giulio Sandini. Tactile sensing—from humans to humanoids. *IEEE transactions on robotics*, 26(1):1–20, 2009. pages 1, 18
- [2] Ravinder S Dahiya, Philipp Mittendorfer, Maurizio Valle, Gordon Cheng, and Vladimir J Lumelsky. Directions toward effective utilization of tactile skin: A review. *IEEE Sensors Journal*, 13(11):4121–4138, 2013. pages 1
- [3] Roland S Johansson and J Randall Flanagan. Coding and use of tactile signals from the fingertips in object manipulation tasks. *Nature Reviews Neuroscience*, 10(5):345–359, 2009. pages 2, 14, 43
- [4] Roland S Johansson and Ake B Vallbo. Tactile sensibility in the human hand: relative and absolute densities of four types of mechanoreceptive units in glabrous skin. *The Journal of physiology*, 286(1):283–300, 1979. pages 14, 16, 59
- [5] Roland S Johansson and Åke B Vallbo. Tactile sensory coding in the glabrous skin of the human hand. *Trends in neurosciences*, 6:27–32, 1983. pages 2, 15
- [6] Rufin VanRullen, Rudy Guyonneau, and Simon J Thorpe. Spike times make sense. *Trends in neurosciences*, 28(1):1–4, 2005. pages 2
- [7] Wolfgang Maass. Networks of spiking neurons: the third generation of neural network models. *Neural networks*, 10(9):1659–1671, 1997. pages 3
- [8] Goran Westling and Roland S Johansson. Factors influencing the force control during precision grip. *Experimental brain research*, 53(2):277–284, 1984. pages 12
- [9] Jan BF van Erp and Hendrik AHC van Veen. Touch down: the effect of artificial touch cues on orientation in microgravity. *Neuroscience letters*, 404(1-2):78–82, 2006. pages 12
- [10] JM Loomis, SJ Lederman, KR Boff, L Kaufman, and JP Thomas. Cognitive processes performances. *Tactual Perceptio*, 2, 1986. pages 13
- [11] Michael SA Graziano and Matthew M Botvinick. How the brain represents the body: insights from neurophysiology and psychology. *Common mechanisms in perception and action: Attention and performance XIX*, 19:136–157, 2002. pages 13

- [12] Chiara Lucarotti, Calogero Maria Oddo, Nicola Vitiello, and Maria Chiara Carrozza. Synthetic and bio-artificial tactile sensing: A review. *Sensors*, 13(2):1435–1466, 2013. pages 13
- [13] Javad Dargahi, Saeed Sokhanvar, Siamak Najarian, and Siamak Arbatani. *Tactile sensing and displays: haptic feedback for minimally invasive surgery and robotics*. John Wiley & Sons, 2012. pages 16
- [14] E Bruce Goldstein and James Brockmole. *Sensation and perception*. Cengage Learning, 2016. pages 17
- [15] Robert E Fyffe, Surindar S Cheema, and ALDO Rustioni. Intracellular staining study of the feline cuneate nucleus. i. terminal patterns of primary afferent fibers. *Journal of Neurophysiology*, 56(5):1268–1283, 1986. pages 17
- [16] Kenneth D Cliffer and William D Willis. Distribution of the postsynaptic dorsal column projection in the cuneate nucleus of monkeys. *Journal of Comparative Neurology*, 345(1):84–93, 1994. pages 17
- [17] Hannes P Saal and Sliman J Bensmaia. Touch is a team effort: interplay of submodalities in cutaneous sensibility. *Trends in neurosciences*, 37(12):689–697, 2014. pages 17
- [18] Henrik Jörntell, Fredrik Bengtsson, Pontus Geborek, Anton Spanne, Alexander V Terekhov, and Vincent Hayward. Segregation of tactile input features in neurons of the cuneate nucleus. *Neuron*, 83(6):1444–1452, 2014. pages 17, 18
- [19] J Andrew Pruszynski and Roland S Johansson. Edge-orientation processing in first-order tactile neurons. *Nature neuroscience*, 17(10):1404–1409, 2014. pages 18, 31, 32, 48, 57, 58, 60, 63, 118
- [20] Chiara Bartolozzi, Lorenzo Natale, Francesco Nori, and Giorgio Metta. Robots with a sense of touch. *Nature materials*, 15(9):921–925, 2016. pages 18, 21
- [21] Zexiang Li, Ping Hsu, and Shankar Sastry. Grasping and coordinated manipulation by a multifingered robot hand. *The International Journal of Robotics Research*, 8(4):33–50, 1989. pages 19
- [22] Robert D Howe and Mark R Cutkosky. Integrating tactile sensing with control for dextrous manipulation. In *Proc. IEEE Int. Workshop Intell. Motion Control*, volume 1, pages 369–374, 1990. pages
- [23] Alan D Berger and Pradeep K Khosla. Using tactile data for real-time feedback. *The International journal of robotics research*, 10(2):88–102, 1991. pages 19
- [24] Ronald S Fearing. Tactile sensing mechanisms. *The International Journal of Robotics Research*, 9(3):3–23, 1990. pages 19
- [25] Antonio Bicchi, J Kenneth Salisbury, and Paolo Dario. Augmentation of grasp robustness using intrinsic tactile sensing. In *1989 IEEE International Conference on Robotics and Automation*, pages 302–303. IEEE Computer Society, 1989. pages 19

- [26] Mark R Cutkosky and Imin Kao. Computing and controlling compliance of a robotic hand. *IEEE transactions on robotics and automation*, 5(2):151–165, 1989. pages 19
- [27] Richard M Murray, Zexiang Li, and S Shankar Sastry. *A mathematical introduction to robotic manipulation*. CRC press, 2017. pages 19
- [28] Claudio Domenici, D De Rossi, Adolfo Bacci, and Stefano Bennati. Shear stress detection in an elastic layer by a piezoelectric polymer tactile sensor. *IEEE transactions on electrical insulation*, 24(6):1077–1081, 1989. pages 19
- [29] D De Rossi, G Canepa, G Magenes, F Germagnoli, A Caiti, and Thomas Parisini. Skin-like tactile sensor arrays for contact stress field extraction. *Materials Science and Engineering: C*, 1(1):23–36, 1993. pages 19
- [30] Peer A Schmidt, Eric Maël, and Rolf P Würtz. A sensor for dynamic tactile information with applications in human–robot interaction and object exploration. *Robotics and Autonomous Systems*, 54(12):1005–1014, 2006. pages 19
- [31] Robert D Howe and Mark R Cutkosky. Dynamic tactile sensing: Perception of fine surface features with stress rate sensing. *IEEE transactions on robotics and automation*, 9(2):140–151, 1993. pages 19
- [32] Daisuke Yamada, Takashi Maeno, and Yoji Yamada. Artificial finger skin having ridges and distributed tactile sensors used for grasp force control. In *Proceedings 2001 IEEE/RSJ International Conference on Intelligent Robots and Systems. Expanding the Societal Role of Robotics in the the Next Millennium (Cat. No. 01CH37180)*, volume 2, pages 686–691. IEEE, 2001. pages 19
- [33] Hitoshi Maekawa, Kazuo Tanie, Kiyoshi Komoriya, Makoto Kaneko, Chiyoharu Horiguchi, and Takao Sugawara. Development of a finger-shaped tactile sensor and its evaluation by active touch. In *Proceedings 1992 IEEE International Conference on Robotics and Automation*, pages 1327–1328. IEEE Computer Society, 1992. pages 19
- [34] Jonathan Rossiter and Toshiharu Mukai. An led-based tactile sensor for multi-sensing over large areas. In *SENSORS, 2006 IEEE*, pages 835–838. IEEE, 2006. pages 19
- [35] Vivek Maheshwari and Ravi F Saraf. High-resolution thin-film device to sense texture by touch. *Science*, 312(5779):1501–1504, 2006. pages 19
- [36] Richard Crowder. Toward robots that can sense texture by touch. *Science*, 312(5779):1478–1479, 2006. pages 20
- [37] Sadao Omata, Yoshinobu Murayama, and Christos E Constantinou. Real time robotic tactile sensor system for the determination of the physical properties of biomaterials. *Sensors and Actuators A: Physical*, 112(2-3):278–285, 2004. pages 20

- [38] Mitsuhiro Shikida, Takeshi Shimizu, Kazuo Sato, and Koichi Itoigawa. Active tactile sensor for detecting contact force and hardness of an object. *Sensors and Actuators A: physical*, 103(1-2):213–218, 2003. pages 20
- [39] Takashi Maeno, Tomoyuki Kawamura, and Sen-Chieh Cheng. Friction estimation by pressing an elastic finger-shaped sensor against a surface. *IEEE Transactions on Robotics and Automation*, 20(2):222–228, 2004. pages 20
- [40] Alexander Schmitz, Perla Maiolino, Marco Maggiali, Lorenzo Natale, Giorgio Cannata, and Giorgio Metta. Methods and technologies for the implementation of large-scale robot tactile sensors. *IEEE Transactions on Robotics*, 27(3):389–400, 2011. pages 20
- [41] Perla Maiolino, Marco Maggiali, Giorgio Cannata, Giorgio Metta, and Lorenzo Natale. A flexible and robust large scale capacitive tactile system for robots. *IEEE Sensors Journal*, 13(10):3910–3917, 2013. pages 21, 25, 26
- [42] Makoto Shimojo, Akio Namiki, Masatoshi Ishikawa, Ryota Makino, and Kunihiro Mabuchi. A tactile sensor sheet using pressure conductive rubber with electrical-wires stitched method. *IEEE Sensors journal*, 4(5):589–596, 2004. pages 21
- [43] Hanna Yousef, Mehdi Boukallel, and Kaspar Althoefer. Tactile sensing for dexterous in-hand manipulation in robotics—a review. *Sensors and Actuators A: physical*, 167(2):171–187, 2011. pages 21
- [44] Craig Chorley, Chris Melhuish, Tony Pipe, and Jonathan Rossiter. Development of a tactile sensor based on biologically inspired edge encoding. In *2009 International Conference on Advanced Robotics*, pages 1–6. IEEE, 2009. pages 22
- [45] Yoshiyuki Ohmura, Yasuo Kuniyoshi, and Akihiko Nagakubo. Conformable and scalable tactile sensor skin for curved surfaces. In *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.*, pages 1348–1353. IEEE, 2006. pages 22
- [46] Rui Li and Edward H Adelson. Sensing and recognizing surface textures using a gelsight sensor. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1241–1247, 2013. pages 22
- [47] Yoshiyuki Ohmura and Yasuo Kuniyoshi. Humanoid robot which can lift a 30kg box by whole body contact and tactile feedback. In *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1136–1141. IEEE, 2007. pages 22
- [48] Masahiro Ohka, Hiroaki Kobayashi, Jumpei Takata, and Yasunaga Mitsuya. Sensing precision of an optical three-axis tactile sensor for a robotic finger. In *ROMAN 2006-The 15th IEEE International Symposium on Robot and Human Interactive Communication*, pages 214–219. IEEE, 2006. pages 22
- [49] Luca Ascari, Paolo Corradi, Lucia Beccai, and Cecilia Laschi. A miniaturized and flexible optoelectronic sensing system for tactile skin. *Journal of Micromechanics and Microengineering*, 17(11):2288, 2007. pages 22

