

COMENIUS UNIVERSITY IN BRATISLAVA
FACULTY OF MATHEMATICS, PHYSICS AND INFORMATICS

BIOLOGICALLY INSPIRED PREDICTIVE MODEL
OF PROPRIOCEPTIVE BODY REPRESENTATIONS
BACHELOR THESIS

2016
LEJLA METOHAJROVÁ

COMENIUS UNIVERSITY IN BRATISLAVA
FACULTY OF MATHEMATICS, PHYSICS AND INFORMATICS

BIOLOGICALLY INSPIRED PREDICTIVE MODEL
OF PROPRIOCEPTIVE BODY REPRESENTATIONS
BACHELOR THESIS

Study programme: Informatics
Field of study: 2508 Informatics
Study department: Department of Informatics
Advisor: prof. Ing. Igor Farkaš, Dr.
Consultant: Mgr. Matěj Hoffmann, PhD.

Bratislava, 2016
Lejla Metohajrová



Comenius University in Bratislava
Faculty of Mathematics, Physics and Informatics

THESIS ASSIGNMENT

Name and Surname: Lejla Metohajrová
Study programme: Computer Science (Single degree study, bachelor I. deg., full time form)
Field of Study: Computer Science, Informatics
Type of Thesis: Bachelor's thesis
Language of Thesis: English
Secondary language: Slovak

Title: Biologically inspired predictive model of proprioceptive body representations

Aim:

1. Study the relevant literature from the area of neuroscience related to body representations, as well as from area of machine learning (neural networks).
2. Implement and evaluate the predictive model of an artificial neural network, utilizing proprioceptive and haptic information. For training the model use the data acquired from the simulator of the humanoid robot iCub.

Annotation: The project topic belongs to cognitive robotics, where one of the complex goals of developing the (robotic) agent is to create its so-called body schema, a multimodal representation of robot's body that is used in various tasks. Formation of such a representation is a part of ontogenesis.

Supervisor: prof. Ing. Igor Farkaš, Dr.
Consultant: Mgr. Matěj Hoffmann, PhD.
Department: FMFI.KAI - Department of Applied Informatics
Head of department: prof. Ing. Igor Farkaš, Dr.

Assigned: 27.10.2015

Approved: 30.10.2015

doc. RNDr. Daniel Olejár, PhD.
Guarantor of Study Programme

.....
Student

.....
Supervisor



Univerzita Komenského v Bratislave
Fakulta matematiky, fyziky a informatiky

ZADANIE ZÁVEREČNEJ PRÁCE

Meno a priezvisko študenta: Lejla Metohajrová
Študijný program: informatika (Jednoodborové štúdium, bakalársky I. st., denná forma)
Študijný odbor: informatika
Typ záverečnej práce: bakalárska
Jazyk záverečnej práce: anglický
Sekundárny jazyk: slovenský

Názov: Biologically inspired predictive model of proprioceptive body representations
Biologicky inšpirovaný predikčný model proprioceptívnych reprezentácií tela

Cieľ: 1. Naštudujte relevantnú literatúru z oblasti neurovedy týkajúcu sa reprezentácií tela ako aj z oblasti strojového učenia (neurónové siete).
2. Implementujte a vyhodnoťte predikčný model umelej neurónovej siete, ktorý využíva proprioceptívne a dotykové informácie. Na tréningovanie použite dáta zo simulátora humanoidného robota iCub.

Anotácia: Náplň projektu spadá do oblasti kognitívnej robotiky, kde jedným z komplexných cieľov vývinu (robotického) agenta je vytvorenie tzv. schémy tela, čo je multimodálna reprezentácia tela robota, ktoré potom využíva v rôznych úlohách. Vytvorenie takejto reprezentácie je súčasťou procesu ontogenézy.

Vedúci: prof. Ing. Igor Farkaš, Dr.
Konzultant: Mgr. Matěj Hoffmann, PhD.
Katedra: FMFI.KAI - Katedra aplikovanej informatiky
Vedúci katedry: prof. Ing. Igor Farkaš, Dr.
Dátum zadania: 27.10.2015

Dátum schválenia: 30.10.2015

doc. RNDr. Daniel Olejár, PhD.
garant študijného programu

.....
študent

.....
vedúci práce

Acknowledgements:

I would like to express my sincere gratitude to my advisor prof. Ing. Igor Farkaš, Dr. for his guidance and innumerable consultations that helped me to get through various topics of this thesis.

Besides my advisor, I would like to thank Mgr. Matěj Hoffmann, PhD., external consultant from Italian Institute of Technology in Genova, for the provided materials, professional advice and early responses to my e-mails.

Last but not the least, I would like to thank my family for their patience and for supporting me throughout writing this thesis.

Abstract

The goal of this thesis is to create a simplified, biologically inspired multimodal representation of the body (body schema). For this we use the open simulator of the humanoid robot iCub that learns to associate two modalities of somatosensation - proprioception and tactile sensation. The proposed solution employs a predictive model based on a multilayer perceptron that is trained off-line in a supervised way, given the current proprioceptive state and an action as inputs, to predict the future body state and a potential tactile sensation. Prior to training the model, the training data was collected from the simulated iCub that was made to perform various types of the self-touch using its both arms. The neural network model was successfully tested for its ability to predict touch in the corresponding proprioceptive states, when no action was implied as input, but also to anticipate the touch during action, shortly before it occurred.

Keywords: body schema, somatosensory, iCub, predictive model, population coding

Abstrakt

Cieľom tejto práce je vytvoriť zjednodušenú, biologicky inšpirovanú multimodálnu reprezentáciu tela (schému tela). V tejto práci používame open-source simulátor humanoidného robota iCub, ktorý sa učí asociovať medzi dvoma modalitami somatosenzorického systému, propiocepciou a hmatovým vnemom. Navrhnuté riešenie využíva predikčný model založený na viacvrstvovom perceptróne, ktorý je učený off-line, metódou učenia s učiteľom. Dokáže predvídať nasledujúci stav tela a potenciálny hmatový vnem na základe aktuálneho propioceptívneho stavu a vstupnej akcie. Pred samotným učením boli pomocou simulátora iCub-a, ktorý vykonával rôzne vzájomné dotyky oboch jeho rúk, zozbierané trénovacie dáta. Neurónová sieť bola úspešne otestovaná a preukázala vlastnosť predvídať dotyk v príslušných propioceptívnych stavoch, v prípade, že nebola na vstupe daná žiadna akcia, ale aj anticipovať dotyk pri pohybe, krátko pred tým, ako nastal.

Kľúčové slová: schéma tela, somatosenzorický, iCub, predikčný model, populačné kódovanie

Contents

Introduction	1
1 Biological Motivation	2
1.1 Body representation	2
1.2 Proprioception	2
1.3 Tactile sensation	3
1.4 Somatoperception and body schema	3
2 Humanoid Robot iCub	4
2.1 The body representation	4
2.1.1 Proprioceptive representation	4
2.1.2 Tactile representation	5
2.1.3 iCub simulator	6
3 Modeling Approach	7
3.1 Forward and inverse models	7
3.1.1 Forward models	7
3.1.2 Inverse models	8
3.2 Multi-layer perceptron in a nutshell	8
3.2.1 Supervised learning	8
3.2.2 A feedforward neural network	8
3.3 Implementation design	11
3.3.1 Population coding	11
3.3.2 Problem simplification	12
4 Implementation	13
4.1 The data	13
4.1.1 Tactile data	14
4.1.2 Proprioceptive data	15
4.2 Training	16
4.2.1 Encoding problem	16

<i>CONTENTS</i>	vii
4.2.2 Setting the parameters	17
4.2.3 Backward reconstruction of sigmoidal population coding	17
5 Results	20
5.1 The general ability of learning	20
5.2 Prediction of touch	22
Conclusion	24
Appendix A Ranges of DoF of iCub's arm	26
Appendix B Tactile regions	27

List of Figures

2.1	iCub kinematic structure	5
2.2	Triangular modules of artificial skin	5
2.3	Self touch	6
3.1	Forward and inverse models	7
3.2	Multilayer perceptron	9
3.3	Network architecture	11
3.4	Gaussian population coding	12
4.1	iCub simulator: double touch	14
4.2	Output with Gaussian encoding	16
4.3	MSE using Gaussian and sigmoidal encoding	17
4.4	Model of MLP	18
4.5	MSE with and without bias units	19
4.6	Backward reconstruction	19
5.1	Proprioceptive output before and after training	20
5.2	Tactile output before and after training	21
5.3	Average tactile activation on a neuron	22
5.4	Average tactile activation in time	23
5.5	Tactile output modality test	23

List of Tables

A.1	Ranges of DoF of iCub's arm	26
B.1	Tactile regions	30

Introduction

In recent years, the scientists in the field of cognitive robotics have been trying to simulate processes, inspired by those of human mind, to create autonomous robots that could perform various tasks. The iCub is an open-source humanoid robot, developed at Italian Institute of Technology, which by its size, body morphology and sensorimotor capacities reminds of a 4-year old child. For research purposes, there is the iCub simulator, a software that mimics the functionality of the physical robot.

Our goal is to create a body representation of the iCub, which maps spatial properties of its body. The representation, referred to as the body schema, is inspired by the behavior of an infant who is believed to create such representation by touching itself. Based on proprioceptive senses that contain information of the position and the movement of the body, we will try to predict tactile stimuli and its localization on the artificial skin of the robot in case of self-touch configuration.

