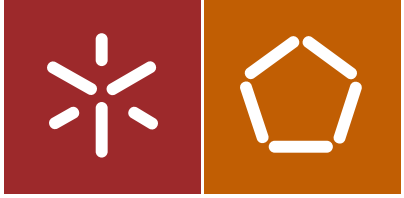


Universidade do Minho
Escola de Engenharia

Emanuel Augusto Freitas de Sousa

On Learning and Generalizing Representations in a Dynamic Field Based Architecture for Human-Robot Interaction

Novembro de 2014



Universidade do Minho
Escola de Engenharia

Emanuel Augusto Freitas de Sousa

On Learning and Generalizing Representations in a Dynamic Field Based Architecture for Human-Robot Interaction

Tese de Doutoramento
Programa Doutoral em Engenharia Eletrónica e de Computadores

Trabalho efectuado sob a orientação de
Professora Doutora Estela Bicho
Professor Doutor Wolfram Erlhagen

Novembro de 2014

STATEMENT OF INTEGRITY

I hereby declare having conducted my thesis with integrity. I confirm that I have not used plagiarism or any form of falsification of results in the process of the thesis elaboration.

I further declare that I have fully acknowledged the Code of Ethical Conduct of the University of Minho.

University of Minho, ___ /___ / _____

Full Name: Emanuel Augusto Freitas de Sousa

Signature: _____

This page has been intentionally left blank.

Acknowledgements

I wish to thank several people that, through their knowledge and support, contributed to this thesis or helped me in some way during my PhD years.

Firstly, I would like to address my scientific advisers Prof. Dr. Estela Bicho and Prof. Dr. Wolfram Erlhagen. During these years, I have had the opportunity to benefit from their guidance and support, their vast scientific expertise and, most importantly, their friendship. To them goes my deepest gratitude.

I am thankful to my friends and colleagues of the MARLab, at the University of Minho, namely, (alphabetically) Carlos Faria, Carolina Vale, Eliana Silva, Flora Ferreira, Gianpaolo Gulleta, Luís Louro, Miguel Cunhal, Miguel Sousa, Nzoji Hipólito, Rui Silva, Simão Antunes, Tiago Malheiro, Toni Machado and Weronika Wojtak. Much of the work described here, specially the real-world implementations in the robot ARoS would not have been possible without the cooperative working dynamics that we maintain at the lab. Beyond that, I believe I am deeply fortuned for sharing a workspace with such an intelligent, friendly and supportive group of people.

I would also like to thank to Dr. Alfredo Pereira from the Psychology Department at the University of Minho that, although not directly connected to this work, has granted me his support and advice during the final phase of the work and thesis writing.

At last, I cannot even begin to express my gratitude to my parents and brothers. The patience and care they have provided me over the years has been the greatest of gifts.

The work was supported by a Portuguese FCT grant SFRH/BD/48529/2008 financed by POPH-QREN 4.1-Advanced Training, co-funded by the European Social Fund and national funds from MEC.

Abstract

On Learning and Generalizing Representations in a Dynamic Field Based Architecture for Human-Robot Interaction

Due to the increasing demand for adaptive robots able to assist humans in their everyday tasks, furnishing robots with learning abilities is one of the most important goals of current robotics research. The work reported in this thesis is focused on the integration of learning capacities in an existing Dynamic Field based control architecture developed for natural human-robot collaboration. Specifically, it addresses two important serial order problems that appear in the architecture at distinct but closely coupled levels of abstraction: 1) the learning of the sequential order of sub-goals that has to be followed to accomplish a certain task, and 2) the learning of representations of motor primitives that can be chained to achieve a certain sub-goal.

A model based on the theoretical framework of Dynamic Neural Fields (DNFs) is developed that allows the robot to acquire a multi-order sequential plan of a task from demonstration by human tutors. The model is inspired by known processing principles of human serial order learning. Specifically, it implements the idea of two complementary learning systems. A fast system encodes the sequential order of a single demonstration. During periods of internal rehearsal, it acts as a teacher for a slow system that is responsible for extracting generalized task knowledge from memorized demonstrations of different users.

The efficiency of the learning model is tested in a real world experiment in which the humanoid robot ARoS learns the plan of an assembly task by observing human tutors executing possible sequential orders of sub-goals.

An extension of the basic model is also proposed and tested in a real-world experiment. It addresses the fundamental problem of a hierarchical encoding of complex sequential tasks. It is shown how verbal feedback by the tutor about a serial order error may lead to the autonomous development of a neural representation of a group of sub-goals forming a sub-task.

The second serial order problem of learning goal-directed chains of motor primitives is addressed by combining the associative learning mechanism of the dynamic field model with self-organizing properties. Inspired by the basic idea of the Kohonen's map algorithm, it is shown how self-organizing principles can be exploited to develop field representations of motor primitives, like for instance, a specific grasping behaviour, from observed motion trajectories. Moreover, the integration of additional contextual cues (e.g. object properties) in the learning process may cause the splitting of an existing motor primitive representation into two new representations that are context sensitive. In model simulations, it is shown that the learning mechanisms for representing sequential task knowledge in the DNF model can be also applied to establish chains of motor primitives directed towards a final goal (e.g. reach-grasp-place). Such a chained organization has been discussed in the neurophysiological literature to support not only a fluent execution of known action sequences but also the cognitive capacity of inferring the goal of observed motor behaviour of another individual.

Resumo

Aprendizagem e Generalização de Representações numa Arquitetura Baseada em Campos Dinâmicos para Interação Humano-Robô

Devido à procura crescente por robôs adaptativos capazes de auxiliar humanos nas suas tarefas diárias, um dos mais importantes objetivos da investigação em robótica atual é o de dotar robôs com a capacidade de aprender. O trabalho apresentado nesta tese foca-se na integração de capacidades de aprendizagem numa arquitetura de controlo, baseada em campos dinâmicos, para colaboração fluente entre Humano e Robô. Especificamente são abordados dois problemas importantes relacionados com ordem sequencial que surgem na arquitetura em níveis de abstração distintos embora relacionados: 1) a aprendizagem da ordem sequencial de sub-objetivos que tem de ser seguida para completar uma determinada tarefa e 2) a aprendizagem de representações de primitivas motoras que podem ser encadeadas para atingir um certo sub-objetivo.

Foi desenvolvido um modelo baseado em teoria de Campos Dinâmicos Neuronais (CDNs) que permite ao robô adquirir um plano sequencial multi-ordem de uma tarefa a partir de demonstrações de tutores humanos. O modelo é inspirado em princípios conhecidos da aprendizagem da ordem sequencial por humanos. Especificamente, implementa a ideia de dois sistemas de aprendizagem complementares. Um mecanismo rápido codifica a ordem sequencial de uma demonstração única. Durante períodos de repetição interna, este mecanismo age como professor de um sistema mais lento responsável por

extrair conhecimento generalizado da tarefa a partir das demonstrações de diferentes utilizadores.

A eficiência do modelo de aprendizagem é testada numa experiência em cenário real na qual o robô humanoide ARoS aprende o plano de uma tarefa de montagem através da observação de tutores humanos que executam possíveis ordens sequenciais de execução dos sub-objetivos.

Uma extensão do modelo básico é também proposta e testada em ambiente real. A extensão aborda o problema fundamental da codificação hierárquica de tarefas sequenciais complexas. É mostrado como feedback verbal fornecido pelo tutor acerca de erros na sequência pode levar ao desenvolvimento autónomo de uma representação neuronal de um grupo de sub-objetivos que formam uma sub-tarefa.

O segundo problema de ordem sequencial, que consiste na aprendizagem de cadeias de primitivas motoras direcionadas a um objetivo, é abordado através da combinação do mecanismo de aprendizagem associativa do modelo baseado em Campos Dinâmicos com propriedades de auto-organização. Partindo da ideia fundamental do algoritmo do mapa de Kohonen, é mostrado como princípios de auto-organização podem ser explorados para desenvolver representações, em campos dinâmicos, de primitivas motoras, como por exemplo, um gesto específico de agarrar, a partir de trajetórias de movimentos observados. Além disso, a integração de informação contextual adicional (e.g. propriedades do objeto) no processo de aprendizagem pode causar a divisão de uma representação de uma primitiva motora em duas novas representações específicas de cada contexto. É mostrado em simulação que os modelos de aprendizagem para representação do conhecimento da tarefa sequencial no modelo baseado em CDNs podem ser também aplicados na formação de cadeias de primitivas motoras direcionadas a um objetivo final (e.g. aproximar-agarrar-colocar). A organização em cadeia tem sido discutida na literatura sobre neurofisiologia, como sendo o suporte não só da execução fluente de ações sequenciais conhecidas, mas também da capacidade cognitiva de inferir o objetivo do comportamento motor observado num outro indivíduo.

Contents

I Introduction **1**

1 Introduction **3**

1.1 Cognitive robotics and Human-Robot Interaction 4

1.2 The DNF approach to Human-Robot Interaction 5

1.3 Learning 8

1.3.1 "Learning-from-Demonstration" 8

1.3.2 LfD in Human-Robot Interaction: What to learn? 9

1.3.3 Sequence learning 11

1.3.4 Evolving new neuronal representations of observed motor primitives 13

1.4 Contributions of this thesis 15

1.5 Thesis outline 17

II Materials and methods **19**

2 Anthropomorphic Robotic System (ARoS) **21**

2.1 General description 21

2.2 Vision system 23
2.3 Arm and hand 23
2.4 Speech recognition/synthesis 24

3 Dynamic Neural Fields **25**

3.1 Behaviour and representations 25
3.2 The dynamical systems approach to cognition 27
3.3 The Amari neural field model 31
3.3.1 Detection 34
3.3.2 Memory and forgetting 35
3.3.3 Decision making 37

4 Artificial Neural Networks **39**

4.1 Why neural networks 39
4.2 Basic functioning 41
4.3 Network organization 44
4.4 Types of learning 47
4.5 Hebbian learning 48
4.6 Delta rule 50
4.7 The Self-Organizing Map 50
4.8 Neural Networks and Dynamic Neural Fields 54
4.8.1 Activity propagation 54
4.8.2 Adaptation rules 56
4.9 Conclusion 59