- [50] Ravinder S Dahiya, Giorgio Metta, Maurizio Valle, Leandro Lorenzelli, and Andrea Adami. Piezo-polymer-fet devices based tactile sensors for humanoid robots. In *Sensors and Microsystems*, pages 369–372. Springer, 2010. pages 23
- [51] Khaled S Ramadan, Dan Sameoto, and Sthephane Evoy. A review of piezoelectric polymers as functional materials for electromechanical transducers. *Smart Materials and Structures*, 23(3):033001, 2014. pages 23
- [52] T Nelson, S Jin, S Hackwood, G Beni, et al. Shear-sensitive magnetoresistive robotic tactile sensor. *IEEE Transactions on Magnetics*, 22(5):394–396, 1986. pages 23
- [53] Satoru Takenawa. A magnetic type tactile sensor using a two-dimensional array of inductors. In *2009 IEEE International Conference on Robotics and Automation*, pages 3295–3300. IEEE, 2009. pages 23, 24
- [54] Nicholas Wettels, Veronica J Santos, Roland S Johansson, and Gerald E Loeb. Biomimetic tactile sensor array. *Advanced Robotics*, 22(8):829–849, 2008. pages 24
- [55] Janine Hoelscher, Jan Peters, and Tucker Hermans. Evaluation of tactile feature extraction for interactive object recognition. In *2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids)*, pages 310–317. IEEE, 2015. pages 25
- [56] Filipe Veiga, Herke Van Hoof, Jan Peters, and Tucker Hermans. Stabilizing novel objects by learning to predict tactile slip. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5065–5072. IEEE, 2015. pages 25
- [57] Miao Li, Yasemin Bekiroglu, Danica Kragic, and Aude Billard. Learning of grasp adaptation through experience and tactile sensing. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3339–3346. Ieee, 2014. pages 25
- [58] Philipp Mittendorfer and Gordon Cheng. Open-loop self-calibration of articulated robots with artificial skins. In *2012 IEEE International Conference on Robotics and Automation*, pages 4539–4545. IEEE, 2012. pages 25
- [59] Nathan F Lepora, Uriel Martinez-Hernandez, Mathew Evans, Lorenzo Natale, Giorgio Metta, and Tony J Prescott. Tactile superresolution and biomimetic hyperacuity. *IEEE Transactions on Robotics*, 31(3):605–618, 2015. pages 26
- [60] Joseph L Betthausen, Christopher L Hunt, Luke E Osborn, Rahul R Kaliki, and Nitish V Thakor. Limb-position robust classification of myoelectric signals for prosthesis control using sparse representations. In *2016 38th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pages 6373–6376. IEEE, 2016. pages 26
- [61] Darryl PJ Cotton, Paul H Chappell, Andy Cranny, Neil M White, and Steve P Beeby. A novel thick-film piezoelectric slip sensor for a prosthetic hand. *IEEE sensors journal*, 7(5):752–761, 2007. pages 38

- [62] Andy Cranny, Darryl PJ Cotton, Paul H Chappell, SP Beeby, and NM White. Thick-film force and slip sensors for a prosthetic hand. *Sensors and Actuators A: Physical*, 123:162–171, 2005. pages 26
- [63] Paolo Gastaldo, Luigi Pinna, Lucia Seminara, Maurizio Valle, and Rodolfo Zunino. A tensor-based approach to touch modality classification by using machine learning. *Robotics and Autonomous Systems*, 63:268–278, 2015. pages 27
- [64] M Talbot, M Arvandi, and A Sadeghian. A neural network based surface roughness discrimination algorithm. In *2008 World Automation Congress*, pages 1–8. IEEE, 2008. pages 27, 28
- [65] Sergio Decherchi, Paolo Gastaldo, Ravinder S Dahiya, Maurizio Valle, and Rodolfo Zunino. Tactile-data classification of contact materials using computational intelligence. *IEEE Transactions on Robotics*, 27(3):635–639, 2011. pages 27
- [66] Alin Drimus, Gert Kootstra, Arne Bilberg, and Danica Kragic. Design of a flexible tactile sensor for classification of rigid and deformable objects. *Robotics and Autonomous Systems*, 62(1):3–15, 2014. pages 27
- [67] Calogero M Oddo, Marco Controzzi, Lucia Beccai, Christian Cipriani, and Maria Chiara Carrozza. Roughness encoding for discrimination of surfaces in artificial active-touch. *IEEE Transactions on Robotics*, 27(3):522–533, 2011. pages 27
- [68] Zachary Pezzementi, Erion Plaku, Caitlin Reyda, and Gregory D Hager. Tactile-object recognition from appearance information. *IEEE Transactions on Robotics*, 27(3):473–487, 2011. pages 27
- [69] Danfei Xu, Gerald E Loeb, and Jeremy A Fishel. Tactile identification of objects using bayesian exploration. In *2013 IEEE International Conference on Robotics and Automation*, pages 3056–3061. IEEE, 2013. pages 28
- [70] Tapomayukh Bhattacharjee, James M Rehg, and Charles C Kemp. Haptic classification and recognition of objects using a tactile sensing forearm. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4090–4097. IEEE, 2012. pages 28
- [71] Jeremy A Fishel and Gerald E Loeb. Bayesian exploration for intelligent identification of textures. *Frontiers in neurorobotics*, 6:4, 2012. pages 28
- [72] Zhe Su, Jeremy A Fishel, Tomonori Yamamoto, and Gerald E Loeb. Use of tactile feedback to control exploratory movements to characterize object compliance. *Frontiers in neurorobotics*, 6:7, 2012. pages 28
- [73] Hiroyasu Iwata and Shigeki Sugano. Human-robot-contact-state identification based on tactile recognition. *IEEE Transactions on Industrial Electronics*, 52(6):1468–1477, 2005. pages 28

- [74] Walter Dan Stiehl and Cynthia Breazeal. Applying a "somatic alphabet" approach to inferring orientation, motion, and direction in clusters of force sensing resistors. In *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (IEEE Cat. No. 04CH37566), volume 3, pages 3015–3020. IEEE, 2004. pages 28
- [75] Futoshi Naya, Junji Yamato, and Kazuhiko Shinozawa. Recognizing human touching behaviors using a haptic interface for a pet-robot. In *IEEE SMC'99 Conference Proceedings. 1999 IEEE International Conference on Systems, Man, and Cybernetics* (Cat. No. 99CH37028), volume 2, pages 1030–1034. IEEE, 1999. pages 28
- [76] David Silvera Tawil, David Rye, and Mari Velonaki. Interpretation of the modality of touch on an artificial arm covered with an eit-based sensitive skin. *The International Journal of Robotics Research*, 31(13):1627–1641, 2012. pages 28
- [77] Nawid Jamali and Claude Sammut. Material classification by tactile sensing using surface textures. In *2010 IEEE International Conference on Robotics and Automation*, pages 2336–2341. IEEE, 2010. pages 28
- [78] Abdul Md Mazid and ABM Shawkat Ali. Opto-tactile sensor for surface texture pattern identification using support vector machine. In *2008 10th International Conference on Control, Automation, Robotics and Vision*, pages 1830–1835. IEEE, 2008. pages 28
- [79] Mohamad Alameh, Yahya Abbass, Ali Ibrahim, Gabriele Moser, and Maurizio Valle. Touch modality classification using recurrent neural networks. *IEEE Sensors Journal*, 21(8):9983–9993, 2021. pages 28
- [80] Hongbin Liu, Juan Greco, Xiaojing Song, Joao Bimbo, Lakmal Seneviratne, and Kaspar Althoefer. Tactile image based contact shape recognition using neural network. In *2012 IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI)*, pages 138–143. IEEE, 2012. pages 28
- [81] Emil M Petriu, Stephen KS Yeung, Sunil R Das, A-M Cretu, and Hans JW Spoelder. Robotic tactile recognition of pseudorandom encoded objects. *IEEE Transactions on Instrumentation and Measurement*, 53(5):1425–1432, 2004. pages 28
- [82] Seiji Aoyagi, Takaaki Tanaka, and Mamoru Minami. Recognition of contact state of four layers arrayed type tactile sensor by using neural network. In *2006 IEEE International Conference on Information Acquisition*, pages 393–397. IEEE, 2006. pages 28
- [83] Catherine D Schuman, James S Plank, Adam Disney, and John Reynolds. An evolutionary optimization framework for neural networks and neuromorphic architectures. In *2016 International Joint Conference on Neural Networks (IJCNN)*, pages 145–154. IEEE, 2016. pages 28
- [84] Alexander Schneider, Jürgen Sturm, Cyrill Stachniss, Marco Reisert, Hans Burkhardt, and Wolfram Burgard. Object identification with tactile sensors using bag-of-features. In *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 243–248. IEEE, 2009. pages 28

- [85] Marianna Madry, Liefeng Bo, Danica Kragic, and Dieter Fox. St-hmp: Unsupervised spatio-temporal feature learning for tactile data. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2262–2269. IEEE, 2014. pages 29
- [86] Shan Luo, Wenxuan Mou, Kaspar Althoefer, and Hongbin Liu. Novel tactile-sift descriptor for object shape recognition. *IEEE Sensors Journal*, 15(9):5001–5009, 2015. pages 29
- [87] An-An Liu, Yu-Ting Su, Wei-Zhi Nie, and Mohan Kankanhalli. Hierarchical clustering multi-task learning for joint human action grouping and recognition. *IEEE transactions on pattern analysis and machine intelligence*, 39(1):102–114, 2016. pages 29
- [88] Harold Soh, Yanyu Su, and Yiannis Demiris. Online spatio-temporal gaussian process experts with application to tactile classification. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4489–4496. IEEE, 2012. pages 29
- [89] Tasbolat Taunyazov, Yansong Chua, Ruihan Gao, Harold Soh, and Yan Wu. Fast texture classification using tactile neural coding and spiking neural network. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 9890–9895. IEEE, 2020. pages 29
- [90] Udaya B Rongala, Alberto Mazzoni, Anton Spanne, Henrik Jörntell, and Calogero M Oddo. Cuneate spiking neural network learning to classify naturalistic texture stimuli under varying sensing conditions. *Neural Networks*, 123:273–287, 2020. pages 29
- [91] Jaehun Kim, Sung-Phil Kim, Jungjun Kim, Heeseon Hwang, Jaehyun Kim, Doowon Park, and Unyong Jeong. Object shape recognition using tactile sensor arrays by a spiking neural network with unsupervised learning. In *2020 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pages 178–183. IEEE, 2020. pages 29, 121
- [92] M Pearson, Mokhtar Nibouche, Ian Gilhespy, K Gurney, Chris Melhuish, Benjamin Mitchinson, and Anthony G Pipe. A hardware based implementation of a tactile sensory system for neuromorphic signal processing applications. In *2006 IEEE International Conference on Acoustics Speech and Signal Processing Proceedings*, volume 4, pages IV–IV. IEEE, 2006. pages 29
- [93] Martin J Pearson, Anthony G Pipe, Benjamin Mitchinson, Kevin Gurney, Chris Melhuish, Ian Gilhespy, and Mokhtar Nibouche. Implementing spiking neural networks for real-time signal-processing and control applications: A model-validated fpga approach. *IEEE Transactions on Neural Networks*, 18(5):1472–1487, 2007. pages 30
- [94] Martin J Pearson, Ben Mitchinson, J Charles Sullivan, Anthony G Pipe, and Tony J Prescott. Biomimetic vibrissal sensing for robots. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 366(1581):3085–3096, 2011. pages 30