In order to create the body schema, we will employ a predictive model of a feed-forward neural network (multi-layer perceptron) that is given the current state of the body and an action (a movement) and predicts, together with a potential touch, the next state of the body. The data used for training and testing of our model were generated by the iCub simulator.

In Chapter 1, we describe the biological motivation behind the body schema and define the important biological terms. Chapter 2 introduces the iCub and its essential properties. The technical concepts, as the predictive model, the neural networks and also the design of our model, are depicted in Chapter 3. The details of the implementation, the data and the chosen parameters are specified in Chapter 4. Finally, the results of the implemented predictive model are analyzed in Chapter 5.

Chapter 1

Biological Motivation

In this chapter, we will discuss representations of a human body constructed by a human mind. We will also introduce some biological terms important for understanding the matter of this thesis.

1.1 Body representation

We perceive our body through senses, which are subsequently being processed, the central nervous system decides on the response and we react accordingly. Those senses such as touch, pressure, pain and temperature perception are referred to as somatosensation. Processing of somatosensation can be divided into two levels: physiological somatoperception and cognitive somatorepresentation (Longo et al., 2010).

Somatoperception represents the processes of our nervous system on the physical level. Physiological processes that are parts of somatoperception, are, for instance, localisation of somatic stimuli on the body surface, perceiving the current posture, metric properties of our body, as well as processing of responses to those stimuli.

Somatorepresentation, on the other hand, involves cognitive processes of the human mind. Encyclopaedic, lexical, and topological knowledge about bodies in general and attitudes towards one's body are parts of somatorepresentation. It represents the link between the physical and psychological self.

1.2 Proprioception

Proprioception is the sense that allows us to perceive the position and movement of our body in space. The sensory receptors of proprioception, i.e. proprioceptors, are mechano-receptors located in joints and tendons and stretch-sensitive receptors in muscles and skin (Proske and Gandevia, 2012).

The neurons responsible for these sensations are called afferent neurons. They carry

afferent information all the way to the somatosensory cortex, where those signals are being processed by the brain. The impulse released from the central nervous system as a response is carried by efferent neurons to the effectors.

The configuration of our body can be described using the afferent signals from proprioceptors and the efferent signals specifying our movements. It is important to realize that this concept does not employ sight and therefore is a powerful tool. It gives us information about the relative position of our body parts, even if we do not see them (e.g. we can touch our hands behind our back without seeing where they lead).

1.3 Tactile sensation

On the basis of tactile perception, we can distinguish between detection and localization of touch on the skin. According to neurological studies, the ability to localize tactile stimuli depends on an inward representation of the body surface, the superficial schema (Longo et al., 2010).

1.4 Somatoperception and body schema

As a part of this thesis, we will create an artificial body schema, in neuroscience often called the postural schema, which maps spatial properties of the body. In order to do this, we will combine two aspects of somatoperception, proprioception and tactile sensation. In spite of the fact, a body schema involves other modalities like vision, we will leave that behind and focus on somatosensory input.

The schema uses touch localization to determine the accuracy of the prediction of a contact point, based on proprioceptive perception of the body parts position and their movements. This approach was inspired by the behavior of infants and certain animals who are believed to calibrate their body representations by self-stimulation (Roncone et al., 2014). Technical view of the model will be described in Chapter 3.

Chapter 2

Humanoid Robot iCub

As technology advances, people are starting to create intelligent robotic systems of all kinds. On the one hand, there are robots meant for labour needs, as household helpers or to be used for what is impossible for a human. For instance the Mars Exploration Rover. On the other hand, scientists search to reveal and simulate cognitive processes beyond human mind. Therefore, they try to create artificial humanoid robots, complex systems that can learn and behave as humans.

The iCub is one of these humanoid robots, used for research in cognitive neuroscience and artificial intelligence. It was developed at Italian Institute of Technology in Genova as an open-source project. The robot, with its height 105 *cm* reminds of a 4-year old child (Metta et al., 2010).

2.1 The body representation

Considering the iCub's limited structure of the body, compared to a human body, we need to use representations that do not completely correspond to the human body. As we mentioned in Chapter 1, our outlined body schema consists of two representations: proprioceptive and tactile. The tactile input allows us to calibrate the kinematic parameters by enclosing the kinematic chain with self-touch. This is referred to as the closed-loop system.

2.1.1 Proprioceptive representation

The motor system of the iCub is formed by 53 motors attached to its joints. Each joint has 1 to 3 axes of rotation that together creates 53 degrees of freedom (DoF). For demonstration, as indicated in Figure 2.1, there are 6 DoF on the head, 3 on the torso, 2×16 on the arms and 2×6 on the legs (Metta et al., 2010).

The configuration of iCub's current posture can be represented using the angles of its joints in all DoF. Each DoF has a different range of angles and a different default

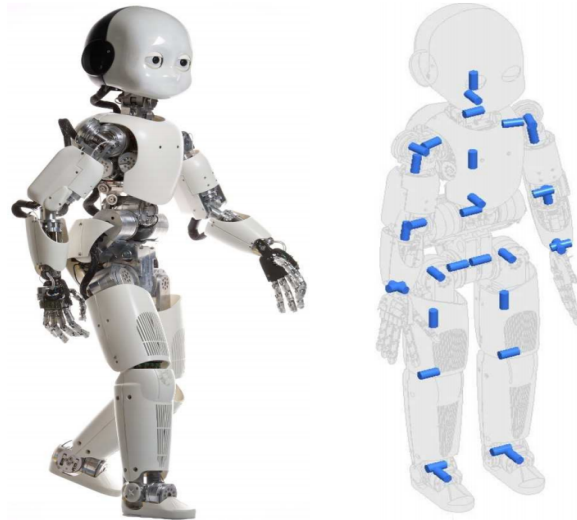


Figure 2.1: *Humanoid robot iCub and its kinematic structure (Parmiggiani et al., 2012). The joints on the fingers are not depicted.*

position (not necessarily 0). The ranges of DoF in robot's arm are stated in Table A.1 (the same for both arms). The information about the body's next movement can be represented as a vector of angular changes, one per each DoF.

2.1.2 Tactile representation

Several parts of iCub's body are covered by artificial skin, organized into 18 patches (1 in each hand, 2 in each forearm, 4 in each upper arm and 4 in the torso). Each patch is composed by triangular modules, each consisting of 10 tactile sensors (a.k.a. taxels) (Bruchi et al., 2016). There are approximately 1750 taxels on iCub's skin altogether. The triangular modules are visible in Figure 2.2.

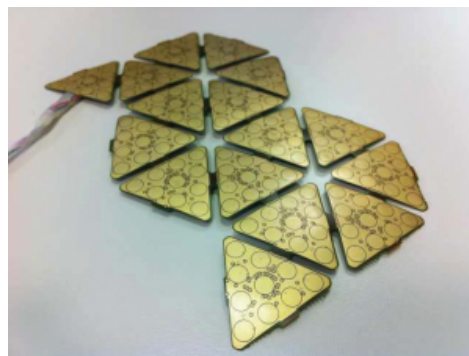


Figure 2.2: *Triangular modules of artificial skin (Bruchi et al., 2016).*

In Figure 2.3, we can see iCub touching its arm and the visualization of the activation of corresponding sensors on the sensory map on the left side in the figure.



Figure 2.3: *iCub touching its arm (Roncone et al., 2014).*

2.1.3 iCub simulator

The data we will be using have been generated using iCub simulator¹, which is an open source software developed for research purposes. It sufficiently mimics the functionality of the iCub, therefore we do not need the access to the physical robot.

¹http://wiki.icub.org/wiki/ICub_Simulator_Installation

Chapter 3

Modeling Approach

The implementation design, along with the explanation of necessary concepts will be outlined in this chapter.

3.1 Forward and inverse models

Autonomous robots should be able to learn to make predictions of their future state based on their observations. There is evidence of predictive models employed by a human brain (Davidson and Wolpert, 2005). The basic ones used in machine learning are forward and inverse models.

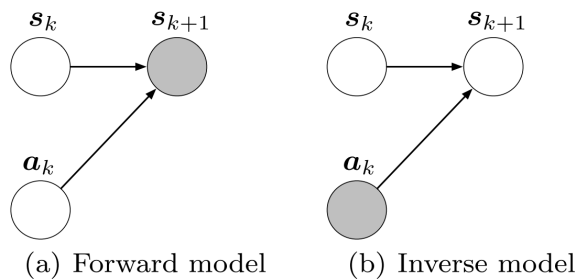


Figure 3.1: (a) The forward model allows inferring the next state s_{k+1} given the current state s_k and the action a_k . (b) The inverse model determines the action a_k required to move the system from the current state s_k to the next state s_{k+1} (Nguyen-Tuong and Peters, 2011).

3.1.1 Forward models

Forward models are functions that imitate physical systems. A forward model represents a causal relationship between the given state of a system and an intervening action (Nguyen-Tuong and Peters, 2011). It simply outputs the conclusive state. On the basis of psychophysical and electrophysiological studies, it has been suggested that

the sensorimotor system employs forward models to predict the consequences of motor movements (Davidson and Wolpert, 2005). Hence, they are otherwise known as predictive models.