III State of the art **61**

5 Learning sequential tasks from demonstration in robotics **63**

5.1 General overview 63

5.2 Precedence graphs 65

5.2.1 Precedence analysis 66

5.2.2 Petri-Nets 67

5.2.3 Hidden Markov Models 68

5.3 Hierarchical representations 69

5.3.1 Hierarchy based on task specific knowledge 70

5.3.2 Guidance from a tutor 70

5.3.3 Graphs and hierarchies 71

5.4 Final remarks 71

6 Models of sequential behavior: an overview **73**

6.1 Main conceptual approaches 73

6.1.1 The chaining approach 74

6.1.2 Ordinal theory 75

6.1.3 Positional theory 76

6.2 Connectionism: a major trend 78

6.3 Hierarchical models 79

6.4 Application to robotics 81

6.5 Final remarks 82

IV Sequence Learning **83**

7 Model of learning sequential tasks - Design **85**

7.1 General overview 86

7.1.1 Goal representations 86

7.1.2 The sequence model paradigm 88

7.1.3 The rehearsal mechanism 89

7.1.4 Chunking representations 90

7.2 Learning sub-goal sequences 90

7.2.1 The vision input 91

7.2.2 The Sequence Learning Layer (SL) 91

7.2.3 The social learning paradigm 97

7.2.4 Mathematical formulation of the model 98

7.3 Short-Term Memory Layer (STM) - The rehearsal mechanism 102

7.3.1 Mathematical formulation of the rehearsal mechanism 105

7.4 Extending the model for chunking sub-goals into a sub-task representation 106

7.4.1 The Task Monitoring Layer (TM) 107

7.4.2 High-level Cognitive Memory Layer (HCM) 108

7.4.3 Mathematical formulation of the chunking mechanism 112

8 Model of learning sequential tasks - Implementation and results **117**

8.1 Learning of a sequential task and the rehearsal mechanism 118

8.1.1 Experimental setup 118

8.1.2 First experiment: learning a sequence 121

8.1.3 Second experiment: learning an alternative sequence 124

8.1.4 Third experiment: errors occurrence and correction 126

8.1.5 Time differences between demonstration and internal rehearsal . 132

8.2 Development of sub-task representations 134

8.2.1 Forming a representation in High-level Cognitive Memory Layer (HCM) Layer 135

8.2.2 Fourth experiment: developing a second representation in the HCM Layer 136

8.3 Discussion 139

V Emerging neuronal representations of motor primitives 143

9 Model for the development of neural representations - Design 145

9.1 General Overview 146

9.1.1 Evolving neuronal representations in Dynamic Neural Fields . . . 146

9.1.2 Integrating Self-Organizing Maps and Dynamic Neural Fields . . 148

9.1.3 Chaining motor primitives 148

9.2 Segmentation 150

9.3 Developing representations of primitives 153

9.3.1 Input normalization 153

9.3.2 Variable transformation 154

9.3.3 Complement coding 155

9.3.4 The context signal 157

9.3.5 Self-Organizing Layer 158

9.4 Encoding the sequence of primitives 161

10 Model for the development of neural representations - Implementation and results 167

10.1 Simulation inspired by Mirror Neurons 167

10.2 Setup 168

10.3 First phase - Description 170

10.4 First phase - Results 173

10.4.1 Segmentation 173

10.4.2 Primitive encoding 173

10.4.3 Sequence learning 177

10.5 Second phase - Description 178

10.6 Second learning phase - Results 180

10.6.1 Encoding new primitives 180

10.6.2 Adjusting the sequences 182

10.7 Effect of the Back-propagation 183

10.8 Discussion 185

VI Conclusion and future work 189

11 Conclusion and future work 191

11.1 Future work 192

VII Appendices **195**

A Implementation parameters of Chapter 8 **197**

A.1 Sequence Learning Layer (SL) 197

A.1.1 Past field u^{pa} 197

A.1.2 Present field u^{pr} 198

A.1.3 Learning rule 198

A.2 Short-Term Memory Layer (STM) 198

A.3 Task Monitoring Layer (TM) 199

A.4 High-level Cognitive Memory Layer (HCM) 199

A.4.1 Past field $u^{pa_{HL}}$ 199

A.4.2 Present field $u^{pr_{HL}}$ 200

A.4.3 Learning rules 200

B Implementation parameters of Chapter 10 **201**

B.1 Self-organizing field u^{som} 201

B.1.1 Self-organizing learning rule 201

B.2 Sequence Learning Layer (SL) 202

B.2.1 Past field u^{pa} 202

B.2.2 Present field u^{pr} 202

B.2.3 Learning rule between past and present fields 203

This page has been intentionally left blank.

List of Abbreviations

HRI	Human-Robot Interaction.
MARLab	Mobile and Anthropomorphic Robotics Laboratory.
ARoS	Anthropomorphic Robotic System.
RL	Reinforcement Learning.
AI	Artificial Intelligence.
HMM	Hidden Markov Models.
AOL	Action Observation Layer.
ASL	Action Simulation Layer.
IDL	Intention Detection Layer.
AEL	Action Execution Layer.
OML	Object Memory Layer.
CSGL	Common Sub-goals Layer.
EML	Error Monitoring Layer.
HRL	Hierarchical Reinforcement Learning.
TOTE	"test-operate-test-exit".
ANN	Artificial Neural Networks.
LMS	Least-Mean Square.
SRN	Simple Recurrent Network.
RNN	Recurrent Neural Network.
OSCAR	Oscillator-based Associative Recall.
SEM	Start-End Model.
CQ	Competitive Queueing.

DNF	Dynamic Neural Field.
LfD	Learning-from-Demonstration.
SOM	Self-Organizing Map.
PFC	Prefrontal Cortex.
SL	Sequence Learning Layer.
TM	Task Monitoring Layer.
HCM	High-level Cognitive Memory Layer.
STM	Short-Term Memory Layer.
SOL	Self-Organizing Layer.
MSV	Mean Square Velocity.
MN	Mirror Neurons.
YARP	Yet Another Robotic Platform.
HLC	High Level Cognition.
fMRI	Functional Magnetic Resonance Imaging.
DoF	Degrees of Freedom.

Nomenclature

ϕ_{tar}	Direction of a target in a navigation scenario
ϕ_{obs}	Direction of an obstacle in a navigation scenario
x	Spatial dimension
t	Time variable
τ	Time constant
s	Neuronal input
u	Neuronal activation
h	Neuronal resting level
W	Interaction kernel in a dynamic neural field
$f_\lambda(.)$	Heaviside function with horizontal shift λ
$f_0(.)$	Heaviside function with horizontal shift 0
A	Amplitude of a gaussian curve
w_{inhib}	Negative part of an interaction kernel
σ	Standard deviation of a gaussian curve
B	Amplitude of the negative component of a *Mexican-Hat* kernel function
α	Parameter for controlling the zero crossings of a sigmoidal kernel

q_i Value of an input node i

a_i Weight of a synaptic connection in a single neurons model

m Total number of input nodes in a discrete neural network

$\varphi(\cdot)$ Activation function of a neuron

$a_{i,j}$ Weight of the synaptic connection of input i to neuron j

$\mathbf{A}$ Matrix of synaptic weights

$\Delta a_{i,j}$ Change applied to weight of synaptic connection $a_{i,j}$

$\gamma(\cdot)$ Variation function of a synaptic weight

η Rate parameter for an neural network learning rule

u^{tar} Target value of a neuronal activation

$j^{win}(\mathbf{q})$ Winning neuron from a Self-Organizing Map (SOM) for an input vector q

$\Delta \mathbf{a}_j$ Variation of the vector $\mathbf{a}_j$ corresponding to the weights associated with the neuron j of the SOM

σ_0 Initial width of the kernel function in the SOM algorithm

x^{in} Spatial dimension of a neural network input

x^{out} Spatial dimension of a neural network output

$e(\cdot)$ Difference between target neuronal activation and total input

τ_a Time scale of the differential equation defining a synaptic weight variation

ζ^k_{stoch} Stochastic noise

x^{pa} Spatial dimension of the past field in Sequence Learning Layer (SL)

$u^{pa}\big(x^{pa}, t\big)$ Neural field activation of the past field of in the Sequence Learning Layer (SL)

$s^{pa}\big(x^{pa},t\big)$ Input to field $u^{pa}\big(x^{pa},t\big)$

$v^{pa}\big(x^{pa},t\big)$ Contribution from the vision system to the input s^{pa}

C^{pa} Gain parameter controlling the strength of contribution v^{pa} to s^{pa}

$h^{pa}\big(x^{pa},t\big)$ Resting level of field u^{pa}

h^{pa}_{bas} Variable encoding the baseline value of $h^{pa}\Big(x^{pa},t\Big)$

H^{pa}_{low} Lower baseline value of $h^{pa}\Big(x^{pa},t\Big)$

H^{pa}_{high} Higher baseline value of $h^{pa}\Big(x^{pa},t\Big)$

H^{pa}_{dec} Value toward which $h^{pa}\Big(x^{pa},t\Big)$ decays when $u^{pa}\big(x^{pa},t\big)>0$

$\tau_{h^{pa}}$ Variation rate of $h^{pa}\Big(x^{pa},t\Big)$

x^{pr} Spatial dimension of the present field in Sequence Learning Layer (SL)

$u^{pr}\big(x^{pr},t\big)$ Neural field activation of the present field in the Sequence Learning Layer (SL)

$s^{pr}\big(x^{pr},t\big)$ Input to field to $u^{pr}\big(x^{pr},t\big)$

$v^{pr}\big(x^{pr},t\big)$ Contribution from the vision system to the input s^{pr}

C^{pr} Gain parameter controlling the strength of contribution v^{pr} to s^{pr}

C^{rec} Gain parameter controlling the strength of contribution from $u^{stm}\big(x^{stm},t\big)$ to $s^{pr}\big(x^{pr},t\big)$, during recall

C^{lea} Gain parameter controlling the strength of contribution from $u^{stm}\big(x^{stm},t\big)$ to $s^{pr}\big(x^{pr},t\big)$, during learning

b^{lea} Flag active when the model is learning

$\underset{pr\to pa}{E}\big(x^{pr},x^{pa},t\big)$ Excitatory predefined synaptic connections from $u^{pr}\big(x^{pr},t\big)$ to $u^{pa}\big(x^{pa},t\big)$

$\underset{pa\to pr}{I}\big(x^{pa},x^{pr},t\big)$ Inhibitory predefined synaptic connections from $u^{pa}\big(x^{pa},t\big)$ to $u^{pr}\big(x^{pr},t\big)$

$\underset{pa \to pr}{a}\left(x^{pa}, x^{pr}, t\right)$ Adaptable synaptic connections from u^{pa} to u^{pr}

τ_a^{pa} Time scale associated to the connections $\underset{pa \to pr}{a}$

$e^{pr}\left(x^{pr}, t\right)$ Difference between desired and actual value of the activation $u^{pr}\left(x^{pr}, t\right)$

λ^{pr} Learning threshold applied on field u^{pr}

λ^{pa} Learning threshold applied on field u^{pa}

x^{stm} Spatial dimension of the field in Short-Term Memory Layer (STM)

$u^{stm}\left(x^{stm}, t\right)$ Neural field activation of the Short-Term Memory Layer (STM)

$s^{stm}\left(x^{stm}, t\right)$ Input to field to $u^{stm}\left(x^{stm}, t\right)$

$v^{stm}\left(x^{stm}, t\right)$ Contribution from the vision system to the input s^{stm}

C^{stm} Gain parameter controlling the strength of contribution v^{stm} to s^{stm}

$h^{stm}\left(x^{stm}, t\right)$ Resting level of field $u^{stm}\left(x^{stm}, t\right)$

H_{bas}^{stm} Baseline value of $h^{stm}\left(x^{stm}, t\right)$

$\tau_{h^{stm}}$ Time scale of $h^{stm}\left(x^{stm}, t\right)$

x^{tm} Spatial dimension of field in Task Monitoring Layer (TM)

$u^{tm}\left(x^{tm}, t\right)$ Neural field activation of the field Task Monitoring Layer (TM)

$s^{tm}\left(x^{tm}, t\right)$ Input to field $u^{tm}\left(x^{tm}, t\right)$

$h^{tm}(t)$ Resting level of the field $u^{tm}\left(x^{tm}, t\right)$

H_{high}^{tm} Higher value of h^{tm}

H_{low}^{tm} Lower value of h^{tm}

$x^{pa_{HL}}$ Spatial dimension of the past field in High-level Cognitive Memory Layer (HCM)

$u^{pa_{HL}}\left(x^{pa_{HL}}, t\right)$ Neural field activation of the past field in the High-level Cognitive Memory Layer (HCM)

$s^{pa_{HL}}\left(x^{pa_{HL}},t\right)$ Input to field $u^{pa_{HL}}\left(x^{pa_{HL}},t\right)$

$x^{pr_{HL}}$ Spatial dimension of the present field in High-level Cognitive Memory Layer (HCM)