- [95] Sung Soo Kim, Arun P Sripati, R Jacob Vogelstein, Robert S Armiger, Alexander F Russell, and Sliman J Bensmaia. Conveying tactile feedback in sensorized hand neuroprostheses using a biofidelic model of mechanotransduction. *IEEE transactions on biomedical circuits and systems*, 3(6):398–404, 2009. pages 30
- [96] Elmer K Kim, Scott A Wellnitz, Sarah M Bourdon, Ellen A Lumpkin, and Gregory J Gerling. Force sensor in simulated skin and neural model mimic tactile sai afferent spiking response to ramp and hold stimuli. *Journal of neuroengineering and rehabilitation*, 9(1):1–14, 2012. pages 30
- [97] Luca Leonardo Bologna, Jérémie Pinoteau, Romain Brasselet, Marco Maggiali, and Angelo Arleo. Encoding/decoding of first and second order tactile afferents in a neurobotic application. *Journal of Physiology-Paris*, 105(1-3):25–35, 2011. pages 30
- [98] Luca Leonardo Bologna, Jérémie Pinoteau, Jesús Garrido, and Angelo Arleo. Active tactile sensing in a neurobotic braille-reading system. In *2012 4th IEEE RAS & EMBS International Conference on Biomedical Robotics and Biomechatronics (BioRob)*, pages 1925–1930. IEEE, 2012. pages 30
- [99] LL Bologna, J Pinoteau, JB Passot, JA Garrido, Jörn Vogel, E Ros Vidal, and A Arleo. A closed-loop neurobotic system for fine touch sensing. *Journal of neural engineering*, 10(4):046019, 2013. pages 30
- [100] Giacomo Spigler, Calogero M Oddo, and Maria Chiara Carrozza. Soft-neuromorphic artificial touch for applications in neuro-robotics. In *2012 4th IEEE RAS & EMBS international conference on biomedical robotics and biomechatronics (BioRob)*, pages 1913–1918. IEEE, 2012. pages 30
- [101] WW Lee, J Cabibihan, and NV Thakor. Bio-mimetic strategies for tactile sensing. In *SENSORS, 2013 IEEE*, pages 1–4. IEEE, 2013. pages 30, 84
- [102] Udaya Bhaskar Rongala, Alberto Mazzoni, and Calogero Maria Oddo. Neuromorphic artificial touch for categorization of naturalistic textures. *IEEE transactions on neural networks and learning systems*, 28(4):819–829, 2015. pages 30, 38
- [103] Ken E Friedl, Aaron R Voelker, Angelika Peer, and Chris Eliasmith. Human-inspired neurobotic system for classifying surface textures by touch. *IEEE Robotics and Automation Letters*, 1(1):516–523, 2016. pages 30
- [104] Chris Eliasmith and Charles H Anderson. *Neural engineering: Computation, representation, and dynamics in neurobiological systems*. MIT press, 2003. pages 30
- [105] Steve B Furber, David R Lester, Luis A Plana, Jim D Garside, Eustace Painkras, Steve Temple, and Andrew D Brown. Overview of the spinnaker system architecture. *IEEE Transactions on Computers*, 62(12):2454–2467, 2012. pages 31

- [106] Luke E Osborn, Andrei Dragomir, Joseph L Betthausen, Christopher L Hunt, Harrison H Nguyen, Rahul R Kaliki, and Nitish V Thakor. Prosthesis with neuromorphic multilayered e-dermis perceives touch and pain. *Science robotics*, 3(19), 2018. pages 31
- [107] Calogero Maria Oddo, Stanisa Raspopovic, Fiorenzo Artoni, Alberto Mazzoni, Giacomo Spigler, Francesco Petrini, Federica Giambattistelli, Fabrizio Vecchio, Francesca Miraglia, Loredana Zollo, et al. Intraneural stimulation elicits discrimination of textural features by artificial fingertip in intact and amputee humans. *elife*, 5:e09148, 2016. pages 31
- [108] J Andrew Pruszynski, J Randall Flanagan, and Roland S Johansson. Fast and accurate edge orientation processing during object manipulation. *Elife*, 7:e31200, 2018. pages 31, 32, 49, 58, 59, 60, 63, 67, 79, 118
- [109] Roland S Johansson. Tactile sensibility in the human hand: receptive field characteristics of mechanoreceptive units in the glabrous skin area. *The Journal of physiology*, 281(1):101–125, 1978. pages 32, 58
- [110] Alison I Weber, Hannes P Saal, Justin D Lieber, Ju-Wen Cheng, Louise R Manfredi, John F Dammann, and Sliman J Bensmaia. Spatial and temporal codes mediate the tactile perception of natural textures. *Proceedings of the National Academy of Sciences*, 110(42):17107–17112, 2013. pages 32
- [111] Sliman J Bensmaia, Peter V Denchev, J Francis Dammann, James C Craig, and Steven S Hsiao. The representation of stimulus orientation in the early stages of somatosensory processing. *Journal of Neuroscience*, 28(3):776–786, 2008. pages 32, 57
- [112] Etay Hay and J Andrew Pruszynski. Orientation processing by synaptic integration across first-order tactile neurons. *PLoS computational biology*, 16(12):e1008303, 2020. pages 32
- [113] Adel Parvizi-Fard, Mahmood Amiri, Deepesh Kumar, Mark M Iskarous, and Nitish V Thakor. A functional spiking neuronal network for tactile sensing pathway to process edge orientation. *Scientific reports*, 11(1):1–16, 2021. pages 32, 40, 118
- [114] Ruben D Ponce Wong, Randall B Hellman, and Veronica J Santos. Spatial asymmetry in tactile sensor skin deformation aids perception of edge orientation during haptic exploration. *IEEE transactions on haptics*, 7(2):191–202, 2013. pages 33, 118
- [115] Udaya Bhaskar Rongala, Alberto Mazzoni, Marcello Chiurazzi, Domenico Camboni, Mario Milazzo, Luca Massari, Gastone Ciuti, Stefano Roccella, Paolo Dario, and Calogero Maria Oddo. Tactile decoding of edge orientation with artificial cuneate neurons in dynamic conditions. *Frontiers in neurorobotics*, 13:44, 2019. pages 33, 118

- [116] Uriel Martinez-Hernandez, Giorgio Metta, Tony J Dodd, Tony J Prescott, Lorenzo Natale, and Nathan F Lepora. Active contour following to explore object shape with robot touch. In *2013 World Haptics Conference (WHC)*, pages 341–346. IEEE, 2013. pages 33
- [117] Deepesh Kumar, Rohan Ghosh, Andrei Nakagawa-Silva, Alcimar B Soares, and Nitish V Thakor. Neuromorphic approach to tactile edge orientation estimation using spatiotemporal similarity. *Neurocomputing*, 407:246–258, 2020. pages 33
- [118] Kenneth S Cole and Howard J Curtis. Electrical impedance of nerve during activity. *Nature*, 142(3587):209–210, 1938. pages 35
- [119] Shih-Chii Liu and Tobi Delbruck. Neuromorphic sensory systems. *Current opinion in neurobiology*, 20(3):288–295, 2010. pages 37, 38
- [120] Carver Mead. Neuromorphic electronic systems. *Proceedings of the IEEE*, 78(10):1629–1636, 1990. pages 37
- [121] Nathan F Lepora and Benjamin Ward-Cherrier. Superresolution with an optical tactile sensor. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2686–2691. IEEE, 2015. pages 38
- [122] Mohsin I Tiwana, Arridh Shashank, Stephen J Redmond, and Nigel H Lovell. Characterization of a capacitive tactile shear sensor for application in robotic and upper limb prostheses. *Sensors and Actuators A: physical*, 165(2):164–172, 2011. pages 38
- [123] Stefano Caviglia, Luigi Pinna, Maurizio Valle, and Chiara Bartolozzi. Spike-based readout of posfet tactile sensors. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 64(6):1421–1431, 2016. pages 38
- [124] Mahdi Rasouli, Yi Chen, Arindam Basu, Sunil L Kukreja, and Nitish V Thakor. An extreme learning machine-based neuromorphic tactile sensing system for texture recognition. *IEEE transactions on biomedical circuits and systems*, 12(2):313–325, 2018. pages 38
- [125] Chiara Bartolozzi, Paolo Motto Ros, Francesco Diotalevi, Nawid Jamali, Lorenzo Natale, Marco Crepaldi, and Danilo Demarchi. Event-driven encoding of off-the-shelf tactile sensors for compression and latency optimisation for robotic skin. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 166–173. IEEE, 2017. pages 38
- [126] Zhengkun Yi, Yilei Zhang, and Jan Peters. Biomimetic tactile sensors and signal processing with spike trains: A review. *Sensors and Actuators A: Physical*, 269:41–52, 2018. pages 38
- [127] Kenneth O Johnson, Takashi Yoshioka, and Francisco Vega-Bermudez. Tactile functions of mechanoreceptive afferents innervating the hand. *Journal of Clinical Neurophysiology*, 17(6):539–558, 2000. pages 40