3.1.2 Inverse models

On the other hand, inverse models predict the action that needs to be executed to acquire the desired state from the current state. An inverse model represents an anti-causal relationship (Nguyen-Tuong and Peters, 2011), they do not always exist and can be ambiguous (different actions can cause the same consequences). In general, learning an inverse model is an ill-posed problem.

In other words, a forward model predicts the consequences of a movement from the current configuration, while an inverse model predicts the movement, that would cause a desired configuration. These models are described in Figure 3.1.

In this thesis, we will employ a forward model to predict the consecutive configuration and a possible contact of iCub's body from the current configuration, determined by proprioception, and the given motor command.

3.2 Multi-layer perceptron in a nutshell

For the model implementation, we will use concepts of artificial neural networks (ANNs). In this section we will shortly introduce the ANNs and describe the architecture of the ANN we have chosen for the implementation of a predictive (forward) model.

3.2.1 Supervised learning

Supervised learning is a type of machine learning algorithm, which infers the output function based on training examples. Each example is a pair of an input vector and a desired output (target). There are two types of supervised learning problem: classification (in case of a discrete output function) and regression (if the output function is continuous).

3.2.2 A feedforward neural network

Artificial neural networks are computational models of machine learning inspired by biological neural networks that can be trained (using various methods of machine learning) to perform different tasks (Haykin, 2009). There are multiple types of ANNs, however, we will only be using feedforward neural networks.

A perceptron, an artificial neuron, receives one or more weighted inputs and sums them in order to produce output. The perceptron outputs a non-zero value only if the

weighted sum exceeds a certain threshold. For that reason a bias unit (usually a value of 1) is often added to the input. The weighted sum (linear combination) is then passed to an activation function (usually sigmoid function) to produce a non-linear output.

A feedforward neural network is composed of layers of neurons. The input layer is composed of the input sensors, equivalently, the output layer is the output of the network. A network that only consists of input and output layer is termed a single layer perceptron, while model consisting of multiple layers is a multilayer perceptron (MLP). The other layers, between the input and the output layer, are referred to as hidden layers. A typical MLP with one hidden layer is shown in Figure 3.2.

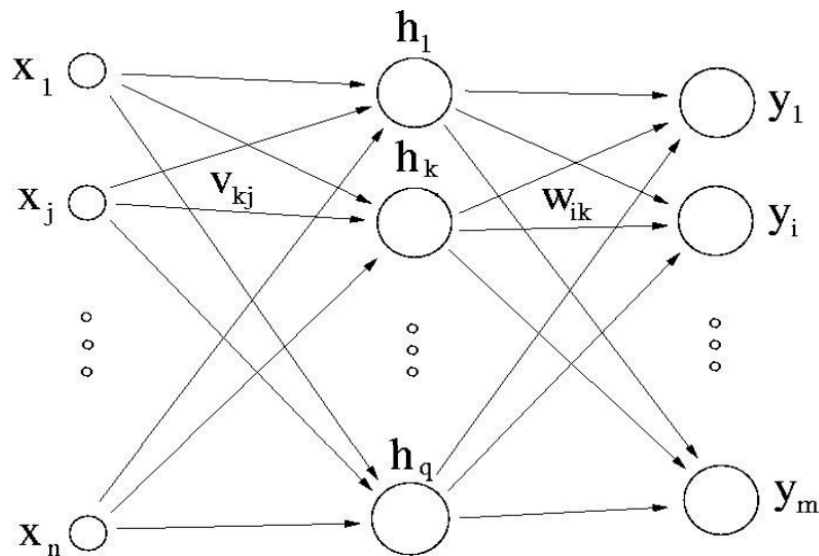


Figure 3.2: *Multilayer perceptron with an input vector \mathbf{x} , one hidden layer \mathbf{h} , an output layer \mathbf{y} and weight matrices \mathbf{V} and \mathbf{W} (Farkaš, 2011).*

To train an ANN, we need to set its weights, so that the output is close to the target value. A properly trained ANN has the ability of generalization, which means it can predict the output of an input vector that was not in the training data. The ANN is trained after multiple cycles of forward propagation and backpropagation of error, called epochs.

3.2.2.1 Forward propagation

To evaluate the output of a neural network, we use the forward propagation. Working with a MLP, we need to compute the activation of the layers in the direction from the input to the output layer. We will demonstrate the forward propagation on the example network in Figure 3.2. The activation of the k -th neuron in the hidden layer is the output of the activation function f that is given the linear combination of the

input vector \mathbf{x} and the k -th vector from the weight matrix \mathbf{V} as the argument.

$$h_k = f\left(\sum_{j=1}^{n+1} V_{kj}x_j\right) \quad (3.1)$$

Similarly, the activation of the i -th neuron in the output layer is calculated by the output of the activation function with the input of the linear combination of the hidden layer and the corresponding vector from the weight matrix \mathbf{W} .

$$y_i = f\left(\sum_{k=1}^{q+1} W_{ik}h_k\right) \quad (3.2)$$

There are various types of activation functions used in neural networks. The typical is the logistic function with the output from the interval $(0, 1)$ that we will be using.

$$f(\text{net}) = \frac{1}{1 + \exp(-\text{net})} \quad (3.3)$$

3.2.2.2 Backpropagation

The backward propagation of errors, i.e. the backpropagation, is a learning algorithm used for training ANNs (Rumelhart et al., 1986). Initially, the weights of a neural network are set randomly. Based on the desired output (target), we compute the error of an ANN by comparing it to the output and then update all weights in the network using backpropagation.

We compute the error of the i -th neuron from the output layer using formula 3.4, where f'_i is the derivative of the activation function with respect to its argument (net).

$$\delta_i = (d_i - y_i)f'_i \quad (3.4)$$

At first, we update the weights closer to the output layer by adding a small value to each of them. The update is computed by the formula

$$W_{ik}(t+1) = W_{ik}(t) + \alpha\delta_i h_k \quad (3.5)$$

where W_{ik} is the weight between the k -th neuron in the hidden layer and the i -th neuron in the output layer, α is the learning rate (usually $0 < \alpha < 1$), δ_i is the error of the i -th neuron, computed by the formula 3.4 and h_k is the k -th neuron in the hidden layer.

Then, we back propagate the error through the network. We compute the error of the hidden layer, as follows

$$\delta_k = \left(\sum_i W_{ik}\delta_i\right)f'_k \quad (3.6)$$

Equivalently, we update the weights between the hidden and the input layer

$$V_{kj}(t+1) = V_{kj}(t) + \alpha\delta_k x_j \quad (3.7)$$

3.3 Implementation design

In this section, we will discuss the architecture of our forward model that can predict the future state of iCub’s body, i.e. limbs configuration and tactile sensation, based on the current posture and the motor command. The diagram of this model is sketched in Figure 3.3.

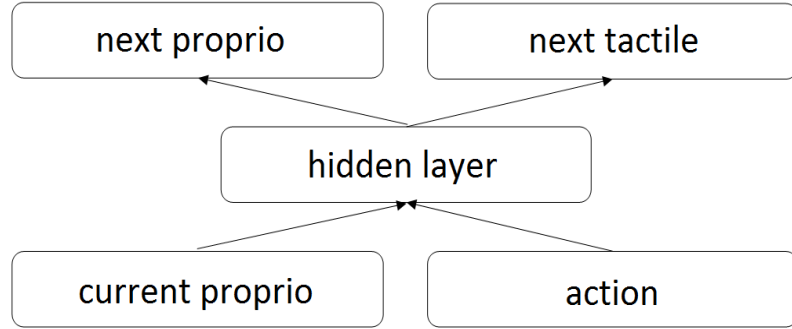


Figure 3.3: *Network architecture design of the predictive model.*

3.3.1 Population coding

In this section, we will describe the representation of the proprioceptive information in our model. A population coding is often used, to encode data by activation of several neurons, termed a population of neurons. Population coding is inspired by the activation of biological neurons, which are believed to encode information in a similar way (Pouget et al., 2000). There are a few different methods of population coding. In this thesis, we will use encoding by Gaussian and by sigmoidal tuning curves.

3.3.1.1 Gaussian tuning curves

Population encoding by Gaussian tuning curves is often referred to as position coding. We can encode a position x by a population of Gaussian tuning curves with a different center μ and the same standard deviation σ (the distance between the center and the inflection point of each curve). The activation of each Gaussian is given by

$$y = \exp\left(-\frac{(x - \mu)^2}{2\sigma^2}\right) \quad (3.8)$$

An example of position coding of a number in range $[-1, 1]$ is shown in Figure 3.4. Since the DoF all have different ranges of possible angles, we need to normalize the encoding, in order to unify the proprioceptive input. For our purpose, we will use the corresponding range of each DoF on the x-axis, the activation of neurons on the y-axis will always be in range $[0, 1]$.