$u^{pr_{HL}}\left(x^{pr_{HL}},t\right)$ Neural field activation of the present field of the High-level Cognitive Memory Layer (HCM)

$s^{pr_{HL}}\left(x^{pr_{HL}},t\right)$ Input to field $u^{pr_{HL}}\left(x^{pr_{HL}},t\right)$

$\lambda^{pr_{HL}}$ Learning threshold applied on field $u^{pr_{HL}}$

$\Lambda_{low}^{pr_{HL}}$ Lower value of the learning threshold $\lambda^{pr_{HL}}$

$\Lambda_{high}^{pr_{HL}}$ Higher value of the learning threshold $\lambda^{pr_{HL}}$

$\underset{pr_{HL}\rightarrow pr}{a}\left(x^{pr_{HL}},x^{pr},t\right)$ Excitatory adaptable synaptic connections from $u^{pr_{HL}}\left(x^{pr_{HL}},t\right)$ to $u^{pr}\left(x^{pr},t\right)$

$\tau_a^{pr_{HL}}$ Time scale of the connections $\underset{pr_{HL}\rightarrow pr}{a}$

$e^{pr_{HL}}$ Difference between desired and actual value of the activation $u^{pr_{HL}}$

$\underset{pa\rightarrow pa_{HL}}{a}\left(x^{pa},x^{pa_{HL}},t\right)$ Adaptable synaptic connections form $u^{pa}\left(x^{pa},t\right)$ to $u^{pa_{HL}}\left(x^{pa_{HL}},t\right)$

$\tau_a^{pa_{HL}}$ Variation rate of the connections $\underset{pa\rightarrow pa_{HL}}{a}$

$\underset{tm\rightarrow pr_{HL}}{R}\left(x^{tm},x^{pr_{HL}},t\right)$ Random, excitatory predefined synaptic connections from $u^{tm}\left(x^{tm},t\right)$ to $u^{pr_{HL}}\left(x^{pr_{HL}},t\right)$

$e^{pa_{HL}}\left(x^{pr_{HL}},t\right)$ Difference between desired and actual value of the activation $u^{pr_{HL}}$

N^{reh} Number of sequence rehearses occurring during the rehearsal process of sequence learning

x_{RE} Location of emerging representation in $u^{pa_{HL}}$ and $u^{pr_{HL}}$

$\theta_j(t)$ Position of joint j at time t

λ_{low}^{msv} Lower threshold value of MSV

λ_{high}^{msv} Higher threshold value of MSV

t_p^{st} Starting time of a primitive p

t_p^{end} Ending time of a primitive p

$\langle \cdot \rangle$ Average value

$n_{j_{min}}$ Trajectory whose minimum value of θ is the minimum of a set of trajectories

$t_{j_{min}}$ Time instant when the minimum value of a $n_{j_{min}}$ is reached

$n_{j_{max}}$ Trajectory whose maximum value of θ is the maximum of a set of trajectories

$t_{j_{max}}$ Time instant when the maximum value of a $n_{j_{max}}$ is reached

$\theta_{j_{min}}$ Minimum value of $n_{j_{min}}$

$\theta_{j_{max}}$ Maximum value of $n_{j_{max}}$

x^{jt} Transformation of the time based variable $\theta_j(t)$ to a spatial dimension

v^{cmp} Complement coding of a vector v

$Q_j(x^{jt})$ Input to the SOM from the joint j

κ^{hue} Function representing a color describing a context

x^{hue} Spatial dimension of the field encoding the Hue value

$u^{hue}\big(x^{hue}, t\big)$ Neural Field activation of the field encoding the Hue value

$Q^{hue}(x^{hue})$ Input to the SOM from the contextual signal κ^{hue}

x^{obs} Spatial dimension encompassing the totality of the sensory input

$Q\big(x^{obs}\big)$ Input function encompassing the totality of the sensory input

$\hat{Q}\big(x^{obs}\big)$ Normalization of function Q

$u^{som}\big(x^{som}, t\big)$ Neural Field activation of the Self-Organizing DNF

$s^{som}\big(x^{som},t\big)$ Input to field $u^{som}\big(x^{som},t\big)$

$\underset{obs\to som}{a}\big(x^{obs},x^{som},t\big)$ Adaptable synaptic connections from x^{obs} to $u^{som}\big(x^{som},t\big)$.

$z^{obs}\big(x^{obs},t\big)$ Transformation of the spatial dimension encompassing the totality of the sensory input

C^{som} Gain parameter for regulating the input strength of input $s^{som}\big(x^{som},t\big)$

$c^{obs}\big(x^{obs},t\big)$ Centroid of the input $Q\big(x^{obs},t\big)$

This page has been intentionally left blank.

List of Figures

1.1 Sketch of the cognitive architecture for human-robot joint action 7

2.1 Anthropomorphic Robotic System (ARoS) 22

3.1 Phase plot of a dynamical system 28

3.2 Examples of field activations in a model for autonomous navigation . . 30

3.3 Interaction between neurons. 32

3.4 Interaction kernels 33

3.5 Examples of input detection and no detection 35

3.6 Memory and forgetting in a DNF 36

3.7 Examples of field decisions 38

4.1 Diagram of a single neuron 42

4.2 Linear separation of input space 43

4.3 Example of network with multiple output neurons 44

4.4 Single-layer and multi-layer networks 46

4.5 Example of recurrent neural network 47

5.1 Graph depicting a task with seven subtasks 65

5.2 Hierarchical organization in three levels 69

7.1 Diagram of the connections between the fields u^{pa} and u^{pr} 87

7.2 Diagram of the Toy Vehicle 88

7.3 Diagram of the learning process 89

7.4 Schematic view of the DNF model for learning sequential tasks 91

7.5 Examples of the visual input 92

7.6 Diagram of the Sequence Learning Layer (SL). 93

7.7 Interaction process between neurons in u^{pa} and u^{pr} for $h^{pa}_{bas} = H^{pa}_{low}$ . . 95

7.8 Interaction process between neurons in u^{pa} and u^{pr} for $h^{pa}_{bas} = H^{pa}_{high}$. 96

7.9 Learning with different $\tau_{h^{pa}}$ 97

7.10 Example of STM pattern 104

7.11 Diagram of the extended model with the TM and the HCM Layers . . . 107

7.12 Associations between the SL and HCM Layers 110

7.13 Time course of the activation in the HCM Layer 111

8.1 Experimental setup 119

8.2 Activation profile of the STM layer after the first demonstration 122

8.3 Time course of population activity during "Observational Learning" and "Internal Rehearsal" 123

8.4 Snapshots of the first experiment: Active Recall 124

8.5 Activation profile of SL: recall after the first demonstration 124

8.6 Activation profile of the SL: recall after the second demonstration. . . . 125

8.7 Snapshot of the second experiment: Active Recall 126

8.8 Integrals of all newly established connection weights that link active populations in u^{pa} to the two populations GC and MC in u^{pr} 126

8.9 Activation profile of the Sequence Learning Layer (SL) Layer: recall after the third demonstration. 127

8.10 Activation profile of SL: first error. 128

8.11 Graphic depiction of the integral of weights connecting active populations after the third demonstration 128

8.12 Probability graph of the decision process in the SL Layer following the first demonstration of the third sequence. 129

8.13 Snapshot of the third experiment: first error 129

8.14 Activation profile of the SL: second error 130

8.15 Probability graph of the decision process in the SL Layer following the second demonstration of the third sequence 130

8.16 Activation profile of the SL: correct learning 131

8.17 Activation profile of SL: correct recall of the Top Platform 131

8.18 Probability graph of the decision process in the SL Layer following the third demonstration of the third sequence 132

8.19 Time Evolution of u^{pa} during the second (a) and third (b) rehearsal periods of the third demonstration 133

8.20 Comparison of the time intervals between activations in the past field during observation and rehearsal 134

8.21 Time evolution of fields u^{pr}, u^{tm} and $u^{pr_{HL}}$ during an error trial 136

8.22 Time evolution of fields u^{tm}, $u^{pa_{HL}}$ and $u^{pr_{HL}}$ during the correction of an error 137

8.23 Snapshot of the fourth experiment: demonstration 138

8.24 Probability trees of the decision process for the first sub-goal after learning the higher-level sub-task representation 139

9.1 Examples of joint trajectory and Mean Square Velocity (MSV) 152

9.2 Process of complement coding 156

9.3 Example of a contextual input 157

9.4 Model for primitives encoding 159

9.5 Effect of the back-propagation 162

9.6 Model for chaining representations of primitives 163

10.1 Example of motions employed in the first phase 171

10.2 Trajectories of θ_2 172

10.3 MSV of an arm trajectory 173

10.4 Segmented trajectories 174

10.5 Emerging field activations 175

10.6 Centroid and standard deviation of the representations 176

10.7 Minimum distance between centroids. 177

10.8 Sequence field activations during a trial 178

10.9 Prediction error 179

10.10 Example of motions employed for the second learning phase 180

10.11 Centroid and standard deviation of the representations - second phase . 181

10.12 Splitting of populations in u^{som} 182

10.13 Minimum distance between centroids - second phase 183

10.14 Prediction error - second phase 183

10.15Centroid and standard deviation of the representations - no back-propagations184

This page has been intentionally left blank.

Part I

Introduction

Chapter 1

Introduction

The main goal of this thesis is to provide new insights on how learning capabilities can be integrated into a dynamic neural field based cognitive architecture for control of an anthropomorphic robot able to engage in joint action with a human partner. The architecture was developed to allow humans to interact with the robot in a natural, that is, human-like manner (Bicho et al., 2010, 2011a). Ideally, this "human-like" approach to Human-Robot Interaction (HRI) should be also applied to the learning of new skills and tasks, since new generations of assistive robots are supposed to work with ordinary people without programming knowledge. The work presented in this thesis follows the logic of a social learning approach in which the robot acquires knowledge through demonstration and non-verbal and verbal interactions with a human tutor, in a similar way a toddler learns from interaction with his parents. Concretely, the thesis addresses two major challenges towards achieving this demanding goal.

The first challenge is the development of a model for learning and recall of sequential tasks. Most of our daily routine tasks are sequential in nature. It is therefore fundamental that, a socially aware robot, possesses a set of basic skills allowing it to learn sequential tasks by observing demonstrations from a human user.

The second challenge is the autonomous development of neuronal representations

within a dynamic neural field, using self-organizing principles. Here the thesis work adopts a neuro-inspired approach compatible with the idea implemented in Kohonen networks, that map sensory input patterns according to their similarity onto neuronal sub-populations. As a specific application, the learning dynamics is applied to the development of representations of observed motor primitives. Following principles of motor simulation, these representations can be chained, allowing the prediction of the outcome of complex motor sequences.

1.1 Cognitive robotics and Human-Robot Interaction

One of the most important lines of current robotics research is the interaction between humans and robots. This fact is motivated by a development pathway that is increasingly placing robots in non-traditional functions like nurse-caring (Pineau et al., 2003; Chen et al., 2010), search and rescue (Nourbakhsh et al., 2005; Murphy et al., 2008), entertainment industries (Weinberg and Driscoll, 2006; Solis et al., 2009) and scientific exploration of hazard environments (Kunz et al., 2009; Gulati et al., 2010), to name a few. These new tasks require that the robots share the environment with humans, contrarily to what has been the norm in classical industrial environments. More than just sharing the same space, robots are expected to be capable of collaborating with humans, in order to achieve common goals. This new trend places a spotlight on HRI, with an increasing demand for social aware robots capable of interacting with their human counterparts in a human-like way. This challenge has motivated researchers to develop specialized control approaches whose focus is not only on the motor capacities to execute a certain task but also on the social skills that support the intuitive and fluent coordination of actions and decisions with a human partner. In a review on social robotics, Fong et al. (2003) proposed a classification of these approaches in terms of design paradigm into two different groups: "functionally'" designed and "biological inspired".