- [128] Hannes P Saal and Sliman J Bensmaia. Biomimetic approaches to bionic touch through a peripheral nerve interface. *Neuropsychologia*, 79:344–353, 2015. pages 40
- [129] Richard B Stein. Some models of neuronal variability. *Biophysical journal*, 7(1):37–68, 1967. pages 40
- [130] Eugene M Izhikevich. Simple model of spiking neurons. *IEEE Transactions on neural networks*, 14(6):1569–1572, 2003. pages 40, 41
- [131] AL Hodgkin. A quantitative description of ion currents and its application and excitation in nerve membranes. *Journal of Physiology*, 117:500–544, 1952. pages 41
- [132] Richard B Stein. A theoretical analysis of neuronal variability. *Biophysical Journal*, 5(2):173–194, 1965. pages 41
- [133] Eugene M Izhikevich. Neural excitability, spiking and bursting. *International journal of bifurcation and chaos*, 10(06):1171–1266, 2000. pages 42
- [134] Victoria E. Abraira and David D. Ginty. The sensory neurons of touch. *Neuron*, 79(4):618–639, Aug 2013. pages 42
- [135] Marcel Stimberg, Romain Brette, and Dan FM Goodman. Brian 2, an intuitive and efficient neural simulator. *eLife*, 8:e47314, August 2019. pages 42
- [136] Elisabetta Chicca, Giacomo Indiveri, and Rodney J Douglas. An event-based vlsi network of integrate-and-fire neurons. In *2004 IEEE International Symposium on Circuits and Systems (IEEE Cat. No. 04CH37512)*, volume 5, pages V–357. IEEE, 2004. pages 44, 45
- [137] Jinling Wang, Ammar Belatreche, Liam Maguire, and Thomas Martin McGinnity. An online supervised learning method for spiking neural networks with adaptive structure. *Neurocomputing*, 144:526–536, 2014. pages 45
- [138] Giorgio Metta, Lorenzo Natale, Francesco Nori, Giulio Sandini, David Vernon, Luciano Fadiga, Claes Von Hofsten, Kerstin Rosander, Manuel Lopes, José Santos-Victor, et al. The icub humanoid robot: An open-systems platform for research in cognitive development. *Neural networks*, 23(8-9):1125–1134, 2010. pages 45, 48
- [139] Lorenzo Natale, Ali Paikan, Marco Randazzo, and Daniele E Domenichelli. The icub software architecture: evolution and lessons learned. *Frontiers in Robotics and AI*, 3:24, 2016. pages 46
- [140] Jeffrey M Yau, Anitha Pasupathy, Paul J Fitzgerald, Steven S Hsiao, and Charles E Connor. Analogous intermediate shape coding in vision and touch. *Proceedings of the National Academy of Sciences*, 106(38):16457–16462, 2009. pages 57
- [141] Paul J Fitzgerald, John W Lane, Pramodsingh H Thakur, and Steven S Hsiao. Receptive field properties of the macaque second somatosensory cortex: representation of orientation on different finger pads. *Journal of Neuroscience*, 26(24):6473–6484, 2006. pages

- [142] Robert Shapley, Michael Hawken, and Dario L Ringach. Dynamics of orientation selectivity in the primary visual cortex and the importance of cortical inhibition. *Neuron*, 38(5):689–699, 2003. pages
- [143] David H Hubel and Torsten N Wiesel. Receptive fields and functional architecture of monkey striate cortex. *The Journal of physiology*, 195(1):215–243, 1968. pages 57
- [144] Steven Hsiao. Central mechanisms of tactile shape perception. *Current opinion in neurobiology*, 18(4):418–424, 2008. pages 57
- [145] David Ferster and Kenneth D Miller. Neural mechanisms of orientation selectivity in the visual cortex. *Annual review of neuroscience*, 23(1):441–471, 2000. pages 57
- [146] Tim Gollisch and Markus Meister. Eye smarter than scientists believed: neural computations in circuits of the retina. *Neuron*, 65(2):150–164, 2010. pages 57
- [147] Sowmya Venkataramani and W Rowland Taylor. Orientation selectivity in rabbit retinal ganglion cells is mediated by presynaptic inhibition. *Journal of Neuroscience*, 30(46):15664–15676, 2010. pages 57
- [148] N Cauna. Nerve supply and nerve endings in meissner’s corpuscles. *American Journal of Anatomy*, 99(2):315–350, 1956. pages 57
- [149] Daine R Lesniak, Kara L Marshall, Scott A Wellnitz, Blair A Jenkins, Yoshichika Baba, Matthew N Rasband, Gregory J Gerling, and Ellen A Lumpkin. Computation identifies structural features that govern neuronal firing properties in slowly adapting touch receptors. *Elife*, 3:e01488, 2014. pages 57
- [150] Jeffrey L Gauthier, Greg D Field, Alexander Sher, Martin Greschner, Jonathon Shlens, Alan M Litke, and EJ Chichilnisky. Receptive fields in primate retina are coordinated to sample visual space more uniformly. *PLoS biology*, 7(4):e1000063, 2009. pages 58
- [151] JR Phillips, Roland S Johansson, and Kenneth O Johnson. Responses of human mechanoreceptive afferents to embossed dot arrays scanned across fingerpad skin. *Journal of neuroscience*, 12(3):827–839, 1992. pages 58
- [152] Melvyn D Goldfinger. Random-sequence stimulation of the g1 hair afferent unit. *Somatosensory & motor research*, 7(1):19–45, 1990. pages 58
- [153] Choonseog Park, Yoon H. Bai, and Yoonsuck Choe. Tactile or visual?: Stimulus characteristics determine receptive field type in a self-organizing map model of cortical development. In *2009 IEEE Symposium on Computational Intelligence for Multimedia Signal and Vision Processing*, page 6–13, Mar 2009. pages 58
- [154] Charlie W Zhao, Mark J Daley, and J Andrew Pruszynski. Neural network models of the tactile system develop first-order units with spatially complex receptive fields. *PloS one*, 13(6):e0199196, 2018. pages 58, 64

- [155] Roland S Johansson and Ingvars Birznieks. First spikes in ensembles of human tactile afferents code complex spatial fingertip events. *Nature neuroscience*, 7(2):170–177, 2004. pages 59
- [156] Garrett B Stanley. Reading and writing the neural code. *Nature neuroscience*, 16(3):259–263, 2013. pages 59
- [157] AB Vallbo. Properties of cutaneous mechano receptors in the human hand related to touch sensation, *human neuro biology*, 3, 1973. pages 59
- [158] Nicholas M Timme and Christopher Lapish. A tutorial for information theory in neuroscience. *eneuro*, 5(3), 2018. pages 68
- [159] Wulfram Gerstner and Werner M Kistler. *Spiking neuron models: Single neurons, populations, plasticity*. Cambridge university press, 2002. pages 81
- [160] Michael N Shadlen and William T Newsome. Noise, neural codes and cortical organization. *Current opinion in neurobiology*, 4(4):569–579, 1994. pages 81
- [161] Rufin VanRullen, Rudy Guyonneau, and Simon J Thorpe. Spike times make sense. *Trends in neurosciences*, 28(1):1–4, 2005. pages 81
- [162] Wolfgang Maass and Henry Markram. On the computational power of circuits of spiking neurons. *Journal of computer and system sciences*, 69(4):593–616, 2004. pages 82
- [163] Wolfgang Maass. Networks of spiking neurons: the third generation of neural network models. *Neural networks*, 10(9):1659–1671, 1997. pages 82
- [164] Robert Gütig and Haim Sompolinsky. The tempotron: a neuron that learns spike timing–based decisions. *Nature neuroscience*, 9(3):420–428, 2006. pages 82
- [165] Luca Leonardo Bologna, Jérémie Pinoteau, Romain Brasselet, Marco Maggiali, and Angelo Arleo. Encoding/decoding of first and second order tactile afferents in a neurorobotic application. *Journal of Physiology-Paris*, 105(1-3):25–35, 2011. pages 82
- [166] Robert Gütig, Tim Gollisch, Haim Sompolinsky, and Markus Meister. Computing complex visual features with retinal spike times. *PLoS One*, 8(1):e53063, 2013. pages 82
- [167] Bernhard Nessler, Michael Pfeiffer, and Wolfgang Maass. Stdp enables spiking neurons to detect hidden causes of their inputs. *Advances in neural information processing systems*, 22:1357–1365, 2009. pages 82
- [168] Robert Legenstein, Dejan Pecevski, and Wolfgang Maass. A learning theory for reward-modulated spike-timing-dependent plasticity with application to biofeedback. *PLoS computational biology*, 4(10):e1000180, 2008. pages 82
- [169] Benjamin Schrauwen, Michiel D’Haene, David Verstraeten, and Jan Van Campenhout. Compact hardware liquid state machines on fpga for real-time speech recognition. *Neural networks*, 21(2-3):511–523, 2008. pages 82

- [170] Kenneth L Rice, Mohammad A Bhuiyan, Tarek M Taha, Christopher N Vutsinas, and Melissa C Smith. Fpga implementation of izhikevich spiking neural networks for character recognition. In *2009 International Conference on Reconfigurable Computing and FPGAs*, pages 451–456. IEEE, 2009. pages 82
- [171] Robert Gütig. To spike, or when to spike? *Current opinion in neurobiology*, 25:134–139, 2014. pages 82
- [172] Guo-qiang Bi and Mu-ming Poo. Synaptic modification by correlated activity: Hebb’s postulate revisited. *Annual review of neuroscience*, 24(1):139–166, 2001. pages 84
- [173] Guo-qiang Bi and Mu-ming Poo. Distributed synaptic modification in neural networks induced by patterned stimulation. *Nature*, 401(6755):792–796, 1999. pages 84
- [174] Michael Beyeler, Nikil D Dutt, and Jeffrey L Krichmar. Categorization and decision-making in a neurobiologically plausible spiking network using a stdp-like learning rule. *Neural Networks*, 48:109–124, 2013. pages 84
- [175] TJ Strain, LJ McDaid, LP Maguire, and TM McGinnity. A novel mixed supervised-unsupervised training approach for a spiking neural network classifier. In *Proc. Conf. Intell. Cybern. Syst., SMC UK-RI Chapter*, pages 202–206, 2004. pages 84
- [176] Sander M Bohte, Joost N Kok, and Johannes A La Poutré. Spikeprop: backpropagation for networks of spiking neurons. In *ESANN*, volume 48, pages 419–424. Bruges, 2000. pages 84
- [177] Ammar Belatreche, Liam Maguire, Martin McGinnity, and Qing Wu. A method for supervised training of spiking neural networks. *Cybernetic Intelligence, Challenges and Advances*, pages 11–17, 2003. pages 84
- [178] Bernd Illing, Wulfram Gerstner, and Johanni Brea. Biologically plausible deep learning—but how far can we go with shallow networks? *Neural Networks*, 118:90–101, 2019. pages 84, 85
- [179] Joseph M Brader, Walter Senn, and Stefano Fusi. Learning real-world stimuli in a neural network with spike-driven synaptic dynamics. *Neural computation*, 19(11):2881–2912, 2007. pages 86
- [180] Srinjoy Mitra, Stefano Fusi, and Giacomo Indiveri. Real-time classification of complex patterns using spike-based learning in neuromorphic vlsi. *IEEE transactions on biomedical circuits and systems*, 3(1):32–42, 2008. pages 86
- [181] Charlotte Frenkel, Giacomo Indiveri, Jean-Didier Legat, and David Bol. A fully-synthesized 20-gate digital spike-based synapse with embedded online learning. In *2017 IEEE International Symposium on Circuits and Systems (ISCAS)*, pages 1–4. IEEE, 2017. pages 86
- [182] Hesham Mostafa. Supervised learning based on temporal coding in spiking neural networks. *IEEE transactions on neural networks and learning systems*, 29(7):3227–3235, 2017. pages 95