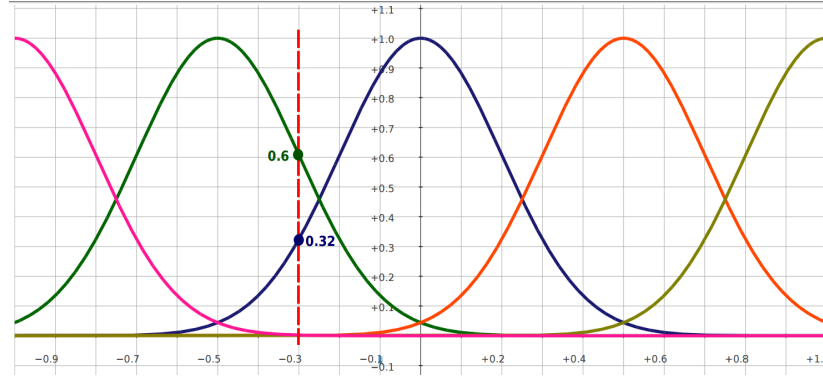


Figure 3.4: *Gaussian encoding of a number -0.3 by a population of 5 neurons with corresponding activations $(0, 0.6, 0.32, 0, 0)$ (Zdechovan, 2012).*

3.3.1.2 Sigmoidal tuning curves

Alternative to the Gaussian encoding is sigmoidal population coding. The only difference is the sigmoidal tuning curve. Activation of neurons, each with specified offset, of a position x is calculated as

$$y = \frac{1}{1 + \exp(-x + \text{offset})} \quad (3.9)$$

3.3.2 Problem simplification

The proprioceptive information tells us only about the configuration of iCub’s body, not about any other external objects. Therefore, we are only concerned about self-touch of the body. That means, there will always be at least two activated tactile regions on iCub’s body. To simplify this a little, we will only take into consideration the robot’s arms touching itself, excluding the proprioceptive information about the fingers, which makes $2 \times 7 = 14$ DoF (see Table A.1). Further on, we will decrease the contact surface to the fingers, palms and forearms.

The tactile sensory input space of iCub simulator is reduced in comparison to the physical iCub. The surface on the artificial skin in the simulator is divided into several regions, each containing a few (2 to 3) triangular modules (Varga, 2016).

Regarding our decision to only operate with the tactile sensors on fingers, palms and forearms, we will describe the tactile regions on these parts. The tactile input space consists of 42 regions. There is one region on each finger, $5 \times 2 = 10$ regions on palms and $11 \times 2 = 22$ regions on forearms. The complete list of corresponding taxels of each region can be found in Table B.1.

In our model of a multilayer perceptron, we will represent the 42-dimensional space of tactile regions by 42 neurons, each with activation in the range $[0, 1]$, where 1 means touch, and 0 means no touch (activated taxel) in that particular region.

Chapter 4

Implementation

The complete implementation of the predictive model of the iCub’s body representation is covered in this chapter. We chose Python for the implementation of the neural network and for the data processing. For operations with matrices, we used Numpy, a package for scientific computing with Python. The source files `formater.py`, `encoder.py` and `mlp.py`, and the data files `leftArm.log`, `rightArm.log` and `skinEvents.log` are located in the attached CD.¹

4.1 The data

All data used for training and testing have been generated using the iCub simulator by Dr. Matěj Hoffmann. The data capture the course of a number of trajectories (more than 200). At the beginning of each trajectory, iCub starts in the default (home) position, then moves both arms, one at a time, and ends up in a double-touch configuration, where the fingers of one arm touch the contralateral palm or forearm. An example of double touch configuration in the iCub simulator is shown in Figure 4.1. The dataset duration is 60 minutes. Each trajectory takes less than 20 seconds, from which the touch lasts roughly 5 seconds.

The sampling of the tactile and the proprioceptive data was different. The sampling of the proprioceptive data varies between 10-30 ms, while the tactile varies between 30-70 ms. However, we needed both information at the same time. Therefore, we merged the data with a lower sampling rate and got approximately 100000 records. This was still too dense, because the change between a position in time t and $t + 1$ (record number t and $t + 1$) was too little, so we took into consideration every 5-th record, ending up with 20000 records of the iCub’s body state. The script that processes the data is saved in the file `formater.py`. Before training, we divided the dataset into the training and the testing data in a ratio of 3 to 1. In order to train and test the

¹also at <https://github.com/LejlaMetohajrova/proprio-haptic-prediction>

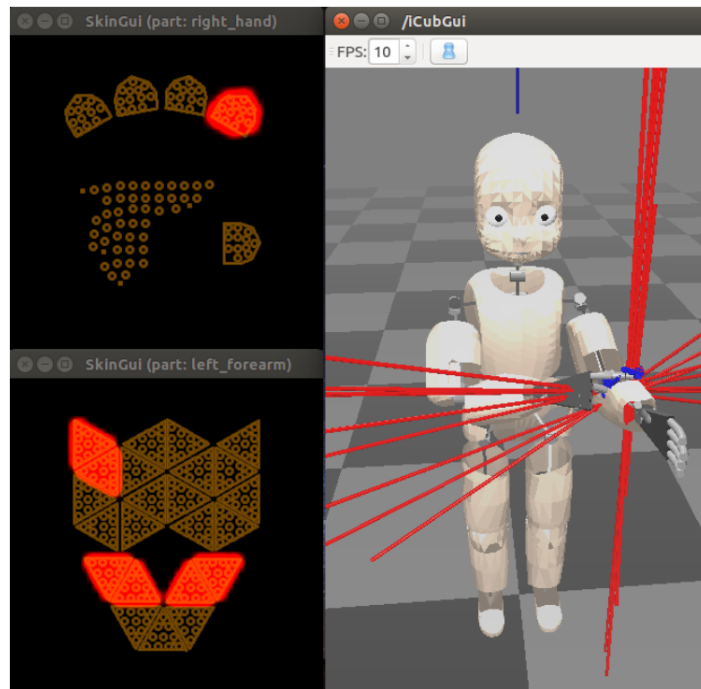


Figure 4.1: *Double touch of the iCub simulator. Skin GUI on the left side shows the activated regions of touch. (Varga, 2016).*

model, we needed to process the data into a form suitable for a neural network.

4.1.1 Tactile data

As mentioned in Section 3.3.2, the tactile input space of our model consists of 42 neurons. Although the data contain much complex information, we had to extract what is important and create a 42-dimensional vector of values in interval $[0, 1]$ representing the tactile input.

The original tactile data consist of multiple skin contact vectors per record. Each skin contact vector has the following format (Bruchi et al., 2016):

```
((contactId bodyPartId linkNumber skinPart)
(centerOfPressure_x cOP_y cOP_z)
(force_x f_y f_z)
(moment_x m_y m_z)
(geometricCenter_x gC_y gC_z)
(surfaceNormalDirection_x sND_y sND_z)
(activatedTaxelId1 aTId2 ... aTIdN) average_pressure)2
```

The interesting values are `skinPart` and the vector of the activated taxel IDs

²For clarity we split the skin contact vector into several lines.

(marked as `activatedTaxelId1 aTid2 ... aTidN`). The values of `skinPart` in our dataset could be integers 1, 2, 4 or 5 with the following meaning: (left hand = 1, left forearm = 2, right hand = 4, right forearm = 5). We ignored other potential values that could represent a different skin part (e.g. the torso). The taxel IDs are unique for each skin part.

According to the list of activated taxel IDs and Table B.1, we distinguished between the activated (1) and not activated (0) regions, and thereby created the 42-dimensional binary vector per each record. The tactile data were transformed during the merging process and the scripts are included in the file `formater.py`.

4.1.2 Proprioceptive data

The original structure of the proprioceptive data is one angle per each DoF, that is to say 2×7 angles, from the corresponding range according to Table A.1. The data representing the action that enters into the configuration state was obtained by the difference between the successive states, i.e., the angle of change of each DoF.

$$action = next\ proprio - current\ proprio \quad (4.1)$$

In order to unify the data, we encoded each angle by a population of neurons (see Section 3.3.1). At first, we used Gaussian encoding, because it is the most common in neural modeling. We used 20 Gaussian neurons for encoding each angle (DoF). Actions were encoded using 15 Gaussian neurons, due to the smaller range (-25, 25) degrees.

The parameters of the Gaussian tuning curves were chosen as follows. The centers of the Gaussians μ_i were uniformly distributed along the corresponding range. The standard deviation σ was computed as a half of the distance between the centers of the two neighbouring Gaussians.

Later, we encountered a problem (to be mentioned in Section 4.2.1), which we solved using the sigmoidal tuning curves instead of Gaussians to encode only the target values. The input remained the same. The encoding of the output and the input values does not have to be identical; the network can still learn the mapping. The number of the sigmoidal tuning curves, 20, was chosen on the basis of the best performance of the network.

The implementation of the encoding class is located in the file `encoder.py`. The class is, afterwards, imported as a module into the main file containing the multilayer perceptron named `mlp.py`. In the function `read_data()` in `mlp.py`, the data is read and processed by the encoder module, before the training.

4.2 Training

In order to successfully train the network, we need to find its optimal parameters to minimize the prediction error. Typical parameters that can vary in a neural network are e.g. the number of neurons in the hidden layer and the learning rate (α). For the best performance, we also needed to find a proper amount of neurons encoding the angles of DoF.

To evaluate the performance of the network, we compute the mean squared error (MSE) of the network by comparing the outputs \mathbf{y} to the desired responses (targets) \mathbf{d} . The formula for MSE is

$$\text{MSE} = \frac{1}{n} \sum_{i=1}^n (y_i - d_i)^2 \quad (4.2)$$

where n is the number of output neurons. During the training, we will usually calculate the MSE on both, the training and the testing data, to evaluate the performance on previously unseen (testing) data, too, and plot them as two independent curves.