Functionally designed approaches are the ones whose purpose is to produce an interaction experience that creates an impression of social intelligence, but whose imple-

mentation is not committed to any of the known psychological or biological principles supporting interactions in social species like humans and other primates. Because they are frequently based on heuristic rules, the interactions with humans are normally very situation specific. Examples of such robots can be found for instance in museums (Bennewitz et al., 2005; Faber et al., 2009), medical care environments (Tapus et al., 2009; McColl and Nejat, 2013), offices (Rosenthal et al., 2010), schools (Kanda et al., 2007) or in scenarios built for study of human-robot interaction. (Minato et al., 2004; Salichs et al., 2006).

Biologically inspired approaches are the ones whose design is grounded on known biological and cognitive mechanisms supporting social interaction in humans and other animals. The approach is based on the assumption that, in order for a robot to manifest human-like social competences, its control must be based on the cognitive processing principles that allow fluent human joint action. Since humans are fundamentally social beings, we have high expectations about intuitive and pleasant interactions with other agents. Examples of bio-inspired approaches to HRI can be found for instance in the works of Breazeal and colleagues (e.g. Breazeal et al., 2006, 2009) based on theory of mind (Whiten, 1991), Dautenhahn and colleagues (e.g. Otero et al., 2008; Dautenhahn et al., 2009) who explore concepts of developmental psychology or Bicho and colleagues (Erlhagen et al., 2006a,b; Erlhagen and Bicho, 2006; Bicho et al., 2011a,b) based on neural dynamics and motor resonance theory.

1.2 The DNF approach to Human-Robot Interaction

To address the issue of HRI in social scenarios, the Mobile and Anthropomorphic Robotics Laboratory (MARLab) at the University of Minho developed a cognitive robot control architecture inspired by recent findings about the neuro-cognitive mechanisms supporting joint action in humans (Bicho et al., 2008, 2009, 2010, 2011a,b). The architecture integrates two main processing principles. The first states that we are able to anticipate the action outcomes of others by internally simulating their motor behaviours

using our own motor repertoire, a process called motor resonance (Wilson and Knoblich, 2005). For instance, observing someone grasping a full cup by the handle activates the motor program corresponding to a drinking action, in the observer. The second principle asserts that decisions within the joint action space are the result of a dynamic process that continuously integrates information about the inferred goal of the partner (through motor resonance), contextual cues given by the surrounding environment and knowledge about the task shared between the two actors (Bicho et al., 2011a).

The architecture can be described as a distributed network of neuronal populations organized in interconnected layers, which are formalized as coupled Dynamic Neural Fields (DNFs). DNFs, were originally proposed to model the formation of activity patterns in neuro-cortical tissue (Wilson and Cowan, 1973; Amari, 1977) and later introduced to robotics by Schöner et al. (1995). DNFs implement the idea that strong recurrent intra-field interactions constitute a basic mechanism of neuronal processing. These recurrent connections allow localized neuronal pools to generate self-stabilized activity patterns capable of encoding and memorizing task-relevant information. Consistent with the motor resonance principle (Wilson and Knoblich, 2005), the architecture implements the mapping of the observed actions of a co-actor on the robot's motor repertoire, activating the corresponding neuronal representations within the DNFs (Figure 1.1). By internally simulating the observed sequence of primitives (e.g. reach-grasp-place) the robot is able to infer the motor intention of the co-actor. This prediction, together with contextual information and shared task-knowledge, is integrated in the decision process about an adequate complementary action towards the completion of the joint task. The architecture was implemented in ARoS (Anthropomorphic Robotic System), an anthropomorphic robot built at the University of Minho in the scope of the EU integrated project JAST [1] and a project funded by the Portuguese Foundation for Science and Technology (FCT)[2], as a test platform for human-robot interaction tasks (Silva et al., 2008). It was successfully validated in several human-robot interaction

[1] JAST "Joint action Science and Technology", project financed by the European Commission, (ref. IST-2-003747-IP.

[2] FCT ref. CONC-REEQ/17/2001

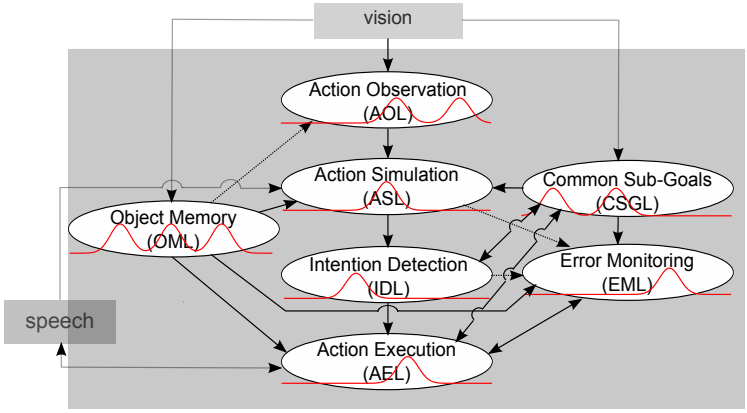


Figure 1.1: **Sketch of the cognitive architecture for human-robot joint action**. (See (Bicho et al., 2010)) Information from the vision system about the observed actions triggers the neuronal representations of the motor primitives in the Action Observation Layer (AOL), supplying the Action Simulation Layer (ASL) and allowing the detection of the intention performed in the Intention Detection Layer (IDL). The decision about the complementary action is made in the Action Execution Layer (AEL) and integrates the inferred intention (IDL), Contextual Cues (Object Memory Layer (OML)) and Shared Task Knowledge (Common Sub-goals Layer (CSGL)). The occurrence of errors is represented in the Error Monitoring Layer (EML).

tasks in the assembly and user assistance domain (see Bicho et al., 2012, for a review).

In the previous work, the task information including the possible sequential orders of task execution were predefined by the designer. This limits of course the ability of ARoS to adapt to new constraints and tasks. A promising approach for dealing with this open problem is to couple the DNF architecture with synaptic adaptation mechanisms, allowing the readjustment of the inter-field connections in response to the interaction of the robot with the environment including the human user. This would allow the robot to acquire new knowledge with only modest interventions of the human, who acts as a tutor (and not a programmer) in the same way humans learn from each other). It is expected that this learning competence will be beneficial for HRI in multiple ways. First of all, it would reduce the need of a complete pre-specification of the task, which is often difficult and time consuming. Secondly, it would allow ARoS to quickly learn different tasks directly from demonstrations, that could be performed by multiple users. Thirdly, pre-existing knowledge could be altered to reflect changes in task specification, or could be extended to incorporate new contextual variations, in a open-ended process, increasing the ability of ARoS to understand and execute new goal-directed actions. To

summarize, developing mechanisms for allowing the architecture to adapt to novel joint action scenarios is the logical next step to increase ARoS capabilities as a cooperative agent.

1.3 Learning

The question of learning poses a major challenge not just for the team working with ARoS, but for roboticists in general, specially for those working on the field of collaborative robotics (e.g. Breazeal et al., 2006; Pardowitz et al., 2007; Calinon et al., 2007; Saunders et al., 2007; Otero et al., 2008). A much higher degree of adaptability is expected from social aware robots (like ARoS) in comparison to their typical industrial counterparts (Fong et al., 2003). On the one hand, they must deal with complex human environments highly unstructured and constantly changing (take the simple example of the infinite number of dispositions in which tableware could be laid on a table). On the other hand, the tasks in which they are required to collaborate with their human partners are also inherently dynamic. Changes in context due to the user's decisions may require the same task to be performed by a different course of actions. For robots to be able to understand the meaning of goal-directed actions in face of changing environments and constantly varying tasks makes it highly desirable to furnish robots with the ability to learn action goals and the associated means to achieve it. In other words, the challenge is "*to develop a new generation of industrial and service robots and underpinning technologies, in particular enabling robotic systems to operate in dynamic real-world environments, reaching measurable improvements of abilities such as autonomy and adaptability and interacting in a safe way with with humans*" [3]. Ideally, the learning process should follow a human-like approach, with robots learning from humans in the same way humans learn from each other. This is an almost fundamental requisite, considering that most of the expected users of an assistive robot (e.g. elderly)

[3]citation from the HORIZON 2020 - Work program 2014-2015, topic "ICT 23 - 2014: Robotics" of the European Commission, (`http://ec.europa.eu/research/participants/portal/doc/call/h2020/common/1617606-part_5_i_ict_v2.0_en.pdf`

will not have the technical skills required for a classical programming approach.

1.3.1 "Learning-from-Demonstration"

Learning-from-Demonstration (LfD) is today one of the most popular approaches to robot learning. As already mentioned, the fact that it is an intuitive method than can in principle be used by non-expert users is one of its main advantages (Billard et al., 2008). But it also presents some attractive features when compared to other exploration based learning methods. For instance, Reinforcement Learning (RL) depends on precise reward functions than can often be very difficult to define in complex robotic scenarios. Also, RL requires the exploration of a wide part of the states-space, so that a control policy can be derived from the rewards (Argall et al., 2009). LfD on the contrary, reduces the complexity of search space, since teachers provide, in principle, good examples and avoid the bad ones. Also, by guiding the learning process instead of just letting the robot explore, the teacher can keep the focus on situations actually encountered during task execution, potentially reducing the necessary time for defining a good control policy (Saunders et al., 2007).

Due to its attractive features, LfD has been used in a wide number of robotic related tasks like autonomous navigation in complex unknown terrain (Ollis et al., 2007; Silver et al., 2010), autonomous helicopter flight (Coates et al., 2008), teaching of human gestures to humanoid robots (Calinon and Billard, 2007), object manipulation by robotic arms (Pastor et al., 2009), byped locomotion (Nakanishi et al., 2004) and teaching of motor primitives (Kober and Peters, 2009) to name just a few. These examples are mostly centred on the learning of low level skills ranging from navigation to motor control.

1.3.2 LfD in Human-Robot Interaction: What to learn?

In the scope of HRI, LfD has been mostly focused on the learning of task representations (Breazeal et al., 2006; Pardowitz et al., 2007). This tendency can be explained by

the fundamental role that shared-task knowledge plays in human-human and human-robot interaction. For instance, knowing which step of the task is to be taken next may facilitate the prediction of the partner's intention or allow one to timely prepare a complementary action. ARoS architecture represents a good example, since shared-task knowledge, which is encoded in the CSGL, supports both the intention detection process and the decision about complementary actions.

Learning a task representation often means encoding sequences of goals that must be achieved through a specific order. Depending on the task, this order may be fixed or may show some degree of variability, but if ignored may lead to failure in accomplishing the end-goal. Most of our daily routine tasks are sequential in nature (e.g., preparing the dinner table, cleaning, cooking, etc). A mechanism for correctly encoding and retrieving sequences of goal-directed actions is therefore a fundamental tool for creating a useful task representation for HRI.

Although several works have addressed the problem of sequence learning by developing algorithms for encoding of sequential tasks based on demonstrations (e.g. Nicolescu and Matarić, 2003; Pardowitz et al., 2005, 2007; Zöllner et al., 2005), only few have taken inspiration from cognitive models for sequence learning in humans (e.g. Sandamirskaya and Schöner, 2010). However, humans are remarkably efficient in the learning and execution of sequential tasks. Therefore, it seems to be a reasonable approach to try to build mechanisms for sequence learning that can be framed in the LfD paradigm and that are grounded on existing knowledge about learning mechanisms in the human brain. **The first main objective of this thesis is to develop a bio-inspired model for learning sequential tasks**.