- [183] Nicholas Soures, Lydia Hays, Eric Bohannon, Abdullah M Ziyarah, and Dhireesha Kudithipudi. On-device stdp and synaptic normalization for neuromemristive spiking neural network. In *2017 IEEE 60th International Midwest Symposium on Circuits and Systems (MWSCAS)*, pages 1081–1084. IEEE, 2017. pages 96
- [184] Gina Turrigiano. Homeostatic signaling: the positive side of negative feedback. *Current opinion in neurobiology*, 17(3):318–324, 2007. pages 100
- [185] Gina G Turrigiano. Homeostatic plasticity in neuronal networks: the more things change, the more they stay the same. *Trends in neurosciences*, 22(5):221–227, 1999. pages 100
- [186] Eve Marder and Astrid A Prinz. Modeling stability in neuron and network function: the role of activity in homeostasis. *Bioessays*, 24(12):1145–1154, 2002. pages
- [187] Julio Echegoyen, Axel Neu, Kevin D Graber, and Ivan Soltesz. Homeostatic plasticity studied using in vivo hippocampal activity-blockade: synaptic scaling, intrinsic plasticity and age-dependence. *PloS one*, 2(8):e700, 2007. pages 100
- [188] Larry F Abbott and Sacha B Nelson. Synaptic plasticity: taming the beast. *Nature neuroscience*, 3(11):1178–1183, 2000. pages 100
- [189] Ning Qiao, Giacomo Indiveri, and Chiara Bartolozzi. Automatic gain control of ultra-low leakage synaptic scaling homeostatic plasticity circuits. In *2016 IEEE Biomedical Circuits and Systems Conference (BioCAS)*, pages 156–159. IEEE, 2016. pages 100
- [190] Jiong Sun, Erik Billing, Fernando Seoane, Bo Zhou, Dan Högberg, and Paul Hemeren. Categories of touch: Classifying human touch using a soft tactile sensor. In *The robotic sense of touch: From sensing to understanding, workshop at the IEEE International Conference on Robotics and Automation (ICRA), Singapore, May 29, 2017*, 2017. pages 106, 107
- [191] Eustace Painkras, Luis A Plana, Jim Garside, Steve Temple, Francesco Galluppi, Cameron Patterson, David R Lester, Andrew D Brown, and Steve B Furber. Spinnaker: A 1-w 18-core system-on-chip for massively-parallel neural network simulation. *IEEE Journal of Solid-State Circuits*, 48(8):1943–1953, 2013. pages 122
- [192] Mike Davies, Narayan Srinivasa, Tsung-Han Lin, Gautham Chinya, Yongqiang Cao, Sri Harsha Choday, Georgios Dimou, Prasad Joshi, Nabil Imam, Shweta Jain, et al. Loihi: A neuromorphic manycore processor with on-chip learning. *Ieee Micro*, 38(1):82–99, 2018. pages 122

Appendix A

SDSP Learning Rule

A.1 SDSP Learning Rule Code

```
"""
SDSP Learning Rule Toy Example

This code is made to demonstarte how the SDSP learning
rule works and the output presented in Nature Journal (In preparation)
Author : Ali Dabbous
Cosmic LAB / DITEN / UNIGE
EDPR LAB / IIT
ali.dabbous@edu.unige.it
ali.m.dabbous@gmail.com
"""

from brian2 import*
import numpy as np
np.random.seed(3000)
stimuli1=np.ones(20)/2
stimuli2=np.ones(15)*1
stimuli3=np.ones(15)/3
stimuli=np.concatenate((stimuli1 , stimuli2 , stimuli3) , axis=0)
stimul3=np.random.shuffle(stimuli)
print(stimuli)
plt.figure()
plot(stimuli)

input_stimuli = TimedArray((stimuli)* nA, dt=10*ms)

L0leaky_iaf = Equations('''
    dv/dt= ((-80*mV) - v + (70*Mohm * input_stimuli(t))
    - ((g_sra)*70*Mohm*(v - (-80*mV))))/(20*ms)
    : volt (unless refractory)
    dg_sra/dt= - g_sra/(1000*ms) : siemens
''')
```

```

training = NeuronGroup(1, model=L0leaky_iaf, threshold="v>(-50.0 * mV)",
reset="v=-80*mV;g_sra +=0*nS",refractory=5*ms, method="euler")
training.v = -75*mV
training.g_sra = 'rand()*0*nS'
v_rest=-70*mV
tau_c=200*ms
Jc=0.1
L2leaky_iaf = Equations(''')
    dv/dt = (v_rest-v)/(10*ms) +
    ((ie*15))/(3*nF) : volt (unless refractory)
    dc/dt = (4.4-c+Jc)/tau_c :1
    dJc/dt=-Jc/(20*ms) :1
    die/dt=-ie/(20*ms):amp
    ''')
layer2 = NeuronGroup(1, model=L2leaky_iaf, threshold="v>(-50.0 * mV)",
reset="v=-70*mV",refractory=5*ms, method="euler")

layer2.v = -80*mV
layer2.c=4
a_plus=0.1*nA
b_minus=0*nA

threshold_v=-65*mV
threshold_l_up=4.2
threshold_l_down=4
threshold_h_up=40
threshold_h_down=4.8
Jc_plus=4
J_minus=0.0000001*pA
J_plus=0.00001*pA
Jc=0.1
w_max=1*nA
w_min=0*nA
J_plus=0.003*nA
J_minus=0.003*nA
threshold_w= 0.5*nA
alpha=0.0035*nA
beta=-0.0035*nA
model_Pre_Post = '''

    dw12/dt=((int(w12 > threshold_w) * (alpha)) +
    ((beta)*int(w12<threshold_w)) )/ (1000*ms) :amp
    ''',

```

```

model_on_pre = '''
    w12+=(a_plus*int(v_post>threshold_v)*int(c_post>threshold_l_up)
        *int(c_post<threshold_h_up))+(-b_minus*int(v_post<threshold_v)
        *int(c_post>threshold_l_down)*int(c_post<threshold_h_down))
    w12=((w12+J_plus)*int(w12>threshold_w))+
        ((w12-J_minus)*int(w12<threshold_w))

    ini=w12

    w12=w_max*int(w12 > w_max) + w12*int(w12 <= w_max)
    *int(w12 > w_min) + w_min*int(w12 < w_min)
    e_post+=ini    ,,,

model_on_post = '''
    Jc+=Jc_plus

    ,,,

Synp12 = Synapses(training , layer2 , model=model_Pre_Post ,
on_pre=model_on_pre , on_post=model_on_post , name='raw')
Synp12.connect()

Synp12.w12=0.4*nA

layer2_membrane=StateMonitor(layer2 , ['v' , 'c'] , record=True)
spLayer0 = SpikeMonitor(training , record=True)
spLayer2 = SpikeMonitor(layer2 , record=True)
stSynp12 = StateMonitor(Synp12 , variables='w12' , record=True)

duration=len(stimuli)*10*ms

run(duration)
spikes_pre=spLayer0.t[spLayer0.i == 0]/ms
plt.figure()
for i in range(len(spikes_pre)):
    plt.vlines(x=spikes_pre[i] , ymin=0 , ymax=3)
plt.axis('off')

import seaborn as sns

spikes_times=spLayer2.t[spLayer2.i == 0]/ms
plt.figure()
plt.plot(layer2_membrane.t/ms , layer2_membrane.v[0]/mV , color='black')

```

```
for i in range(len(spikes_times)):
    plt.vlines(x=spikes_times[i], ymin=-50, ymax=-30)
plt.axhline(-50, linestyle='dashed', color='black')
plt.axhline(-60, linestyle='dashed', color='gray')
plt.yticks([])
plt.xticks([])
sns.despine()
plt.figure()
plt.plot(layer2_membrane.t/ms, layer2_membrane.c[0], color='black')
plt.axhline(4.2, linestyle='dashed', color='gray')
plt.axhline(4.8, linestyle='dashed', color='gray')
plt.axhline(14, linestyle='dashed', color='gray')
plt.yticks([])
plt.xticks([])
sns.despine()

plt.figure()
plt.plot(stSynp12.t/ms, stSynp12[Synp12[0]].w12/nA, color='black')
plt.axhline(0.1, linestyle='dashed', color='gray')
plt.axhline(1, linestyle='dashed', color='gray')
plt.axhline(0.5, linestyle='dashed', color='gray')
plt.yticks([])
plt.xticks([])
sns.despine()
```

Appendix B

Object Contact Shape Classification Codes

B.1 Brian2 Codes for Software Implementation

B.1.1 Dataset Splitting

```
'''
```

```
This function is made to split the dataset  
into training and testing datasets
```

```
Author : Ali Dabbous
```

```
Cosmic LAB / DITEN / UNIGE
```

```
EDPR LAB / IIT
```

```
ali.dabbous@edu.unige.it
```

```
ali.m.dabbous@gmail.com
```

```
'''
```

```
def readcsv(filename):  
    ifile = open(filename, "rU")  
    reader = csv.reader(ifile, delimiter=",")  
  
    rownum = 0  
    a = []  
  
    for row in reader:  
        if rownum == 0:  
            rownum += 1  
            continue  
        # if rownum >15:  
        #     rownum+=1  
        #     continue  
        del row[0]  
        rownum+=1  
        a.append (row)
```

```

    ifile.close()
    return a

def read_dataset(seed):
    all_dataset=[]
    np.random.seed(seed)
    random.seed(seed)

    dataset=[]
    path = '/Users/AliDabbous/Desktop/Dataset_objects/'
    for i in range(1,435):
        # print(i)
        all_dataset = readcsv(path+'dataset ('+str(i)+').csv')
        all_dataset=np.array(all_dataset).T
        dataset.append(all_dataset)

    index_dataset=[random.sample(range(0,20),20),
                    random.sample(range(20,55),20),random.sample(range(55,92),20),
                    random.sample(range(92,133),20),
                    random.sample(range(133,161),20),
                    random.sample(range(161,198),20),
                    random.sample(range(198,245),20),
                    random.sample(range(245,304),20),
                    random.sample(range(304,352),20),
                    random.sample(range(352,378),20),
                    random.sample(range(378,434),20)]
    training_dataset=np.zeros((160,1))
    for i in range(17):
        for j in range(11):
            training_dataset=np.concatenate((training_dataset ,
            np.array(dataset[index_dataset[j]][i]),
            np.zeros((160,20))), axis=1)
    training_dataset1=np.zeros((160,1))
    for i in range(30):
        training_dataset1=np.concatenate((training_dataset1 ,
        training_dataset , np.zeros((160,20))), axis=1)
    testing_dataset=[]
    testing_dataset=np.zeros((160,1))
    for i in range(17,20):
        for j in range(11):
            testing_dataset=np.concatenate((testing_dataset ,
            np.array(dataset[index_dataset[j]][i]),
            np.zeros((160,20))), axis=1)
    return training_dataset1 ,testing_dataset