4.2.1 Encoding problem

As mentioned in Section 4.1.2, we encountered a problem with encoding of the targets. The (average) MSE, plotted in Figure 4.3(a), did not get below 0.05. Therefore, the backward reconstruction from the activations of population of neurons to the resulting angle was nearly impossible. An example output compared to target encoded by Gaussian tuning curves is shown in Figure 4.2.

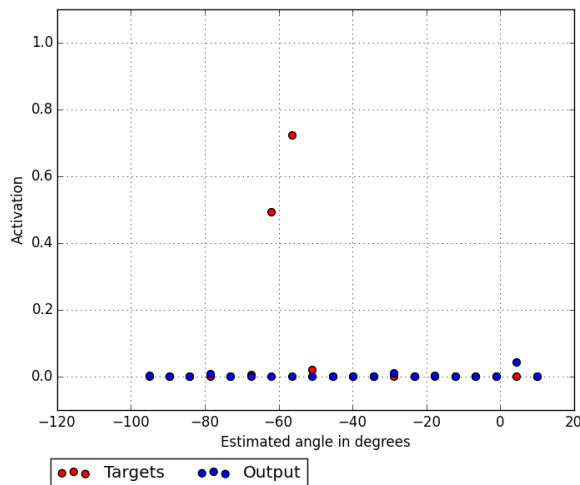


Figure 4.2: *Output and the corresponding target encoded by Gaussian tuning curves.*

Later, we found a solution to this problem. The Gaussian distribution of the output was not consistent with the sigmoid activation function, since the neuron that required the highest activation often had a smaller value than was the output of a neuron with sigmoid activation function, which is limited to 1. Thus, we used sigmoidal tuning

curves, instead of Gaussians, to encode the target angles of DoF. After that, the MSE decreased to 0.01 as shown in Figure 4.3(b).

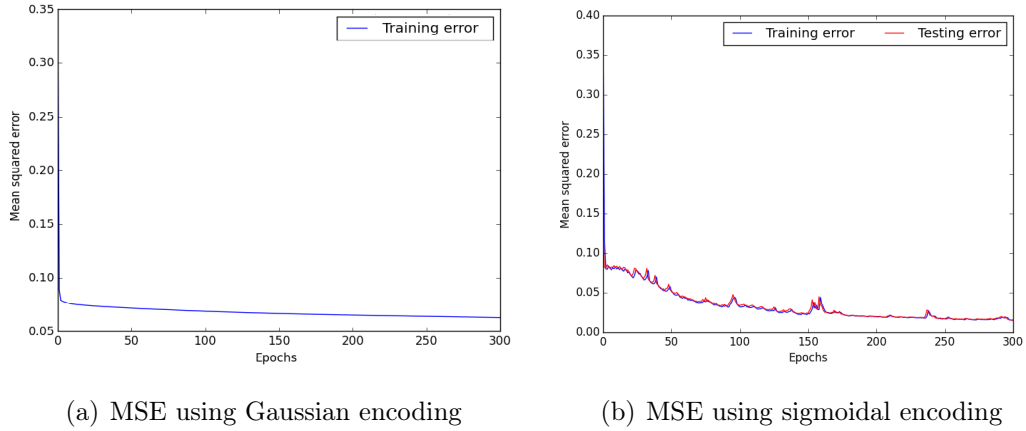


Figure 4.3: Comparison of training errors in case of Gaussian and sigmoidal population coding.

4.2.2 Setting the parameters

The optimal parameters of the population coding were described in Section 4.1.2. After many trials, the optimal number of hidden neurons turned out to be 50. We set the learning rate $\alpha = 0.001$, what smoothed the learning curve (MSE). The final model of the multilayer perceptron is shown in Figure 4.4.

We have tried a variant of a neural network with and without a bias unit. Eventually, adding bias only to the input layer worked the best. In Figure 4.3(b) the optimal variant is used. For illustration, in Figure 4.5(a) we added bias to both, the input and the hidden layer, in Figure 4.5(b) no bias was used.

4.2.3 Backward reconstruction of sigmoidal population coding

In order to get the final output, from the network output, we needed to reconstruct the angles of DoF from the neurons representing sigmoidal population coding. In Figure 4.6, there are depicted the activations in the output layer responsible for encoding the angle of the first DoF and the corresponding encoded target value.

The first DoF takes values from the range $[-95, 10]$ (see Table A.1), which is specified on the x -axis. The target value of the angle was in this case -60.9° , which was encoded by 20 sigmoidal tuning curves and is drawn in Figure 4.6 using red dots. The network output is illustrated by the blue dots.

If we applied the inverse of the sigmoid function (equation 4.3) to all activations of neurons, we would get 20 different values (representing angles), from which only a few

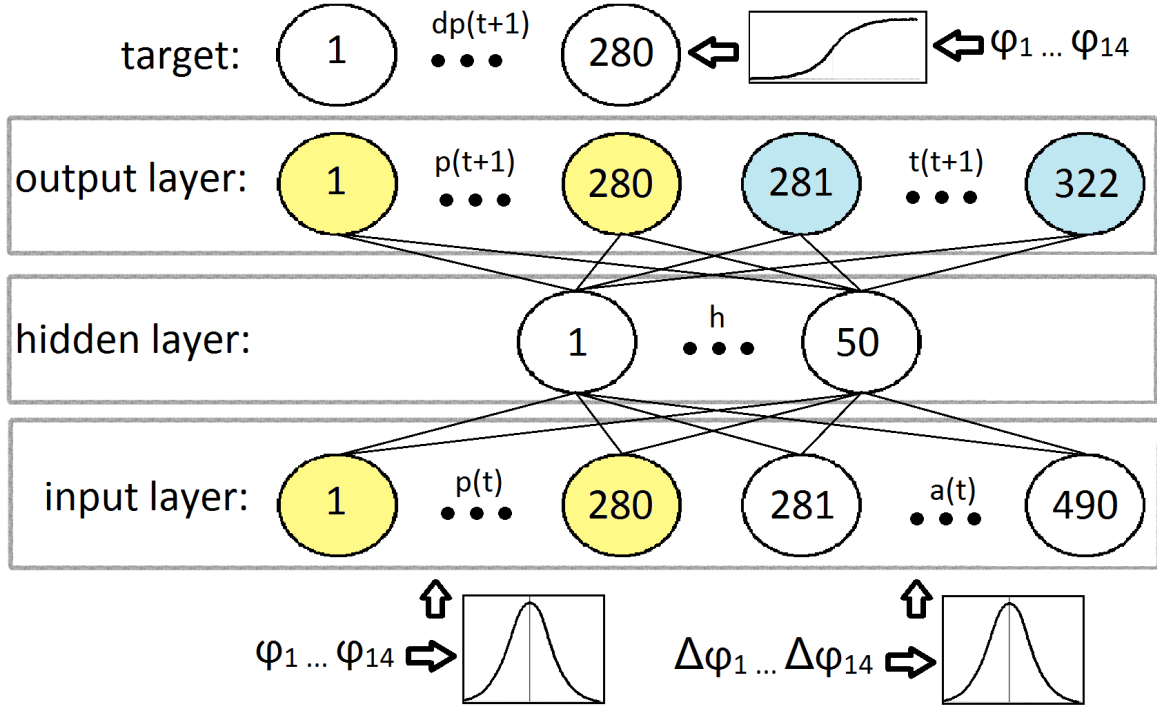


Figure 4.4: Model of the multilayer perceptron with corresponding numbers of neurons. φ_i is the angle of i -th DoF, $\Delta\varphi_i$ is the change of the angle of i -th DoF, $p(t)$ is the proprioceptive input at time t , $a(t)$ is the action at time t , h denotes the hidden neurons, $t(t+1)$ is the tactile output at time $t+1$ and $dp(t+1)$ is the desired proprioceptive output at time $t+1$. For clarity, we depicted the population coding, to allow the reader to see, that the encoding and decoding take place outside the network and do not take a part in the network operations. For simplicity, we did not depict the desired tactile output in the figure, since there is nothing special happening.