The existence of goal-directed action sequences that combine motor primitives in a highly context sensitive manner raises the additional question of how these goal-directed representations can be autonomously learned. This is part of the broader question, already mentioned, about the developmental mechanisms supporting the emergence of new, that is, not totally predefined representations in the control architecture. To address this issue, some clues can be drawn by seeking inspiration from known brain

mechanisms. It is known that the cortical tissue has a remarkable capability for rapidly re-adapting its neuronal representations for encoding new knowledge. This property has been the target of several works (for an overview, see Hubel and Wiesel, 1977) which inspired the development of computational algorithms for self-organizing systems (Kohonen, 1982). These algorithms can provide some guidance for **the second main objective of this thesis: The development of a Self-organizing mechanism for emerging new representations encoded as neuronal populations in a DNF**.

The topics of sequence learning and the formation of neuronal representations are addressed in the next two subsections.

1.3.3 Sequence learning

Sequence generation plays a fundamental role in the complex activities in which we engage everyday and the mechanisms by which we are able to learn and retrieve sequences have long been a matter of research. It is known that several neuro-cognitive mechanisms like decision making, long and short-term memory and time monitoring play an important role (Houghton, 1990; Cooper and Shallice, 2000; Brown et al., 2000; Botvinick and Plaut, 2004). But the question of how these processes interact to generate sequential behaviour remains, to a large extent, unanswered. The debate spans from psychology to neuroscience, computational science and even robotics (Henson, 1998; Lewandowsky et al., 2006; Sandamirskaya and Schöner, 2010).

In 1951 Karl Lashley (1951) gave a seminal talk in which he argued that the recall of a sequence could not be explained as the simple retrieval of a chain of serially connected mental representations, where the sensory feedback of one would trigger the next. His insights were mostly based on empirical observations about speech production/recognition in different languages as well as typical errors in serial order tasks. His considerations marked the beginning of a wide interest in the topic, resulting in the appearance of several models of serial behaviour, either purely abstract or computationally applied (a more detailed review is given in Chapter 6). These models address several questions on serial behaviour, many of which could not be easily answered by the

simple chaining theory like for instance the encoding of repeated items (Henson, 1998; Burgess and Hitch, 1999; Brown et al., 2000). They also model empirical phenomena, like recency and primacy effects (the first and last items of a sequence are more easily remembered than the ones in the middle) (Henson, 1998; Brown et al., 2000). Still, some important aspects of sequential behaviour remain a matter of discussion, two of them being specially important in the context of this thesis: i) the ability of humans to learn sequences from a reduced number of observations, and ii) the hierarchical nature of many sequential tasks.

Fast vs slow learning mechanisms

Regarding the first topic, current knowledge about neural learning mechanisms suggests the existence of two complementary learning systems, one tuned to rapid encoding of new memories (the hippocampus) and the other to long-term consolidation of acquired memories (the neocortex). The information transition between these two systems has been the target of recent neurophysiological studies, many of them agreeing with the existence of strong correlation between activation patterns in these two brain areas (e.g. Takashima et al., 2009; van Kesteren et al., 2010). An important insight was given by studies performed in mice (Euston et al., 2007; Ji and Wilson, 2007) which compared the activation patterns of cell assemblies in hippocampus during a repetitive learning task and subsequent sleep periods. The studies showed that the sequential patterns of activation re-occur while the mice are sleeping, suggesting the existence of a fast-forward replay mechanism that plays a role in the consolidation of information in the neocortex networks.

This notion of an interplay between fast and slow learning networks inspired the learning system for ARoS, presented in this thesis. Although humans tend to accept that teaching someone, like a child, is often a slow and repetitive process, they normally are less tolerant when it comes to robots, expecting them to learn rapidly and in only a few demonstration trials (Ehrenmann et al., 2002). However, many computational learning processes like training neural networks or reinforcement learning paradigms are

based on iterative processes that are usually slow and require multiple examples. The challenge of efficient learning is approached in this thesis through the extension of the already mentioned model for learning of sequential tasks with a fast encoding mechanism that is used to avoid the need of repeated demonstrations from a human tutor. The internal rehearsal of memorised knowledge acts as a teacher signal for the slow learning system for extracting generalized task knowledge, thus avoiding the need of repeated demonstrations of the same task sequence.

Hierarchical task encoding

Many of the serial tasks we engage in are hierarchical in nature. Consider the preparation and serving of a coffee cup. One may choose to add first the coffee, then the milk and afterwards the sugar, but there is no restriction on changing the order through which these three ingredients are added. However, all these sub-goals must be completed before the cup is handed over. This sort of hierarchical organization into nested sets of goals and sub-goals is very common in goal-directed behavior (Lashley, 1951; Kurby and Zacks, 2008; Botvinick, 2008). Clusters of goal-related actions seem to be formed primarily according to their end-goal, disregarding the temporal and causal relations they may have. The purpose of this organization scheme is two-folded: On the one hand, it provides more flexibility in terms of the temporal order through which the sub-goals may be executed. On the other hand, it plays an important role in the context of action planning, allowing the preparation of actions according to the achievement of end-goals, even if the contribution of such actions to specific goals is not immediately clear but only becomes evident during the course of actions.

Although several aspects of hierarchically organized sequential behaviour can be properly modelled (Cooper and Shallice, 2006; Botvinick and Plaut, 2004), the neural mechanisms responsible for the clustering of goal-directed actions into more coarse units are mostly unknown. Developmental studies showed that infants at the young age of 24-36 months are already capable of perceiving and imitating the hierarchical structure present in sequential actions (Travis, 1997; Whiten et al., 2006), even with little apriori

knowledge of the task structure. This suggests that bottom-up processes play a major role in the formation of hierarchically structured task representations.

From the robotics point of view, a control architecture able to learn a task hierarchy through observation would be extremely valuable, allowing the robot to be more flexible in performing the task, specially in the context of human-robot interaction occurring in unstructured scenarios (Sousa et al., 2014). More concretely, the thesis explores the idea that the formation of hierarchical structures is the result of an iterative learning process where prediction errors guide the process of clustering sub-goals into more coarse-grained units (Zacks et al., 2011). For the model this implies an additional extension where new sub-task representations forming clusters of sub-goals develop during learning and practice.

1.3.4 Evolving new neuronal representations of observed motor primitives

One of the earliest applications of DNFs in robotics was motivated by the need to add representation and memory to the dynamical systems approach to behaviour-based robotics (for a review see Erlhagen and Bicho, 2006). In navigation tasks, the relevant behavioural dimension is the heading direction. The use of DNFs spanned over this dimension allows for instance to maintain a representation of target direction even if information about its location is no longer available due to noisy sensory input or occluding obstacles. A self-established bump of activation centred over this direction replaces in the DNF approach the symbol in the classical Artificial Intelligence (AI) approach. This and other properties of the field dynamics have made DNFs a popular alternative to implement cognitive functions defined over behavioural variables like location in space (Bicho et al., 2000; Sousa et al., 2012), objects features (Faubel and Schöner, 2006) or color (in HSV space) (Sandamirskaya and Schöner, 2010).

The approach followed in the cognitive architecture of ARoS is different in the sense that the dimensions being coded on the DNFs are abstract and not intrinsically metric. This raises the question of how to establish a topology for assigning the localized

neuronal populations to instances of the behavioural dimensions they represent. In the JAST project, since the focus was not on learning, the locations of the neural representations were predefined. However, to support the encoding of new knowledge not specified by the designer, a learning mechanism must be applied, enabling the recruitment of new populations to represent for instance new motor primitives. The challenge is to extract some higher-level metric information to be mapped on DNFs. Previous work on this topic (Erlhagen et al., 2007) focused on the development of a self-organizing mechanism for neuro-specialization. The approach described in this thesis follows a similar path but is inspired by the Self-Organizing Map (SOM) introduced by Kohonen (1982), as a model to explain the formation of topological organized cortical maps.

It is known that sensory inputs are mapped on the sensory cortex in such a way that the location of the corresponding activity reflects a degree of similarity between inputs (Hubel and Wiesel, 1977). Kohonen (1982, 1990) proposed a bio-inspired computational mechanism, capable of autonomously creating a map of a set of input patterns which is topologically ordered according to statistical features in the input patterns. The *Kohonen* principle is based on competitive learning, meaning that neurons compete to code for specific features of the input space. It also relies on lateral inhibition for the development of the topological distribution of representations, a feature that makes the Kohonen algorithm compatible with the DNF framework.

SOMs have been largely applied as a computational tool for extracting patterns from unstructured data-sets. Application examples spawn from fields as diverse as genetics (Tamayo et al., 1999), finance (Deboeck and Kohonen, 1998), bibliographic organization (Kohonen et al., 2000) or computer vision (Maddalena and Petrosino, 2008). In the work reported in this thesis, the SOM principles are used to cluster input patterns corresponding to sensory information. This sensory data describes observed angular trajectories of the arm joints that may for instance represent object-directed actions of a human tutor in robot teaching experiments. Thus, the mechanism assigns novel neuronal populations to represent observed motion primitives.

The development of representations of motor primitives provides a tool for ex-

ploring an important theory of cognition in humans and other primates that is also fundamental in ARoS control architecture, namely the intention detection through motor simulation. According to this theory, the inference of the end-goal of an action depends on the observation of a specific sequence of motor primitives. This is based on evidences supporting the notion that the representation of a motor primitive is specific to the end-goal of the action to which it belongs (Fogassi et al., 2005; Chersi et al., 2007; Erlhagen et al., 2007). The implication is that, for instance, the representation of the "grasping" primitive in a "reach-grasp-place" sequence is different from the one in a "reach-grasp-eat" sequence. Hence, by recognizing the specific sequence of observed primitives it is possible to anticipate its outcome. While in the ARoS architecture both the motor primitive representations and the motor sequences were predefined, here the emergence of motor primitives is combined with the sequence learning model to predict and execute goal-directed actions (see Chapter 9).

1.4 Contributions of this thesis

The development of mechanisms for furnishing social aware robots with learning competences is a fundamental step for increasing their flexibility in interacting with humans and their adaptability to unstructured environments. This thesis intends to be a contribution in that direction by addressing some of the main challenges that such endeavour implies. The questions it addresses rise in the context of the DNF-based cognitive architecture for joint action used in the robot ARoS. Concretely, the thesis is focused on two main topics identified as fundamental in the purpose of integrating learning abilities in the architecture: i) The learning of sequential task plan representations that can be used as shared task knowledge for human-robot cooperation; ii) The formation of neuronal representations for generalization of novel, not pre-specified, knowledge.

The main results of the research on these topics can be summarized as follows.

- Development of a biologically-based dynamic neural field model (DNF) for learning sequential tasks. The model follows a LfD paradigm, allowing the robot to

learn from demonstrations of human teachers in a naturalistic way, similar to how humans learn from each other.

- Extension of the model through inclusion of a fast learning mechanism for encoding sequences in a single demonstration. The sequence memory is used in an internal rehearsal process to avoid the need for repeated demonstrations from the human tutor.

- Extension of the model to incorporate the hierarchical nature of the sequential tasks by including a mechanism for the development of sub-task representations, formed as clusters of sub-goals.

- Implementation of the model for task learning in ARoS and testing of the model's performance in a real interaction scenario, in which the robot learns from demonstration the sequential structure of a construction task. The rehearsal mechanism and the development of hierarchical representations are also demonstrated.