```

B.1.2 STDP Rule and Training


```

'''
This code demonstrates the STDP learning rule
implementation and the training procedure
Author : Ali Dabbous
Cosmic LAB / DITEN / UNIGE
EDPR LAB / IIT
ali.dabbous@edu.unige.it
ali.m.dabbous@gmail.com
'''

def create_components():
    cell_params = {
        'c_m': 0.025*pF, 'i_offset': 0.0*nA,
        'R_m': 800E6 * ohm,
        'tau_refrac': 10.0*ms,
        'L2tau_syn_e': 10.0*ms,
        'L2tau_syn_i': 10.0 * ms,
        'v_reset': -70.0 * mV,
        'v_rest': -65.0 * mV, 'v_thresh': -50.0 * mV}
    v_rest=-65.0*mV
    L0leaky_iaf = '''
dv/dt= ((-80*mV) - v +
(10*Mohm * input_stimuli(t,i)) )/(20*ms)
: volt (unless refractory)
'''

    layer0 = NeuronGroup(160, model=L0leaky_iaf,
threshold="v>(-50.0 * mV)", reset="v=-80*mV",
refractory=1*ms, method="euler")

    layer0.v = -80*mV
    return layer0

def learning(learning_dataset):
    dataset1=[]
    dataset1=learning_dataset

    defaultclock.dt = 1 * ms
    analog_clock = 10 * ms
    duration = len(dataset1[0])*10*ms
    nNeurons_layer0 = 160

    allinput=[]
    allinput = np.sum(dataset1, axis=0)
    # plt.figure()
    # plt.plot(allinput)
    startpoint_input = []

```

```

endpoint_input = []
print(len(startpoint_input),
len(endpoint_input))
i = 0
avg=0
while i < (len(allinput)):
    if allinput[i] != 0:
        for j in range(i, len(allinput)-18):
            if allinput[j] == 0:
                for n in range(j, j + 18):
                    if allinput[n] == 0:
                        avg += 1
                if avg == 18:
                    endpoint_input.append(j)
                    startpoint_input.append(i)
                    i = j + 1

                avg = 0
                break
        i += 1

print(len(startpoint_input),
len(endpoint_input))

input_stimuli = TimedArray((dataset1.T)* nA,
dt=analog_clock)
layer0= create_components()

spLayer0 = SpikeMonitor(layer0 , record=True)

layer3_current=np.array([np.zeros(len(dataset1[0])),
np.zeros(len(dataset1[0])),
np.zeros(len(dataset1[0])),
np.zeros(len(dataset1[0])),
np.zeros(len(dataset1[0])),
np.zeros(len(dataset1[0])),
np.zeros(len(dataset1[0])),
np.zeros(len(dataset1[0])),
np.zeros(len(dataset1[0])),
np.zeros(len(dataset1[0]))])
cur=0
pressure_value=30
for j in range(len(startpoint_input)):
    # parameter+=layer3_time[j]
    for k in range(startpoint_input[j],
endpoint_input[j]):
        layer3_current[cur][k]=pressure_value

```

```

        cur+=1
        if cur == 11:
            cur=0

    post_current=TimedArray((layer3_current.T)*
nA, dt=10*ms)
    V_rest=-70*mV
    V_reset=-65*mV
    V_threshold=-50*mV
    tau_ref=1*ms
    C_m=1.5*nfarad #1.5
    tau_m=20*ms #10
    tau_current=0*ms
    tau_tr=3*ms
    tgt=2
    eqs_post = '''
dv/dt= ((-80*mV) - v + (10*Mohm * post_current(t,i))
- ((g_sra)*70*Mohm*(v - (-80*mV))))/(20*ms)
: volt (unless refractory)
dg_sra/dt= - g_sra/(1000*ms) : siemens
dtr/dt=(1.9-tr)/tau_tr :1
'''

    post_neuron = NeuronGroup(11, model=eqs_post,
threshold="v>(-50.0 * mV)", reset="v=-80*mV;
g_sra +=0*nS", refractory=10*ms,
method="euler")
    post_neuron.v = -80*mV
    post_neuron.g_sra = 'rand()*0*nS'
    post_neuron.tr=0
    w_max=1*nA
    w_min=-1*nA
#=====STDP synapses=====
    STDP_model = '''
        w: amp
        '''

    STDP_on_pre = '''
        w+=0.003*int(tgt - tr_post <0)*nA +
        (-0.00001)*int(tgt - tr_post >0)*nA*int(w<0.1*nA)
        w=w_max*int(w >= w_max) +
        w*int(w < w_max and w>w_min) +
        w_min*int(w <= w_min)

        '''

    STDP_on_post = '''
        tr_post= tr_post + 0.1
        '''

    pre_post_synapses=Synapses(layer0 , post_neuron ,
model = STDP_model,

```

```

on_pre = STDP_on_pre ,
on_post= STDP_on_post ,name='raw')
pre_post_synapses.connect()
pre_post_synapses.w=np.zeros(1760)*nA
#=====end=====

spikemon_post = SpikeMonitor(post_neuron ,
variables='v')
stSynp12 = StateMonitor(pre_post_synapses ,
variables = ['w'], record=[i for i
in range(0,160*11)], dt = 100*ms)

#===== end =====
run(duration)
plt.figure()
for j in range(11):
    plt.plot(spikemon_post.t[spikemon_post.i == j]/ms,
            zeros(len(spikemon_post.t[spikemon_post.i == j]))+j ,
            'o', color = 'black', markersize=2)
weight_index=0
weight_index1=1
weight_index2=2
weight_index3=3
weight_index4=4
weight_index5=5
weight_index6=6
weight_index7=7
weight_index8=8
weight_index9=9
weight_index10=10

fig , axes=plt.subplots(nrows=3,ncols=4,
sharex=True ,sharey=True)
while weight_index< 1750:
    axes[0,0].plot(stSynp12.t , stSynp12
    [pre_post_synapses[weight_index]].w)
    weight_index+=11
while weight_index1< 1751:
    axes[0,1].plot(stSynp12.t , stSynp12
    [pre_post_synapses[weight_index1]].w)
    weight_index1+=11
while weight_index2< 1752:
    axes[0,2].plot(stSynp12.t , stSynp12
    [pre_post_synapses[weight_index2]].w)
    weight_index2+=11
while weight_index3< 1753:
    axes[0,3].plot(stSynp12.t , stSynp12

```

```

        [pre_post_synapses[weight_index3]].w)
        weight_index3+=11
    while weight_index4 < 1754:
        axes[1,0].plot(stSynp12.t, stSynp12
            [pre_post_synapses[weight_index4]].w)
        weight_index4+=11
    while weight_index5 < 1755:
        axes[1,1].plot(stSynp12.t, stSynp12
            [pre_post_synapses[weight_index5]].w)
        weight_index5+=11
    while weight_index6 < 1756:
        axes[1,2].plot(stSynp12.t, stSynp12
            [pre_post_synapses[weight_index6]].w)
        weight_index6+=11
    while weight_index7 < 1757:
        axes[1,3].plot(stSynp12.t, stSynp12
            [pre_post_synapses[weight_index7]].w)
        weight_index7+=11
    while weight_index8 < 1758:
        axes[2,0].plot(stSynp12.t, stSynp12
            [pre_post_synapses[weight_index8]].w)
        weight_index8+=11
    while weight_index9 < 1759:
        axes[2,1].plot(stSynp12.t, stSynp12
            [pre_post_synapses[weight_index9]].w)
        weight_index9+=11
    while weight_index10 < 1760:
        axes[2,2].plot(stSynp12.t, stSynp12
            [pre_post_synapses[weight_index10]].w)
        weight_index10+=11

    extracted_weights=list(pre_post_synapses.w)
    return extracted_weights

```

```
seeds = [42356,3356,23,120,1140,160,1154,3,897,567]
```

```

for i in range(10):
    training_dataset=[]
    testing_dataset=[]

    training_dataset, testing_dataset=read_dataset(seeds[i])
    training_dataset=training_dataset.astype(float)
    extracted_weights=learning(training_dataset)

```

B.1.3 Testing

```
'''
```

This code demonstrates the STDP learning rule implementation and the training procedure

Author : Ali Dabbous

Cosmic LAB / DITEN / UNIGE

EDPR LAB / IIT

ali.dabbous@edu.unige.it

ali.m.dabbous@gmail.com

```
'''
```

```
def create_components():
```

```
    cell_params = {
```

```
        'c_m': 0.025*pF, 'i_offset': 0.0*nA,
```

```
        'R_m': 800E6 * ohm,
```

```
        'tau_refrac': 10.0*ms,
```

```
        'L2tau_syn_e': 10.0*ms,
```

```
        'L2tau_syn_i': 10.0 * ms,
```

```
        'v_reset': -70.0 * mV,
```

```
        'v_rest': -65.0 * mV,
```

```
        'v_thresh': -50.0 * mV}
```

```
    v_rest=-65.0*mV
```

```
    L0leaky_iaf = '''
```

```
        dv/dt= ((-80*mV) - v +
```

```
        (10*Mohm * input_stimuli(t,i)) )/(20*ms)
```

```
        : volt (unless refractory)
```

```
'''
```

```
    layer0 = NeuronGroup(160,
```

```
        model=L0leaky_iaf,
```

```
        threshold="v>(-50.0 * mV)",
```

```
        reset="v=-80*mV", refractory=1*ms,
```

```
        method="euler")
```

```
    layer0.v = -80*mV
```

```
    return layer0
```

```
def testing(supervised_weights, testing_dataset):
```

```
    dataset1=[]
```

```
    dataset1=testing_dataset
```

```
    supervised_weights11=supervised_weights
```

```
    defaultclock.dt = 0.1 * ms
```

```
    analog_clock = 10 * ms
```

```
    duration = len(dataset1[0])*10*ms
```

```
    input_stimuli = TimedArray((dataset1.T)*
```

```
    nA, dt=analog_clock)
```

```
    layer0 = create_components()
```

```
    spLayer0 = SpikeMonitor(layer0,
```

```
        record=True)
```

```

V_rest=-70*mV
V_reset=-65*mV
V_threshold=-50*mV
tau_ref=1*ms
C_m=1.5*nfarad #1.5
tau_m=20*ms #10
tau_current=0*ms
tau_tr=3*ms
tgt=2
eqs_post = '''
dv/dt= ((-80*mV)-v)/(15*ms) +
((ie)+ii)/(5*nF) :
volt (unless refractory)
die/dt=-ie/(20*ms): amp
dii/dt=-ii/(2*ms) :amp
'''

post_neuron = NeuronGroup(11,
model=eqs_post,
threshold="v>(-50.0 * mV)",
reset="v=-80*mV", refractory=10*ms,
method="euler")
layer_inh=NeuronGroup(1, model='v : volt',
threshold='v > (-50*mV)',
reset='v = -70*mV',
name='Layer_Inh', method = 'euler')
layer_inh.v = -65*mV
post_neuron.v = -80*mV

#-----STDP synapses==
STDP_model = '''
w: amp
'''

STDP_on_pre = '''
ie_post+=w
'''

pre_post_synapses=Synapses(layer0 ,post_neuron ,
model = STDP_model,on_pre = STDP_on_pre,name='raw')
pre_post_synapses.connect()
pre_post_synapses.w=np.array(supervised_weights11)*amp
Synp_to_inh = Synapses(post_neuron ,
layer_inh ,on_pre = 'v_post+=22*mV',name='Synp_to_inh')
Synp_to_inh.connect(i=[0,1,2,3,4,5,6,7,8,9,10],j=0)
Synp_from_inh = Synapses(layer_inh ,
post_neuron , on_pre='ii_post+=-800*nA',
name='Synp_from_inh')
Synp_from_inh.connect(i=0,j=[0,1,2,3,4,5,6,7,8,9,10])