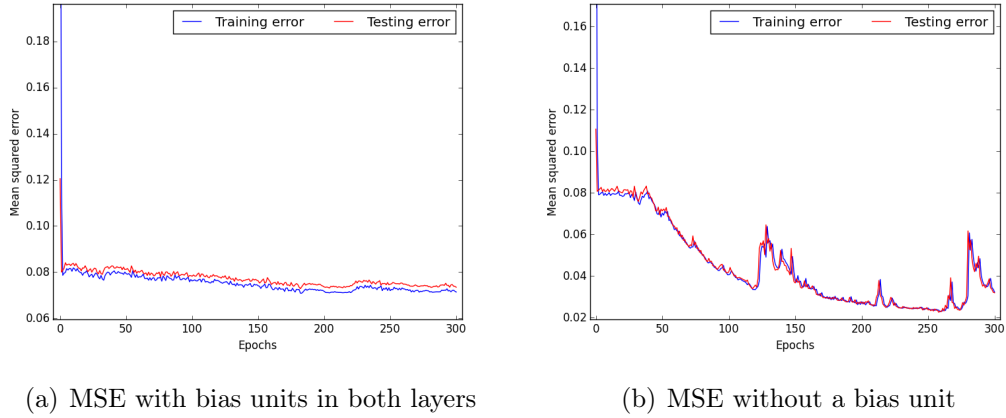
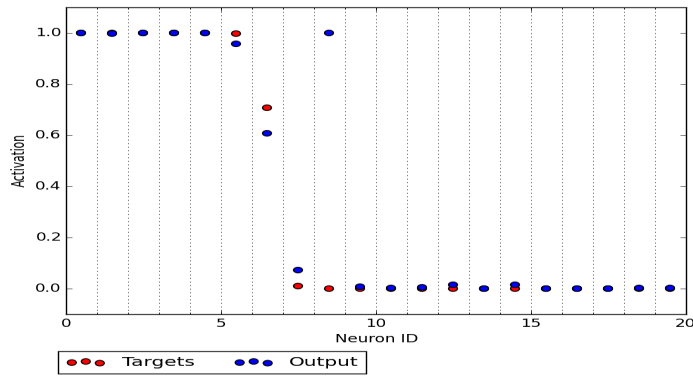
would be useful. This is because if the values close to 0 (or 1) of the sigmoid function differ only slightly, the corresponding value on the x -axis (the angle) could differ a lot, due to the nonlinearity of the function (steeper slope).

$$x = \ln\left(\frac{y}{1-y}\right) + \text{offset} \quad (4.3)$$

Therefore, we find a neuron with the activation closest to 0.5 and apply the inverse of the sigmoid function, with the offset value specified by the chosen neuron (and the range matching the given DoF), to its activation. The offset is calculated as

$$\text{offset} = \text{min} + i/(n-1) \quad (4.4)$$

where l is the length of the range of the given DoF, min is the minimal value from the range, i is the index of the chosen neuron and n is the number of the coding neurons (sigmoidal tuning curves). Decoding of the sigmoidal neurons is implemented in the file `encoder.py`.

Figure 4.5: *MSE with and without bias units.*Figure 4.6: *Encoded target of the first DoF compared to the output. On x-axis are depicted the 20 output neurons representing proprioception of the first DoF (targets are encoded using 20 sigmoidal tuning curves). Y-axis shows the activation of the neurons.*

We will demonstrate the evaluation on the example given in Figure 4.6. The neuron with activation closest to 0.5 was the 7-th (indexing from 0, that is 6) neuron with its activation 0.6. Using the formulas above, we get the following equations:

$$\text{offset} = -95 + 6(105/(20 - 1)) = -61.8 \quad (4.5)$$

$$x = \ln\left(\frac{0.6}{1 - 0.6}\right) - 61.8 = -61.4 \quad (4.6)$$

The decoded value of the angle of the first DoF is then -61.4° , while the desired value was -60.9° .

Chapter 5

Results

In this chapter we will analyze the results of the implemented predictive model.

5.1 The general ability of learning

In this section, we will analyze the general ability of our model of multilayer perceptron to learn. As we previously showed, in Figure 4.3(b), the MSE of the network on the training and also on the testing dataset lowered within the growing number of epochs, until it reached approximately 0.01 MSE on a neuron.

For instance, we compared an example output to its target before and after the network was trained. In Figure 5.1(a), we can see the desired output of the last DoF (wrist yaw on the right hand), compared to the output of an untrained network. The results after we trained the network are shown in Figure 5.1(b).

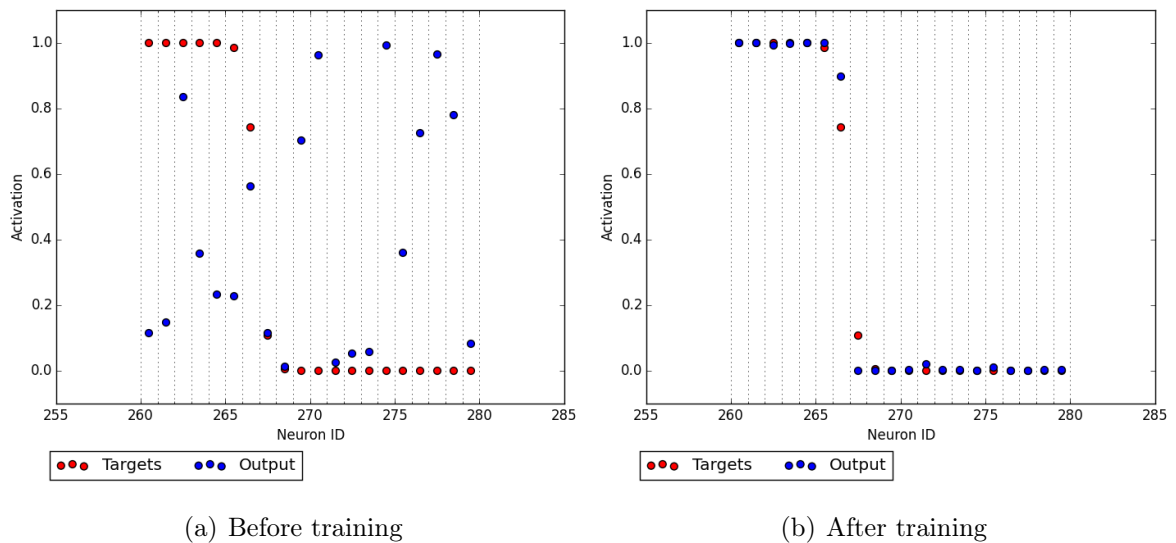
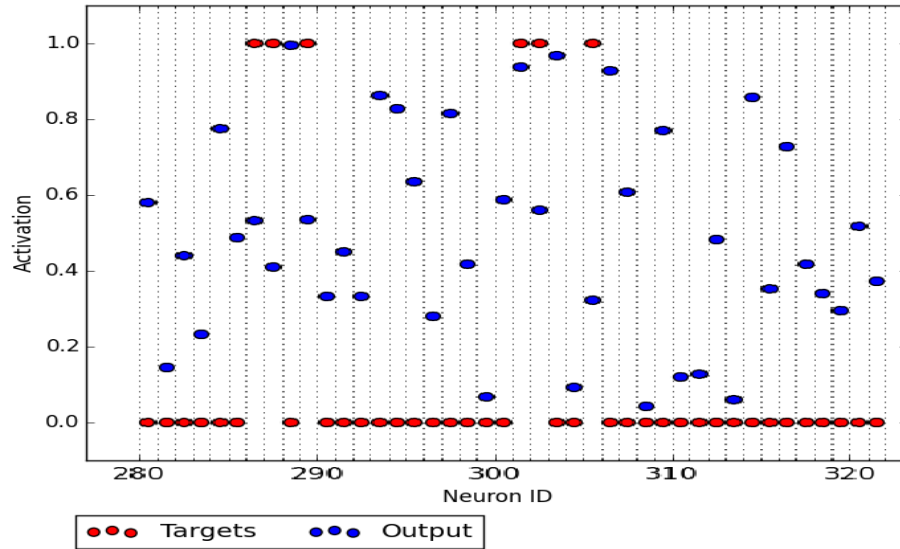
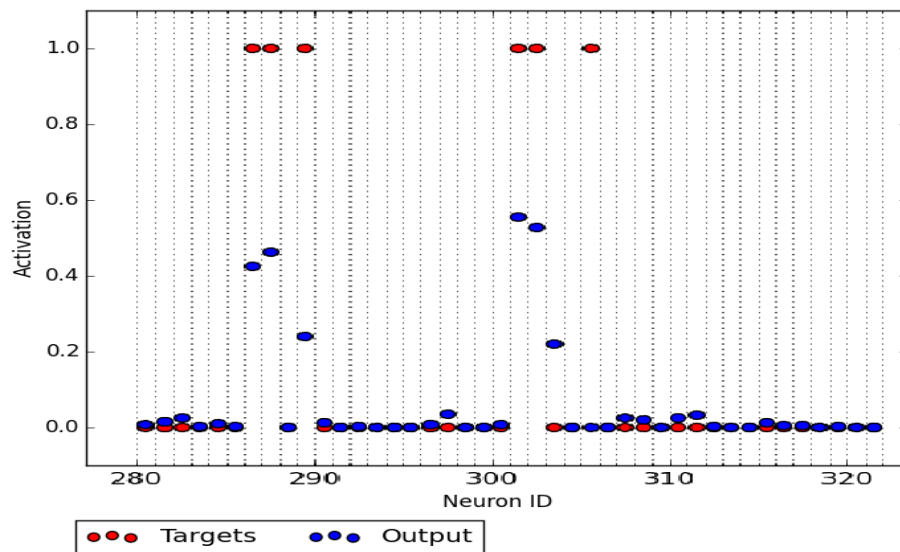


Figure 5.1: *Output representing the last DoF (20 neurons with the corresponding IDs) and the target before and after training.*

The tactile output compared to its target before and after the training is depicted in Figure 5.2. As we can see, the activations of the activated taxels do not reach as high values as the desired ones, however, they acquire much greater values than those, which were not activated.