- Development of a neuro-computational mechanism based on self-organizing principles for evolving new neuronal representations within a dynamic neural field. As a particular example, the mechanism is used to cluster cinematic data of arm movements in object manipulation tasks, forming representations of motor primitives.

- Integration of the self-organizing mechanism with a simplified version of the sequence learning model. The new model is able to both evolve and chain neuronal representations of motor primitives. The impact of the chaining mechanism for the prediction of observed action sequences is demonstrated.

1.5 Thesis outline

The reminder of this thesis is organized as follows:

Part II, presents the materials and methods employed in the thesis. It is formed by the following three chapters:

- In Chapter 2 the robotic platform ARoS is presented.
- The framework of dynamical neural fields is presented in Chapter 3, together with a description of important cognitive functionalities that the field dynamics supports.
- Chapter 4 introduces artificial neural networks and discusses synaptic learning rules that inspired the learning algorithm for the DNF model.

Part III, presents the state-of-the-art of sequence encoding that is relevant for the thesis work. It is formed by two chapters:

- In Chapter 5 work on task learning at the plan level, focused on the field of robotics is reviewed. The primary challenges are identified and several approaches currently used to tackle them are categorized and described.
- Chapter 6 contains an overview about behavioural and computational models of sequence learning, covering the main classes of models and their distinguishing features.

Part IV, describes the model for learning sequential tasks and reports results on real-world implementations. It consists of two chapters:

- Chapter 7 describes the model for learning and recall of sequential tasks/actions. It also addresses the neuro-physiological inspiration and provides the mathematical details of the model.
- Chapter 8 focuses on the implementation of the model for learning sequential task representations, on the robotic platform ARoS. Technical information on the robot is given and the task used for the experimental trials is described. Results of the implementation are also presented.

Part V presents the model for emerging representations of motor primitives from sensory data and their sequencing. It consists of the following two chapters:

- Chapter 9 presents the self-organizing mechanism, for developing new neuronal representations. The chapter also details how the self-organizing and the sequencing mechanisms are combined in a model able to encode motor primitives and to anticipate the final outcome of a motor sequence.
- Chapter 10 details simulation results from the implementation of the model for evolving and sequencing new neuronal representations of motor primitives. It is shown how the new representations are developed and how it is possible to anticipate the outcome of motor actions, by encoding the serial order by which the representations are activated.

Part VI, contains Chapter 11, where final conclusions and future work are presented.

This page has been intentionally left blank.

Part II

Materials and methods

Chapter 2

Anthropomorphic Robotic System (ARoS)

This chapter describes the robot ARoS, the platform used to test the model for learning sequential tasks. It begins with a general overview, followed by a more detailed description of its components.

2.1 General description

The robotic platform ARoS was developed at the University of Minho with the goal to test cognitive models of human joint action. It was designed and built under the scope of European project JAST[1] and a project funded by the Portuguese Foundation for Science and Technology (FCT)[2].

ARoS (Figure 2.1) consists of a static torso with an attached 7 Degrees of Freedom (DoF) Arm (*Lightweight Amtec Arm*) with a three fingered hand (*BarrettHand BH8 Series*). A Pan-Tilt unit (*PTU-D46-17* from *Directed Perception*) serving as neck

[1] JAST "Joint action Science and Technology", project financed by the European Commission, (ref. IST-2-003747-IP)

[2] FCT ref. CONC-REEQ/17/2001

supports a stereo vision system (*STH-DCSG-VAR-C* from *Videre Design*). The robot is also equipped with systems for speech recognition and speech synthesis, based on *Microsoft Speech 6.2 API*.



Figure 2.1: Anthropomorphic Robotic System (ARoS).

The integrated control of the several components that form ARoS is based on the YARP platform (Yet Another Robotic Platform) (Metta et al., 2006). Each of the physical actuators (arm, hand and pan-tilt) as well as the vision system, the speech synthesis and the speech recognition is controlled by an individual application, called wrapper. These control components communicate through a distributed network protocol based on the YARP specification. All wrappers receive commands from a central control module which works as an interface between a high-level control application deemed High Level Cognition (HLC) and the wrappers. In the experiments described in this thesis, the HLC corresponds to an implementation in Matlab of the model for learning sequential tasks.

The control of the individual components of ARoS is briefly described next.

2.2 Vision system

ARoS is equipped with a stereo vision system (*STH-DCSG-VAR-C* from *Videre Design*) which allows it to identify objects in sight and continuously calculate their spatial location. Object are recognized through computer processing of the images given by the cameras (Silva, 2008). Concretely, a colour segmentation is applied to the images for detection of coloured regions which constitute object candidates. Objects are then identified through individual colour, combination of colours and region size. For each recognized object, the spatial location relative to the robot's body is retrieved by querying the stereo system about the corresponding image region.

The stereo vision system is mounted on a Pan-Tilt unit (PTU D46 from *Directed Perception*) which allows the cameras to move around the vertical and horizontal (parallel to shoulders) axis of the robot. This is useful for increasing the robot's range of sight. It can also be used as a social cue during human-robot interaction, since the robot can look on a particular object, communicating its current state of attention to the user.

2.3 Arm and hand

ARoS ability to manipulate objects is conferred by its arm and hand. Together these components allow the robot to perform manipulation tasks like reaching and grasping objects, employing different types of hand grip postures, moving objects to different places, handing over objects to a partner or inserting them in a peg. The trajectories for motion control are defined by the motion planning algorithm proposed by Costa e Silva et al. (2014). The algorithm was developed to produce human-like movements, facilitating motion understanding by a human observer. It is based on observed regularities in human upper-limb movement studies and on the posture-based motion planning model of Rosenbaum et al. (2001).

2.4 Speech recognition/synthesis

The modules for speech recognition and speech synthesis furnish ARoS with basic verbal skills useful for interaction with humans. The speech synthesis system allows reproduction of words or sentences provided as text strings. Likewise, the speech recognition module is able to recognize predefined strings like keywords or entire sentences. Beyond relying on the *Microsoft speech API 6.2* the software uses the *Lehevenstein* algorithm (Levenshtein, 1966) for improving the recognition capabilities. Input is supplied by a head-set microphone, worn by the human.

Chapter 3

Dynamic Neural Fields

Dynamic Neural Fields theory is one of the fundamental tools employed in this thesis. This chapter describes the basic concepts underlying the theoretical framework of DNFs. It begins by addressing the need for representations in robot control architectures by discussing limitations of the purely behaviour-based control. It then describes the basis of the dynamic systems approach to behaviour-based robotics and how dynamic field theory can build a bridge to cognitive systems theory. The chapter continues with a description of the *Amari* dynamic neural field formulation which is extensively used throughout the thesis. It also describes some of the cognitive abilities that can be modelled with DNFs.

3.1 Behaviour and representations

During the nineties, there was an intense debate among robotic researchers about the need of explicit knowledge representations in robotic systems [1]. On one side, there were

[1] This discussion is not limited to robotics. Actually, it spawned from the more general field of Artificial Intelligence and exists in parallel with similar questions, in psychology, regarding human behaviour (see Harnad (1990) for discussion). In this case the discussion is about the need of contemplating mental states when studying behaviour (cognitivism) or simply perform the analysis in terms of associations

those who defended the needless of representations and that robots could be controlled based on a purely behaviouristic approach (Brooks, 1991). For a mobile robot, for example, this means that information given by the sensory system is directly translated on actuation of the locomotion system. There is no need for the robot to have an internal representation of the environment, since a simple set of rules, mapping sensory input to output actuation, controls the robot behaviour. This approach gathered a large support group because it allows to endow robots with complex behaviours without the need of strong computational power (see Bicho and Schöner, 1997; Bicho et al., 2000, for examples). On the other side, there are opponents arguing that the behavioural approach limits robots to be purely reactive machines, in the sense that they are unable to predict future events and are merely reacting to them (Chatila, 1995).

A large body of posterior work seems to favour the latter hypothesis (Matarić, 2002; Steels, 2003; Schöner, 2007; Asada et al., 2009; Bicho et al., 2010) as an increasing demand for more complex behaviours evinced several processes requiring representations. Engels and Schöner (1995) elicited at least two of these processes:

- Memory: Representations are necessary if some sort of sensory memory is to be retained, specifically when sensory information must continue to influence behaviour even after its source stops feeding the sensory system.

- Decision making: In situations with ambiguous sensory information a specific decision between behaviours is required. The trade-off between a desired flexibility in behaviour and stability in decision making has to be solved by the system.

Nevertheless, the behaviour-based approach to robotics remains attractive due to its simplicity and robustness. As such, Engels and Schöner (1995) proposed a theoretical strategy for creating control architectures that, on one side, take advantage of the concepts of memory and representation and, on the other, are grounded on the same principles that underlie the behaviour-based paradigm. The proposal of Engels and

between sensory stimulus and output actions (behaviourism).

Schöner was based on the use of artificial neuronal representations where the behaviour of each neuron is controlled as a dynamical system.

3.2 The dynamical systems approach to cognition

The use of dynamical systems in robotics was first proposed by Schöner et al. (1995), within the behaviour-based paradigm, for planning and control of autonomous vehicle motion. Their approach was the basis for a substantial body of work in the field of mobile robots (Bicho and Schöner, 1997; Bicho et al., 1998, 2000; Bicho, 2000; Monteiro and Bicho, 2002; Monteiro et al., 2004; Soares et al., 2007; Monteiro and Bicho, 2008; Sousa et al., 2012; Machado et al., 2013). The basic concepts of the approach are rather simple. First of all, the robot's behaviour is internally represented by a set of variables, each describing an individual component of behaviour. These variables are called *behaviour variables* and assume values within *behavioural dimensions* that comprise the complete space of values the behaviours can assume. The choice of these variables takes into account that the goals or objectives of a task must be expressed as particular instances of the *behavioural dimensions* they define. Consider the example given in Bicho (2000) where a mobile robot must navigate autonomously towards a target location and avoid obstacles. The important behavioural variable here is the heading direction $\phi(t)$. In order for the robot to reach the target, $\phi(t)$ must evolve towards the target direction ψ_{tar} and avoid the direction ψ_{obs} which corresponds to the direction of an obstacle. Both ψ_{tar} and ψ_{obs} are concrete locations within the behavioural dimension defined by ϕ. The first must be sought and the second avoided. This leads to a second important concept of the dynamical systems approach: a time-continuous dynamical system defines the evolution of the behavioural variables in time. Thus, in the mobile robot example, the evolution of $\phi(t)$ is described by a differential equation

$$\frac{d\phi(t)}{dt} = \Theta(\phi(t), parameters) \ . \tag{3.1}$$

The function $\Theta(.)$ controls the rate of change applied to the behavioural variable

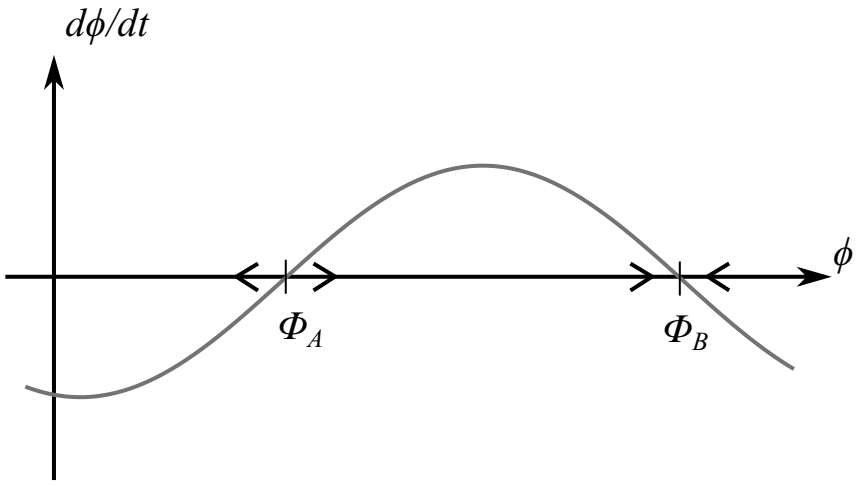


Figure 3.1: **Phase plot of a dynamical system.** The figure depicts the rate of change $d\phi/dt$ as function of ϕ. ϕ_A and ϕ_B are *fixed points*. The positive slope in ϕ_A makes this point a *repeller*, meaning that when the state variable is near this point, the dynamical system is drawn away from it. ϕ_B is an attractor fixed point. When the state variable is in the vicinity of this point the system evolves towards ϕ_B.