```

```

=====end=====
    spikemon_post = SpikeMonitor(post_neuron ,
    variables='v')
===== end =====
    run(duration)
    plt.figure()
    for j in range(11):
        plt.plot(spikemon_post.t[spikemon_post.i == j]/ms,
        zeros(len(spikemon_post.t[spikemon_post.i == j]))+j ,
        'o', color = 'black', markersize=2)

seeds = [42356,3356,23,120,1140,160,1154,3,897,567]

for i in range(1):
    training_dataset=[]
    testing_dataset=[]

    training_dataset , testing_dataset=read_dataset(160)
    testing_dataset=testing_dataset.astype(float)
    testing(supervised_weights , testing_dataset)

```

B.2 Hardware Implementation Codes

B.2.1 Raspberry Pi Code

```
'''
```

```
This code implemented on RP for
real time classification
```

```
Author : Ali Dabbous
```

```
Cosmic LAB / DITEN / UNIGE
```

```
EDPR LAB / IIT
```

```
ali.dabbous@edu.unige.it
```

```
ali.m.dabbous@gmail.com
```

```
'''
```

```
import matplotlib.pyplot as plt
```

```
import serial
```

```
import time
```

```
import numpy as np
```

```
from rpi_lcd import LCD
```

```
lcd=LCD()
```

```
lcd.clear()
```

```
ser= serial.Serial(port='/dev/ttyACM0', baudrate=115200)
```

```
list=[]
```



```

timeout=time.time() + 5
list_array=[]
for i in range(160):
    list_array.append([])
print ("press an object")
while True:
    if time.time() > timeout:
        break
    x=ser.readline()
    xy=x.split(";")
    print (xy[1])
    list_array[int(xy[0])].append(int(xy[1]))

print (" take it off")
length_array=[]
for i in range(160):
    length_array.append(len(list_array[i]))
maximum = np.max(length_array)
print (maximum)
for i in range(160):
    if len(list_array[i])==0:
        list_array[i]=np.zeros(maximum)
    else:
        list_array[i]=np.array(list_array[i])
        if len(list_array[i])< maximum:
            list_array[i]=np.concatenate
            ((np.array(list_array[i]),
              np.zeros(maximum - len(list_array[i])) ))

lcd.text(" CLASSIFYING",1)
I=list_array

#t1=process_time()
#print("a")
#t2=process_time()
# conerting analog signal into neuromorphic spikes
#LIF neuron model
# 160 neurons equal to the number of sensors
duration=len(I[0])
V=np.zeros((160,len(I[0])))
v_reset=-80e-3
v_rest=-80e-3
v_th=-50e-3
v_spike=-20e-3
Rm=10e-3
Rm1=0.1e-3
dt=1

```

```

tau_m=20e-3
for i in range(160):
    V[i][0]=-80e-3

spikes1=np.zeros((160,len(I[0])))
spikes=[]
for i in range(160):
    spikes.append([])
t_start=time.time()
for ind in range(160):
    for i in range(1,duration):
        if (V[ind][i-1]== v_spike):
            V[ind][i]=v_reset
        else:
            V[ind][i]=V[ind][i-1] + (dt*(v_rest -
            V[ind][i-1] + (Rm*I[ind][i-1])))/tau_m

        if (V[ind][i]>=v_th):
            V[ind][i]=v_spike
            spikes[ind].append(i)
            spikes1[ind][i]=1

# end layer one neurons model and converting
#analog signals into neuromorphic spikes
#=====

# starting layer two output layer simulation
# first multiplying the spikes generated by layer
# one with the weights between the two layers
# summation of current coming from all
#neurons in layer one to each of neurons in layer two
# a list of current of 11 rows each
#row represents the current arriving from layer one
#neurons to layer two neurons for 11 object
# normalization of current for the
#wta implementation
# winner take all implementation done
#by calculating the maximum current for
#each object and inhibits the remaining
#neurons which the summation is less than
#the maximum current

I_output=[]
count=0
for i in range(160):
    for j in range(11):
        I_output.append(np.array(spikes1[i])*

```

```

        supervised_weights[count])
        count+=1

sum_array=np.zeros(len(I_output[0]))
counterr=0
I_output_final=[]
for i in range(11):
    counterr=i
    for ii in range(160):
        sum_array=np.sum((np.array(sum_array),
        np.array(I_output[counterr])),axis=0)
        counterr+=11
    I_output_final.append(sum_array)
    sum_array=np.zeros(len(I_output[0]))

max_number=0
max_array=[]
for i in range(len(I_output_final[0])):
    for j in range(11):
        if I_output_final[j][i]>max_number:
            max_number=I_output_final[j][i]
    max_array.append(max_number)
    max_number=0
for i in range(len(I_output_final[0])):
    for j in range(11):
        if I_output_final[j][i]< max_array[i]:
            I_output_final[j][i]=0
spikes_output=[]
for i in range(11):
    spikes_output.append([])

V_output=np.zeros((11,len(I[0])))
for i in range(11):
    V_output[i][0]=-80e-3

for ind in range(11):
    for i in range(1,duration):
        if (V_output[ind][i-1]== v_spike):
            V_output[ind][i]=v_reset
        else:
            V_output[ind][i]=V_output[ind][i-1] +
            (dt*(v_rest - V_output[ind][i-1] +
            (Rm1*I_output_final[ind][i-1])))/tau_m

        if (V_output[ind][i]>=v_th):
            V_output[ind][i]=v_spike
            spikes_output[ind].append(i)
t_stop=time.time()

```

```
print("time is:",t_stop-t_start)
# end of simulation
#=====

output=[' BOTTLE CAP', ' ERASER', 'GAS LASHES',
' HIGHLIGHTER CAP', ' KEY', 'marbel',
'rock', 'shaped screw driver',
'spray cover', 'tape', 'wood']
for i in range(11):
    if len(spikes_output[i])>=1:
        lcd.text(output[i],2)
```

Appendix C

Touch Modality Classification Codes

C.1 Brian2 Codes for Software Implementation

C.1.1 Training

```
"""
This code demonstrates the STDP learning rule
implementation and the training procedure
for touch modality classifications
Author : Ali Dabbous
Cosmic LAB / DITEN / UNIGE
EDPR LAB / IIT
ali.dabbous@edu.unige.it
ali.m.dabbous@gmail.com
"""

import numpy as np

import pickle
import csv

# seed(11922)
import numpy as np
import csv
import matplotlib.pyplot as plt
import pandas as pd

def create_components():
    cell_params = {
        'c_m': 0.025*pF, 'i_offset': 0.0*nA,
        'R_m': 800E6 * ohm,
        'tau_refrac': 10.0*ms,
        'L2tau_syn_e': 10.0*ms,
        'L2tau_syn_i': 10.0 * ms,
```

```

        'v_reset': -70.0 * mV,
        'v_rest': -65.0 * mV, 'v_thresh': -50.0 * mV}
v_rest=-65.0*mV
L0leaky_iaf = '''
        dv/dt= ((-80*mV) - v +
                (10*Mohm * input_stimuli(t,i)) )/(20*ms)
        : volt (unless refractory)
        '''
L1leaky_iaf ='''
        dv/dt = ((-65*mV)-v)/(22*ms) +
        (i_syn + (0.0*nA) + i_inj)/(25*pF) : volt
        i_inj : amp
        die/dt = -ie/(50.0*ms) : amp
        dii/dt = -ii/(50.0*ms) : amp
        i_syn = ie + ii : amp
        tau_syn_e : second
        tau_syn_i : second
        ,,,
layer0 = NeuronGroup(160, model=L0leaky_iaf,
                    threshold="v>(-50.0 * mV)",
                    reset="v=-80*mV", refractory=1*ms,
                    method="euler")
layer1 = NeuronGroup(16, model=L1leaky_iaf,
                    threshold='v > (-50.0 * mV)',
                    refractory=cell_params['tau_refrac'],
                    reset='v = -70.0 * mV', name='Layer1',
                    method='euler')

layer0.v = -80*mV
layer1.v = -65.0*mV
Synp01 = Synapses(layer0, layer1, model='w : 1',
                  on_pre='v+=1.5*w*mV', delay=100 * ms, name='S01')
Synp01.connect()

return layer0, layer1, Synp01

def create_w01(seed):
    np.random.seed(seed)

    data = range(0, 160)
    b = np.ones(160)
    for i in (data):
        b[i] = b[i] * i
    c = np.random.shuffle(b)
    a = np.zeros ([160,16])
    for k in range(0,160):
        rowindex = int(b[k])
        column_index = k//10

```

```

        a[rowindex , column_index] = 1

    return a

seed = 2

np.random.seed(seed)
defaultclock.dt = 1 * ms
analog_clock = 10 * ms
duration = len(dataset1[0])*10*ms #93 #4500 weights two
nNeurons_layer0 = 192 # number of neurons in layer0
nNeurons_layer1 = 16 # number of neurons in layer1
allinput = np.sum(dataset1 , axis=0)
startpoint_input = []
endpoint_input = []
i = 0
avg=0
while i < (len(allinput)):
    if allinput[i] != 0:
        for j in range(i, len(allinput)-18):
            if allinput[j] == 0:
                for n in range(j, j + 18):
                    if allinput[n] == 0:
                        avg += 1
                    if avg == 18:
                        endpoint_input.append(j)
                        startpoint_input.append(i)
                        i = j + 1

                avg = 0
                break
        i += 1

input_stimuli = TimedArray((dataset1.T)* nA,
                           dt=analog_clock)
layer0 , layer1 , Synp01 = create_components()
w01=create_w01(60)
w01=np.concatenate(w01,axis=None)
Synp01.w = w01
#===== current signal for layer three=====

layer3_current=np.array([np.zeros(len(dataset1[0])),
                          np.zeros(len(dataset1[0])),
                          np.zeros(len(dataset1[0])),
                          np.zeros(len(dataset1[0])),
                          np.zeros(len(dataset1[0])),
                          np.zeros(len(dataset1[0]))])

cur=0