(a) Before training



(b) After training

Figure 5.2: *Tactile output and the target before and after training, on example data. The x-axis marks the 42 neurons representing the tactile output, with the corresponding IDs.*

5.2 Prediction of touch

In order to evaluate the ability of the network to predict touch, we made several tests. In Figure 5.3, we can see the average activation of each tactile region in the whole dataset (red) and the average activation of each region predicted on the basis of each proprioceptive configuration from the whole dataset (green). The most were represented the fingers, due to the double-touch configuration, where the fingers of one arm always touch the other arm.

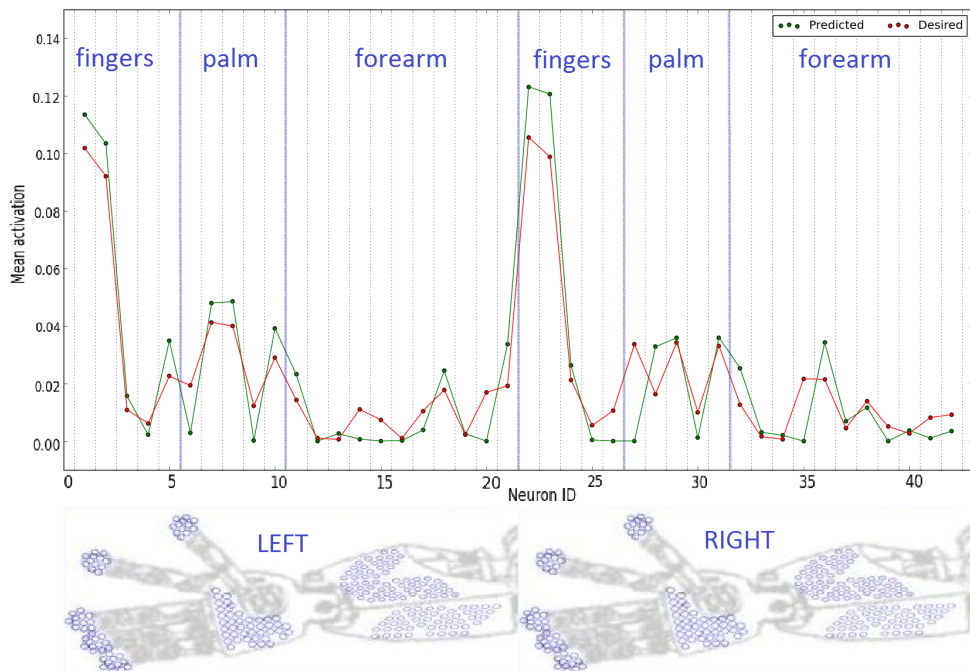


Figure 5.3: Average tactile activation on output neurons compared to the expected average activation.

An interesting fact is the generalization ability of the network that predicts touch a while before and a while after the touch really occurs. That is because the proprioceptive configuration of the iCub’s body is very similar to those, when the touch occurs. We can see that in Figure 5.4, depicting the average activation of neurons in a given time. The values on x -axis mark the successive numbers of the records in the dataset that represent a period of time. The green plot representing the average predicted activation rises before the red plot and descends after. That means the network can anticipate tactile stimuli *before the actual touch*.

On the trained network, we tried to predict the tactile output from the input configuration that causes a touch of iCub’s left fingers and right forearm without any action (an action that equals zero on each DoF). According to the desired and the actual output, shown in Figure 5.5, the network has the ability to connect the two modalities,

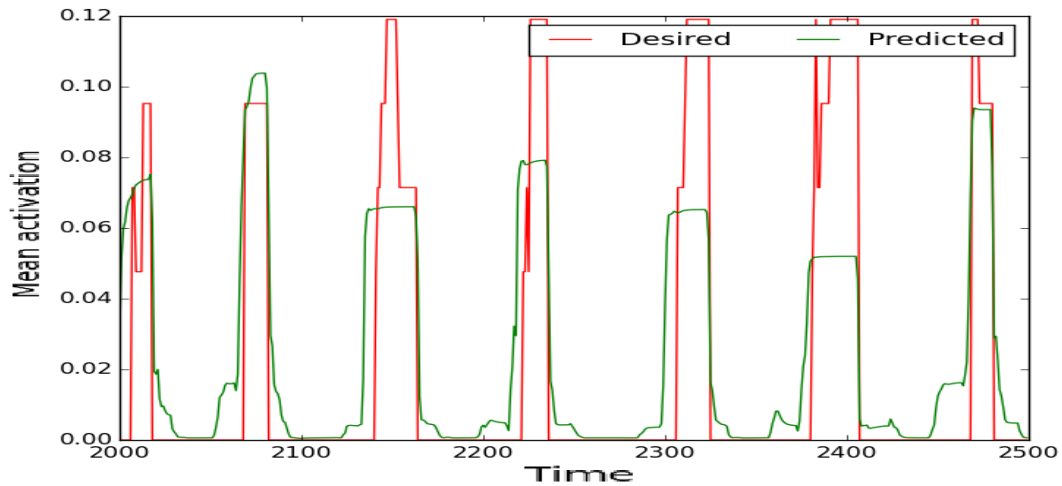


Figure 5.4: Average tactile activation of (all tactile) neurons in time compared to the expected average activation. Time is represented by successive numbers of records of the data. The selected period of time shows 7 trajectories ending by double-touch that are represented by the corresponding peaks.

proprioceptive and tactile, by successfully predicting the touch out of a configuration that causes it. Yet, the output neurons still do not acquire as high activation as desired, which is the same as in Figure 5.2, where there was an action (not equal to zero) in the input. However, the important thing is that these neurons have activations that are significantly higher compared to other neurons, which can be interpreted as correct answer.

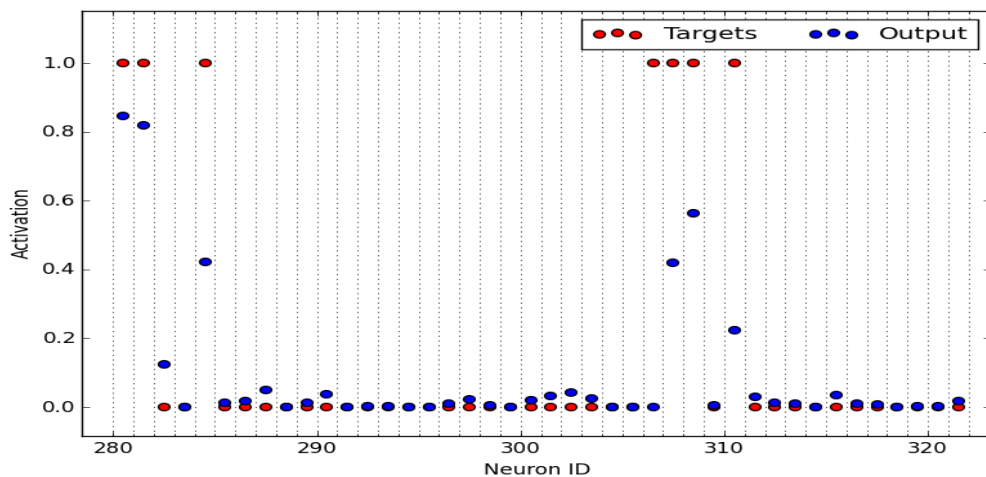


Figure 5.5: Tactile output compared to the targets. The x-axis marks the 42 neurons representing the tactile output, with the corresponding IDs.

Conclusion

The goal of this thesis was to create a simplified body schema, a multimodal representation of the iCub’s body that combines two modalities carrying somatosensory information, proprioception and tactile sensation. In biology, this corresponds to an early developmental stage when a baby learns about its body through the self-exploration. The implemented model was supposed to predict tactile stimuli and the next configuration of the body based on the current configuration and current movement.

The former idea was to include the tactile stimuli to the input of the network and examine the associative capabilities of the network. Nevertheless, we decided not to, because the input of the tactile stimuli had low average activation and, probably, would not bring forth a lot of additional information to the network. Therefore, the weights between the tactile information and the hidden layer would, probably, vanish to zero.

We decided to demonstrate this representation by the movement and self touch of iCub’s arms, therefore we did not consider other parts of its body in our model. For the implementation, we employed a multilayer perceptron with one hidden layer. The data used for training and testing were generated using the iCub simulator. The tactile sensory information was mapped to a 42-dimensional vector that represented the tactile regions on iCub’s fingers, palms and forearms. For the representation of the proprioceptive information, we used population coding. The input configuration along with the action were encoded using Gaussian tuning curves, while the desired output configuration was encoded using sigmoidal tuning curves.

In conclusion, the predictive model demonstrated the ability to learn the mapping between the proprioceptive configuration and the representation of the tactile stimuli. In addition, it was able to learn the forward kinematics of the robot and to predict an incoming self touch. However the neurons representing the predicted tactile output did not acquire as high values as the targets (which were binary). To figure this out, we might try using discrete activation function for the evaluation of the tactile output instead of sigmoid activation function. It also may be caused by the mean of the tactile data, which was closer to 0 (see Figure 5.3).

For the future work, we could try different representations of the somatosensory modalities, include other modalities like vision, or use deeper networks, that could learn to associate them. It is our belief that the exploration of the predictive model

of proprioceptive body representation can help the development of more complex representations that associate the proprioceptive and tactile stimuli and contribute to progress in autonomous cognitive robots.