$\phi(t)$ as a function of its current value and possible additional parameters.

A third important concept is the following: sensory information affects not the variables itself, but the dynamics controlling them. In Bicho (2000) this is implemented by the control of the location of *fixed points* of the dynamical system. These are points of the vector field where the rate of change is zero, meaning that the value of the variable will will not vary unless some external perturbation exists. Fixed points, also called *equilibrium solutions*, can be *attractors* or *repellers*. The phase plot depicted in Figure 3.1 shows the rate of change $d\phi/dt$ as a function of ϕ for both situations. ϕ_A is considered a repeller because when the dynamical system is in its vicinity, the state variable is driven away from the *fixed point*. This happens because the slope in the neighbourhood of ϕ_A is positive. An attractor (depicted as ϕ_B in the Figure 3.1) has the opposite effect. It is an asymptotically stable fixed point, that is, in its vicinity the dynamical system is drawn towards this value.

The dynamical systems approach to behaviour-based robotics allows the implementation of complex navigational behaviours on robots with limited computational capabilities (Bicho and Schöner, 1997; Bicho et al., 2000). However, it is limited in

its ability to address task demanding higher-level cognitive capacities. This is due to two fundamental properties of dynamical systems: Variables in dynamical systems (1) have a single state at each time, and (2) evolve continuously in time, meaning that, in order to go from one state to another, they have to go through all intermediate states. Consequently, there are processes which cannot be addressed by a dynamical system formalized by ordinary differential equations:

- System (3.1) is unable to represent the absence of knowledge about the dimension it codes;
- Each variable is unique, so the presence of multiple instances of the same dimension (for instance multiple directions of obstacles) cannot be easily dealt with. One could think of a solution assigning a new variable for each new source of information. However, that would require solving the problem of matching each source of sensory information to its corresponding variable, in order to maintain temporal continuity.

To address these issues within the framework of dynamical systems, Schöner et al. (1995) proposed the principle of neural representation (see also Engels and Schöner 1995). Specifically, they introduced the concept of field activation over a behavioural dimension.

As mentioned already, a behavioural dimension represents a continuous of possible values a certain behaviour can assume. Thus, a *field of activation* can be defined over this dimension. The strength of activation at each location of the field represents the degree to which that value of the variable is present. Therefore, a specific state of the variable is represented by a supra-threshold field activation centred at the position encoding the specific value. For the navigation task, Bicho (2000), proposed a control architecture that uses a field to represent the target in terms of its angular position (Figure 3.2a). A field can also represent the situation where no target is detected by the sensory system, through a sub-threshold homogeneous activation profile (Figure 3.2b). Multiple instances of a behavioural dimension are represented in the field by multiple

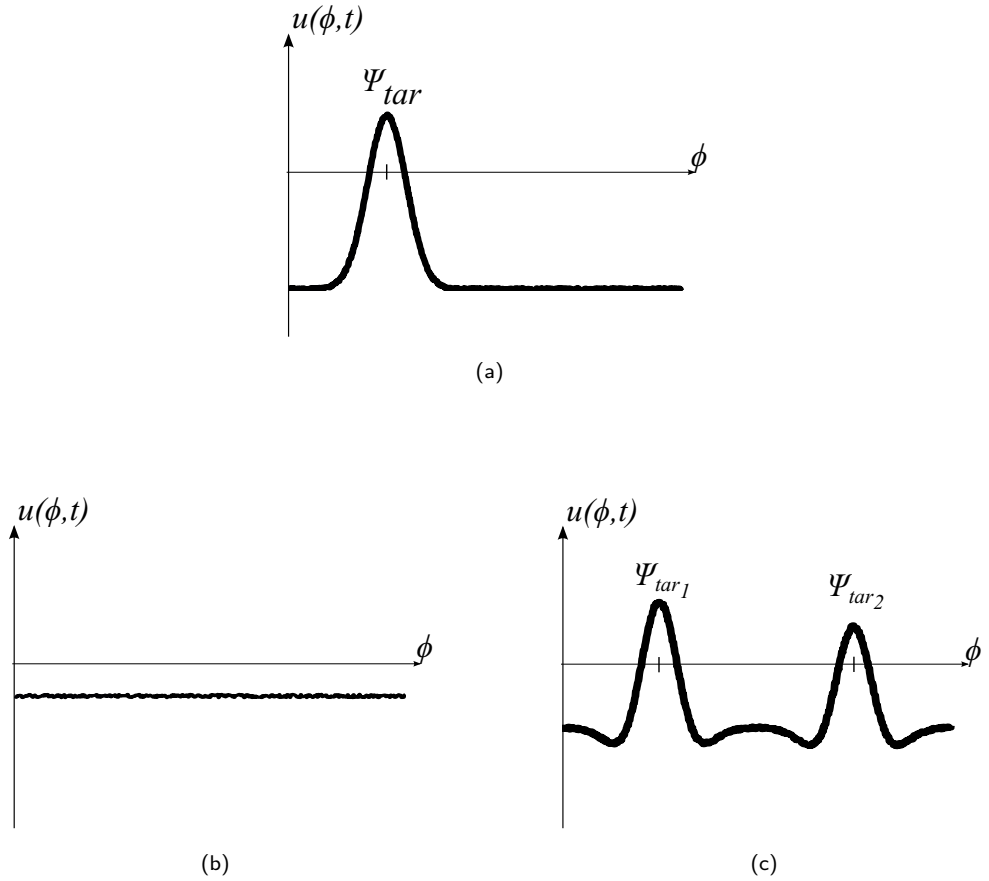


Figure 3.2: **Examples of field activations in a model for autonomous navigation.** (a) Field encoding the target direction. A single activation represents the presence of navigational target at direction Ψ_{tar}; (b) Field with no input represents the absence of sensory input; (c) A field is used to represent two targets. Concretely, two active populations represent the existence of two target in locations Ψ_{tar_1} and Ψ_{tar_2}.

activation peaks (Figure 3.2c). To maintain coherence with the dynamic approach, the activation on each field site evolves continuously in time governed by a dynamical system. Intended activation patterns are defined as attractor solutions resulting from external input and excitatory and inhibitory interactions within the field.

It is important to highlight that the behavioural dimension can encode not only scalar variables (like the angular position of a target on the plane) but also more abstract concepts (Bicho et al., 2011a). Fields may encode representations in higher-dimensional spaces. For instance, in Faubel and Schöner (2008) dynamic fields are used to represent object features like shape. In the cognitive architecture for joint action addressed in this

thesis (Bicho et al., 2010, 2011a), field activations are used to encode entire goal-directed motor acts like "reach" and "grasp" for which a natural metric is not known.

3.3 The Amari neural field model

One particular integro-differential equation has been extensively employed to govern the fields dynamics. It was first proposed by Amari (1977) (see also Wilson and Cowan 1973) as a mathematical model for the dynamics of pattern formation and interaction in the cortical surface. It was later used by Schöner et al. (1995) as a solution for controlling the fields dynamics, because it presents a set of properties that are specially adequate for modelling cognitive processes, like memory, forgetting, detection and decision making, to name a few (see Bicho et al. 2000). Also and contrarily to other field models, its dynamics can be analytically studied, under certain assumptions on model parameters.

The following equation governs the dynamical behaviour of a one-dimensional field:

$$\tau \frac{\partial u\,(x,t)}{\partial t} = -u(x,t) + \int w(x-y) f_0\left[u(y,t)\right] dy + h + s(x,t)\ . \tag{3.2}$$

It defines the activation u at field location x at time t. To understand the equation, one can think of the behavioural dimension x as a layer of neurons with an infinitely dense distribution. u represents the neural activation of the neuron at location x at time t.

For a more comprehensive analysis one can first address the linear part of the equation:

$$\tau \frac{\partial u\,(x,t)}{\partial t} = -u(x,t) + h + s(x,t)\ . \tag{3.3}$$

The term $s(x,t)$ represents the input to neuron x at time t. It is excitatory ($s(x,t) > 0$) when information about the encoding variable is available and zero or inhibitory ($s(x,t) <= 0$) otherwise. The parameter τ defines the relaxation rate of the

field and h its resting level. The dynamic equation 3.3, implements a simple low pass filter over the input, since the field relaxes towards the value $S(x,t)+h$. In other words, it merely reproduces the input pattern $s(x,t)$ with a time delay controlled by τ. The properties that allow the use of the Amari's equation as a tool for modelling cognitive processes result from the integral term:

$$\int w(x-y) f_0\left[u(y,t)\right] dx' \ . \tag{3.4}$$

This term represents the interactions between neurons of the same layer. In *Amari*'s formula, neurons within a layer can either inhibit or excite other neurons, depending on their distance. Typically, a neuron excites its direct neighbours and inhibits those that are further away (Figure 3.3).

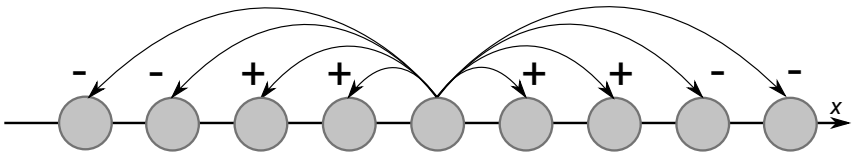


Figure 3.3: **Interaction between neurons.** In the *Amari*'s original formulation, a neuron excites other neurons that are closer to it and inhibits the ones that are further way.

The output function $f_0(u)$ ensures that only sufficiently active neurons can contribute to the interaction. Amari formulated it as a *Heaviside* function with threshold 0 which can be seen as a limit case of a more general function of sigmoidal shape. The *Heaviside* step is also applied in the present work:

$$f_0(u) = \begin{cases} 1, & \text{if } u \geq 0 \\ 0, & \text{if } u < 0 \end{cases} \ . \tag{3.5}$$

The activity of each neuron x, is influenced by all sufficiently active neurons of the field, $f_0[u(x,t)] > 0$. The strength of interaction is given by the interaction kernel $w(x-x')$ as a function of the distance between neurons $(x-x')$. Different formulations

of the kernel can be used, depending on the desired type of interaction. Figure 3.4 shows three possible kernel shapes.