```

```

pressure_value=30
for j in range(len(startpoint_input)):
    # parameter+=layer3_time[j]
    for k in range(startpoint_input[j],endpoint_input[j]):
        layer3_current[cur][k]=pressure_value
    cur+=1
    if cur == 6:
        cur=0

post_current=TimedArray((layer3_current.T)* nA, dt=10*ms)
V_rest=-70*mV
V_reset=-65*mV
V_threshold=-50*mV
tau_ref=1*ms
C_m=1.5*nfarad #1.5
tau_m=20*ms #10
tau_current=0*ms
tau_tr=3*ms
tgt=2
eqs_post = '''
dv/dt= ((-80*mV) - v + (10*Mohm * post_current(t,i))
        - ((g_sra)*70*Mohm*
          (v - (-80*mV))))/(20*ms) :
        volt (unless refractory)
dg_sra/dt= - g_sra/(1000*ms) : siemens
dtr/dt=(1.9-tr)/tau_tr :1
'''

post_neuron = NeuronGroup(6, model=eqs_post,
                           threshold="v>(-50.0 * mV)",
                           reset="v=-80*mV; g_sra +=0*nS",
                           refractory=10*ms,
                           method="euler")

post_neuron.v = -80*mV
post_neuron.g_sra = 'rand()*0*nS'
post_neuron.tr=0
w_max=1*nA
w_min=-1*nA
#=====STDP synapses=====
STDP_model = '''
        w: amp
'''
STDP_on_pre = '''
w+=0.02*int(tgt-tr_post<0)*nA +
(-0.0001)*int(tgt-tr_post>0)*nA
w=w_max*int(w >= w_max) +
w*int(w < w_max and w>w_min) +
w_min*int(w <= w_min)
'''

```



```

'''
STDP_on_post = '''
            tr_post= tr_post + 0.1
'''
pre_post_synapses=Synapses(layer1 ,post_neuron ,
                           model = STDP_model,
                           on_pre = STDP_on_pre ,
                           on_post= STDP_on_post ,
                           name='raw')

pre_post_synapses.connect()
pre_post_synapses.w=np.zeros(96)*nA
#=====end=====
stSynp12 = StateMonitor(pre_post_synapses ,
                        variables = ['w'],
                        record=[i for i in range(0,16*6)], dt = 10*ms)
#=====end=====
run(duration)

```

C.1.2 Testing

```

"""
This code for validating the network
after learning with 6 different
touch modalities
Author : Ali Dabbous
Cosmic LAB / DITEN / UNIGE
EDPR LAB / IIT
ali.dabbous@edu.unige.it
ali.m.dabbous@gmail.com
"""

import numpy as np
from brian2 import *
import pickle
import csv
from brian2 import*
# seed(11922)
import numpy as np
import csv
import matplotlib.pyplot as plt
import pandas as pd
dataset=[]
path = '/Users/adabbous/Desktop/dynamic_static_testing/wheel/'
def readcsv(filename):
    ifile = open(filename, "rU")
    reader = csv.reader(ifile, delimiter=",")

```

```

rownum = 0
a = []

for row in reader:
    a.append (row)
    rownum += 1

ifile.close()
return a

for i in range(1,11):
    grab_dataset = readcsv(path+'dataset ('+str(i)+').csv ')
    grab_dataset1=np.zeros((len(grab_dataset)-1,
                             len(grab_dataset[1])-1))

    for ii in range(1,len(grab_dataset)):
        for jj in range(1,len(grab_dataset[1])):
            grab_dataset1[ii-1][jj-1]=grab_dataset[ii][jj]
    dataset.append(grab_dataset1)

for i in range(len(dataset)):
    dataset[i]=np.array(dataset[i]).T

dataset2=np.zeros((160,1))
for i in range(10):
    dataset2=np.concatenate((dataset2,np.array(dataset[i]),
                             np.zeros((160,20))),axis=1)

dataset1=np.zeros((160,1))
for i in range(1):
    dataset1=np.concatenate((dataset1,dataset2,
                             np.zeros((160,20))),axis=1)

def create_components():
    cell_params = {
        'c_m': 0.025*pF, 'i_offset': 0.0*nA,
        'R_m': 800E6 * ohm,
        'tau_refrac': 10.0*ms, 'L2tau_syn_e': 10.0*ms
        , 'L2tau_syn_i': 10.0 * ms,
        'v_reset': -70.0 * mV, 'v_rest': -65.0 * mV,
        'v_thresh': -50.0 * mV}
    v_rest=-65.0*mV
    L0leaky_iaf = '''

```

```

        dv/dt= ((-80*mV) - v + (10*Mohm *
            input_stimuli(t,i)))/(20*ms)
        : volt (unless refractory)
    '''
L1leaky_iaf = '''
    dv/dt = ((-65*mV)-v)/(22*ms) +
    (i_syn + (0.0*nA) + i_inj)/(25*pF)
    : volt
    i_inj : amp
    die/dt = -ie/(50.0*ms) : amp
    dii/dt = -ii/(50.0*ms) : amp
    i_syn = ie + ii : amp
    tau_syn_e : second
    tau_syn_i : second
    ,,,
layer0 = NeuronGroup(160, model=L0leaky_iaf,
                    threshold="v>(-50.0 * mV)",
                    reset="v=-80*mV", refractory=1*ms,
                    method="euler")
# training = SpikeGeneratorGroup
(N_input, stimuli[0], stimuli[1] * second, name='training')
layer1 = NeuronGroup(16, model=L1leaky_iaf,
                    threshold='v > (-50.0 * mV)',
                    refractory=cell_params['tau_refrac'],
                    reset='v = -70.0 * mV', name='Layer1',
                    method='euler')
layer0.v = -80*mV

layer1.v = -65.0*mV
Synp01 = Synapses(layer0, layer1, model='w : 1',
                 on_pre='v+=2*w*mV',
                 delay=100 * ms, name='S01')
Synp01.connect()

return layer0, layer1, Synp01

def create_w01(seed):
    np.random.seed(seed)

    data = range(0, 160)
    b = np.ones(160)
    for i in (data):
        b[i] = b[i] * i
#print (b)
    c = np.random.shuffle(b)
#print (b)
    a = np.zeros ([160,16])

```

```

    #print (a.shape)
    for k in range(0,160):
        rowindex = int(b[k])
        column_index = k//10
    # print (rowindex, column_index)
        a[rowindex, column_index] = 1

    return a

seed = 2

np.random.seed(seed)
defaultclock.dt = 0.1 * ms
analog_clock = 10 * ms
duration = len(dataset1[0])*10*ms
nNeurons_layer0 = 192 # number of neurons in layer0
nNeurons_layer1 = 16 # number of neurons in layer1
FILE_PATH = '/Users/adabbous/Desktop/'
file_ali=FILE_PATH+'supervised_weights_static_dynamic_1'
supervised_weights=pickle.load(open(file_ali, 'rb'))

allinput = np.sum(dataset1, axis=0)
# plt.figure()
# plt.plot(allinput)
startpoint_input = []
endpoint_input = []
i = 0
avg=0
while i < (len(allinput)):
    if allinput[i] != 0:
        for j in range(i, len(allinput)-8):
            if allinput[j] == 0:
                for n in range(j, j + 8):
                    if allinput[n] == 0:
                        avg += 1
                    if avg == 8:
                        endpoint_input.append(j)
                        startpoint_input.append(i)
                        i = j + 1

                avg = 0
                break
        i += 1

input_stimuli = TimedArray((dataset1.T)*
                            nA, dt=analog_clock)

```

```

layer0 , layer1 , Synp01 = create_components ()
w01=create_w01(60)
w01=np.concatenate(w01,axis=None)
Synp01.w = w01
spLayer1 = SpikeMonitor(layer1 , record=True)

V_rest=-70*mV
V_reset=-65*mV
V_threshold=-50*mV
tau_ref=1*ms
C_m=1.5*nfarad #1.5
tau_m=20*ms #10
tau_current=0*ms
tau_tr=3*ms
tgt=2
eqs_post = '''
dv/dt= ((-80*mV)-v)/(15*ms) +((ie*60)+ ii)/(5*nF)
: volt (unless refractory)

dtr/dt=(1.9-tr)/tau_tr :1
die/dt=-ie/(20*ms): amp
dii/dt=-ii/(2*ms) :amp
'''
post_neuron = NeuronGroup(6, model=eqs_post ,
                           threshold="v>(-50.0 * mV)",
                           reset="v=-80*mV",
                           refractory=10*ms, method="euler")
layer_inh=NeuronGroup(1, model='v : volt ',
                      threshold='v > (-50*mV)',
                      reset='v = -70*mV',

                      name='Layer_Inh ',
                      method = 'euler')

layer_inh.v = -65*mV
post_neuron.v = -80*mV

post_neuron.tr=0
w_max=1*nA
w_min=-1*nA
#=====STDP synapses=====
STDP_model = '''
        w: amp
'''
STDP_on_pre = '''
        w+=0*nA
        ie_post+=w
'''

```

```

'''
STDP_on_post = '''
            tr_post= tr_post + 0.1
'''
pre_post_synapses=Synapses(layer1 ,
                           post_neuron , model = STDP_model ,
                           on_pre = STDP_on_pre ,
                           on_post= STDP_on_post
                           ,name='raw')

pre_post_synapses.connect()
pre_post_synapses.w=np.array(supervised_weights)*amp
Synp_to_inh = Synapses(post_neuron , layer_inh ,
                       on_pre = 'v_post+=22*mV' ,
                       name='Synp_to_inh')
Synp_to_inh.connect(i=[0,1,2,3,4,5],j=0)
Synp_from_inh = Synapses(layer_inh , post_neuron ,
                         on_pre='ii_post+=-800*nA' ,
                         name='Synp_from_inh')
Synp_from_inh.connect(i=0,j=[0,1,2,3,4,5])
#=====end=====
statemon_post = StateMonitor(post_neuron , ['v' , 'tr'] ,
                             record=True)
spikemon_post = SpikeMonitor(post_neuron , variables='v')
stSynp12 = StateMonitor(pre_post_synapses , variables = ['w'] ,
                       record=[i for i in range(0,16*6)] , dt = 1*ms)
#===== end =====
run(duration)

plt.figure()
for j in range(16):
    plt.plot(spLayer1.t[spLayer1.i == j]/ms,
             zeros(len(spLayer1.t[spLayer1.i == j]))+j ,
             'o' , color = 'black' , markersize=0.6)

plt.figure()
for j in range(6):
    plt.plot(spikemon_post.t[spikemon_post.i == j]/ms,
             zeros(len(spikemon_post.t[spikemon_post.i == j]))+j ,
             'o' , color = 'black' , markersize=2)

```