Appendix A







Ranges of DoF of iCub's arm

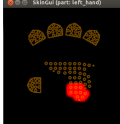






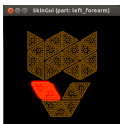

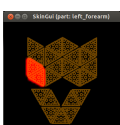
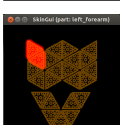

Degree of freedom	Min	Max	Range
shoulder pitch	-95	10	105
shoulder roll	0	161	161
shoulder yaw	-37	80	117
elbow	16	106	122
wrist pronosupination	-90	90	180
wrist pitch	-90	0	90
wrist yaw	-20	40	60
hand finger adduction/abduction	0	60	60
thumb opposition	10	90	80
thumb proximal flexion/extension	0	90	90
thumb distal flexion	0	180	180
index proximal flexion/extension	0	90	90
index distal flexion	0	180	180
middle proximal flexion/extension	0	90	90
middle distal flexion	0	180	180
ring and little finger flexion	0	270	270









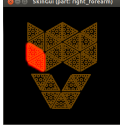



Table A.1: *Ranges of all degrees of freedom of iCub's arm in degrees (Bednárová, 2015).*

Appendix B

Tactile regions

Neuron ID	Taxel ID vector	Skin GUI
Left/Right fingers		
1/22	(0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11)	
2/23	(12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23)	
3/24	(24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35)	
4/25	(36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47)	
5/26	(48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59)	
Left palm		
6	(96, 97, 98, 99, 102, 103, 120, 129, 130)	
7	(100, 101, 104, 105, 106, 113, 116, 117)	
8	(108, 109, 110, 111, 112, 114, 115, 118, 142, 143)	
9	(121, 122, 123, 124, 125, 126, 127, 128)	

10	(132, 133, 134, 135, 136, 137, 138, 140, 141)	
Right palm		
27	(96, 97, 98, 100, 101, 110, 111, 112)	
28	(99, 102, 103, 104, 105, 106, 127, 129, 130)	
29	(108, 109, 113, 114, 115, 116, 117, 118, 142, 143)	
30	(120, 121, 122, 123, 124, 125, 126, 128)	
31	(132, 133, 134, 135, 136, 137, 138, 140, 141)	
Left forearm		
11	(288, 289, 290, 291, 292, 293, 295, 296, 297, 299, 300, 301, 302, 303, 304, 305, 307, 308, 309, 311, 348, 349, 350, 351, 352, 353, 355, 356, 357, 359)	
12	(204, 205, 206, 207, 208, 209, 211, 212, 213, 215, 336, 337, 338, 339, 340, 341, 343, 344, 345, 347)	
13	(252, 253, 254, 255, 256, 257, 259, 260, 261, 263, 312, 313, 314, 315, 316, 317, 319, 320, 321, 323)	
14	(0, 1, 2, 3, 4, 5, 7, 8, 9, 11, 180, 181, 182, 183, 184, 185, 187, 188, 189, 191)	
15	(24, 25, 26, 27, 28, 29, 31, 32, 33, 35, 12, 13, 14, 15, 16, 17, 19, 20, 21, 23)	
16	(156, 157, 158, 159, 160, 161, 163, 164, 165, 167, 144, 145, 146, 147, 148, 149, 151, 152, 153, 155)	

17	(132, 133, 134, 135, 136, 137, 139, 140, 141, 143, 168, 169, 170, 171, 172, 173, 175, 176, 177, 179)	
18	(120, 121, 122, 123, 124, 125, 127, 128, 129, 131, 60, 61, 62, 63, 64, 65, 67, 68, 69, 71)	
19	(96, 97, 98, 99, 100, 101, 103, 104, 105, 107, 108, 109, 110, 111, 112, 113, 115, 116, 117, 119)	
20	(84, 85, 86, 87, 88, 89, 91, 92, 93, 95, 72, 73, 74, 75, 76, 77, 79, 80, 81, 83)	
21	(36, 37, 38, 39, 40, 41, 43, 44, 45, 47, 48, 49, 50, 51, 52, 53, 55, 56, 57, 59)	
Right forearm		
32	(288, 289, 290, 291, 292, 293, 295, 296, 297, 299, 300, 301, 302, 303, 304, 305, 307, 308, 309, 311, 348, 349, 350, 351, 352, 353, 355, 356, 357, 359)	
33	(204, 205, 206, 207, 208, 209, 211, 212, 213, 215, 336, 337, 338, 339, 340, 341, 343, 344, 345, 347)	
34	(252, 253, 254, 255, 256, 257, 259, 260, 261, 263, 312, 313, 314, 315, 316, 317, 319, 320, 321, 323)	
35	(0, 1, 2, 3, 4, 5, 7, 8, 9, 11, 180, 181, 182, 183, 184, 185, 187, 188, 189, 191)	
36	(24, 25, 26, 27, 28, 29, 31, 32, 33, 35, 12, 13, 14, 15, 16, 17, 19, 20, 21, 23)	
37	(156, 157, 158, 159, 160, 161, 163, 164, 165, 167, 144, 145, 146, 147, 148, 149, 151, 152, 153, 155)	
38	(132, 133, 134, 135, 136, 137, 139, 140, 141, 143, 168, 169, 170, 171, 172, 173, 175, 176, 177, 179)	





39	(120, 121, 122, 123, 124, 125, 127, 128, 129, 131, 60, 61, 62, 63, 64, 65, 67, 68, 69, 71)	
40	(96, 97, 98, 99, 100, 101, 103, 104, 105, 107, 108, 109, 110, 111, 112, 113, 115, 116, 117, 119)	
41	(84, 85, 86, 87, 88, 89, 91, 92, 93, 95, 72, 73, 74, 75, 76, 77, 79, 80, 81, 83)	
42	(36, 37, 38, 39, 40, 41, 43, 44, 45, 47, 48, 49, 50, 51, 52, 53, 55, 56, 57, 59)	

Table B.1: *Distribution of tactile regions on the skin surface with the corresponding taxel IDs. For simplicity, the neuron IDs used are marked within the range [1, 42], not [281, 322] as we use in our model of multilayer perceptron.*

Bibliography

- Bednářová, N. (2015). Reprezentácia proprioceptívnych vstupov humanoidného robota iCub pomocou samoorganizujúcich sa máp. *Bakalárska práca*, Fakulta elektrotechnická, České vysoké učení technické v Praze.
- Bruchi, A., Cardellino, A., Fitzpatrick, P., Maggiali, M., Randazzo, M., Metta, G., Natale, L., Nori, F., Pattacini, U., Scalzo, A., and Vernon, D. (2016). The iCub manual. <http://wiki.icub.org/wiki/Manual/>. [Online; accessed 2-May-2016].
- Davidson, P. R. and Wolpert, D. M. (2005). Widespread access to predictive models in the motor system: a short review. *Journal of Neural Engineering*, 2(3):S313.
- Farkaš, I. (2011). Neural networks. <http://cogsci.fmph.uniba.sk/~farkas/courses/NeuralNets/neural-networks.2.pdf>.
- Haykin, S. S. (2009). *Neural networks and learning machines*, volume 3. Pearson Education Upper Saddle River.
- Hoffmann, M. (2014). Minimally cognitive robotics: Body schema, forward models, and sensorimotor contingencies in a quadruped machine. In *Contemporary Sensorimotor Theory*, pages 209–233. Springer.
- Longo, M. R., Azañón, E., and Haggard, P. (2010). More than skin deep: body representation beyond primary somatosensory cortex. *Neuropsychologia*, 48(3):655–668.
- Metta, G., Natale, L., Nori, F., Sandini, G., Vernon, D., Fadiga, L., Von Hofsten, C., Rosander, K., Lopes, M., Santos-Victor, J., et al. (2010). The icub humanoid robot: An open-systems platform for research in cognitive development. *Neural Networks*, 23(8):1125–1134.
- Nguyen-Tuong, D. and Peters, J. (2011). Model learning for robot control: A survey. *Cognitive Processing*, 12(4):319–340.
- Parmiggiani, A., Maggiali, M., Natale, L., Nori, F., Schmitz, A., Tsagarakis, N., Victor, J. S., Becchi, F., Sandini, G., and Metta, G. (2012). The design of the iCub humanoid robot. *International Journal of Humanoid Robotics*, 9(04).

- Pouget, A., Dayan, P., and Zemel, R. (2000). Information processing with population codes. *Nature Reviews Neuroscience*, 1(2):125–132.
- Proske, U. and Gandevia, S. C. (2012). The proprioceptive senses: their roles in signaling body shape, body position and movement, and muscle force. *Physiological Reviews*, 92(4):1651–1697.
- Roncone, A., Hoffmann, M., Pattacini, U., and Metta, G. (2014). Automatic kinematic chain calibration using artificial skin: self-touch in the iCub humanoid robot. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 2305–2312.
- Rumelhart, D. E., McClelland, J. L., Group, P. R., et al. (1986). *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, vol. 1-2. Cambridge, MA.
- Varga, M. (2016). Emulácia kože v simulátore humanoidného robota icub a autonómna explorácia tela. *Diplomová práca*, Fakulta matematiky, fyziky a informatiky, Univerzita Komenského v Bratislave.
- Zdechovan, L. (2012). Modelovanie uchopovania objektov pomocou neurónových sietí v robotickom simulátore iCub. *Diplomová práca*, Fakulta matematiky, fyziky a informatiky, Univerzita Komenského v Bratislave.