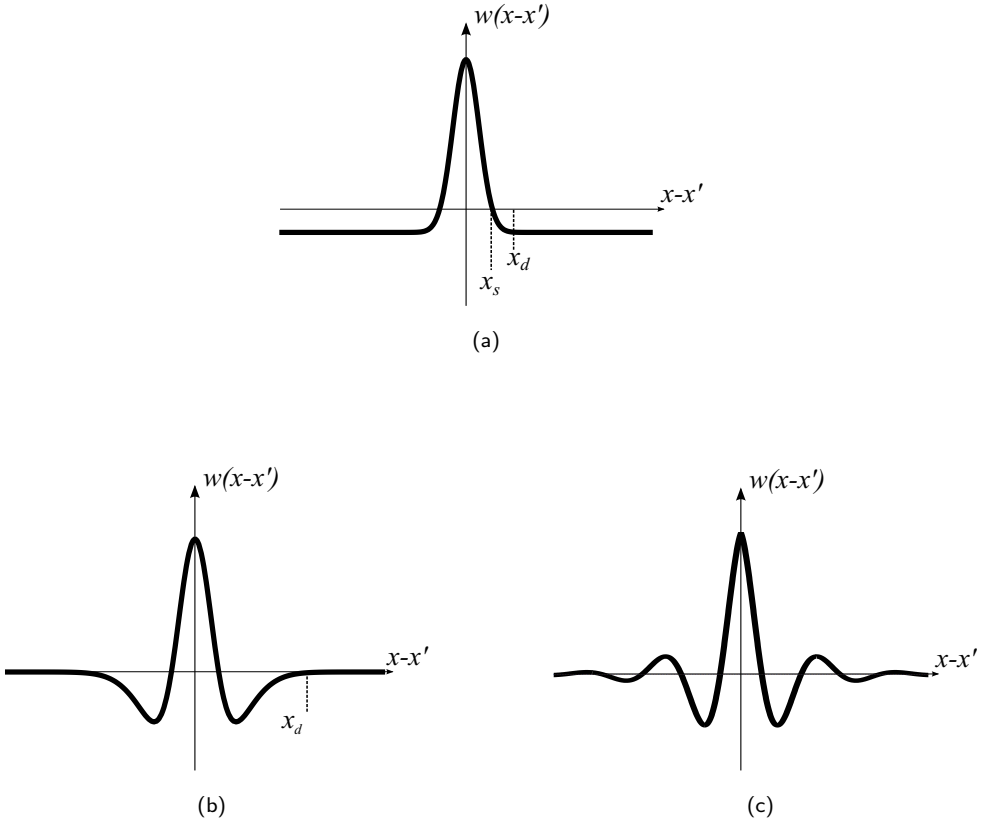


Figure 3.4: Interaction kernels. (a) Gaussian Kernel (b) Mexican-Hat Kernel (c) Oscillatory Kernel

The Gaussian kernel in Figure 3.4a is defined by the equation:

$$w(x-y) = Ae^{\frac{-(x-y)^2}{2\sigma^2}} - w_{inhib} \ , \tag{3.6}$$

where $A > 0$ controls the amplitude of the Gaussian, $\sigma > 0$ defines the standard deviation and w_{inhib} is a positive constant. When the interaction is characterize by (3.6), a neuron will excite another neuron if the distance between them is less than a value x_s (see Figure 3.4a) and inhibit it otherwise. For distances greater than x_d the inhibition strength is approximately constant. A kernel of lateral inhibitory type forces the competition between sufficiently distant neuronal populations. For sufficiently strong interaction, the field keeps a single, localized activity peak at each time in response to

external input, recovering the single state property of the behavioural variables.

The *mexican-hat* kernel in Figure 3.4b is defined by the equation

$$w(x-y) = Ae^{\frac{-(x-y)^2}{2\sigma_A}^2} - Be^{\frac{-(x-y)^2}{2\sigma_B^2}}, \tag{3.7}$$

which represents the difference of two Gaussians with $A > B$ and $\sigma_A < \sigma_B$. The *mexican-hat* kernel allows the coexistence of multiple activity peaks if they are localized at a distance greater than x_d, since the lateral interactions are zero at those distances (Figure 3.4b).

The third kernel (Ferreira et al., 2011) shown in Figure 3.4c, is defined by the equation

$$w(x-x') = Ae^{-k|x-y|}\left(k\sin|\alpha x - y| + \cos(\alpha x - y)\right), \tag{3.8}$$

The interaction dynamics introduced by the field equation with this kernel guarantees the existence and stability of multiple activity bumps (Laing et al., 2002). The oscillatory shape of the kernel with several zero crossings implements regions of (smaller) excitatory interaction also at larger distances. The parameter $k > 0$ controls the rate at which the oscillations in w decay with distance and the parameters $A > 0$ and $0 < \alpha \leq 1$ control the amplitude and the zero crossings, respectively.

Next are addressed three important cognitive processes that can be implemented using the Amari's field dynamics and their dependence on the choice of field parameters is discussed. Although other cognitive functionalities can be implemented as well (Erlhagen and Bicho, 2006), here the focus is in the examples that are more relevant for the presented work.

3.3.1 Detection

It was already mentioned that dynamical neural fields can code the absence of information. This happens when all neurons of the field have activity below the threshold 0. This does not necessarily means that no input exists. The excitatory input $S(x,t)$ might be simply not enough to drive the activation above the threshold for starting the interaction. The field activation pattern evolves to acquire the shape of $h - s(x,t)$ but remains negative for all x (Figure 3.5a). In order for an input detection to occur, a localized input must be sufficiently strong and remain present for the amount of time necessary for $u(x,t)$ to evolve until $f_0[u(x,t)] > 0$, to trigger the lateral interaction. The resulting peak of activation will be the sum of both the input and the recurrent interactions (3.5b).

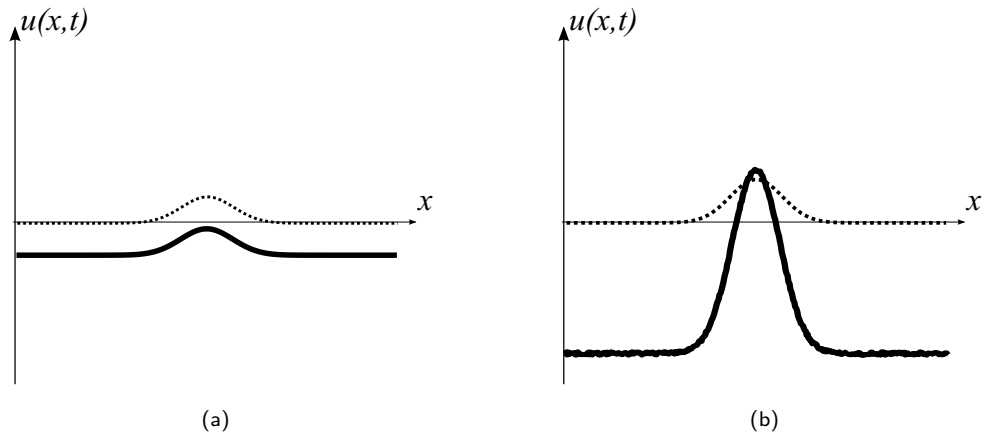


Figure 3.5: **Example of input detection and no detection.** In (a), the input is not strong enough to form a supra-threshold activation pattern and the field converges to $h + s(x,t)$.In (b), the input $s(x,t)$ (dashed line) is strong enough to drive field activity $u(x,t)$ (solid line) above the zero threshold, forming a self-sustained activation peak.

3.3.2 Memory and forgetting

One of the key features of dynamical neural fields is the ability of the recurrent interactions to maintain a peak of activation even if the input that initially created it is

removed. This self-sustained activity is centred around the same input location. For a field representing a behavioural dimension like, for instance, stimulus direction, the self-sustained activity constitutes a memory of the input, since the existing peak is only marginally stable, meaning that external perturbations may alter the position of the peak. This property can be exploited for tracking of a moving stimulus (Bicho et al., 2000).

There are, however, situations where it is necessary to suppress a self-sustained peak of activation if it corresponds to the memory of an event no longer relevant. The memory/forgetting property of a field can be controlled by the globally inhibitory input h. A neuron will be active without input, if the total balance of excitation and inhibition is sufficient to keep it above the threshold level 0. As a result, there is a threshold for the value of h ($h < -W_m = -\max_x\{\int_0^x w(\Delta x)dx\}$) below which the field is no longer able to maintain a self-sustained peak. By controlling the value of h it is possible to make the field either maintain a peak or to suppress it (Figure 3.6).

3.3.3 Decision making

A dynamical neural field can also implement the highly important cognitive ability of decision making, if the lateral inhibition is sufficiently strong. For instance, let us consider the situation in which an input pattern contains two peaks of activation. If these peaks are sufficiently distant, the interaction between them will be purely inhibitory. Consequently, there will be a competition between the two peaks and one of them will suppress the other. The choice of the "winning" peak depends on two factors. One is the amplitude of the input peaks. If one of the inputs is stronger than the other, the corresponding field population will have an advantage (Figure 3.7a). The other factor is the initial state of the field which may reflect, for instance, decaying activity from previous stimulations. If one of the input peaks is located on a field site with a pre-activation (still negative, but less than the other field locations) it will have a competitive advantage over the other peak and may reach positive activation even with a smaller input. (Figure 3.7b).

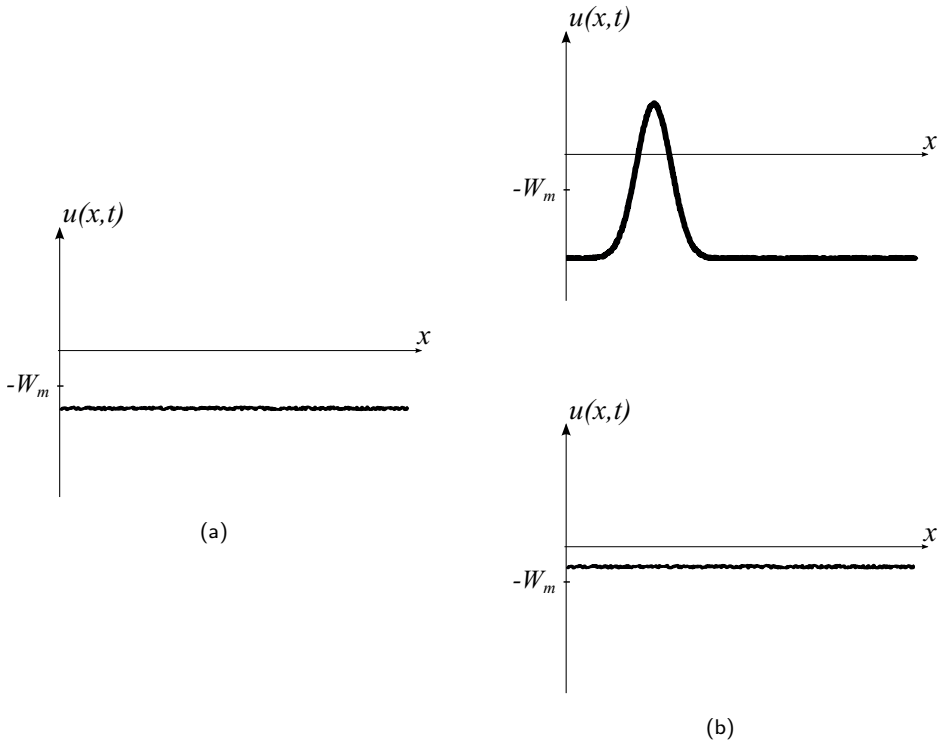


Figure 3.6: **Memory and forgetting in a DNF.** In (a), with $h < -W_m$ the field in not able to maintain a self-sustained peak in the absence of input. Only one stable state exists characterized by $u(x,t) = h$. In (b) $h > -W_m$ allow the existence of two possible stable states. one where all the field is equal to h and other where an activation is kept in the location of a previously detected input.

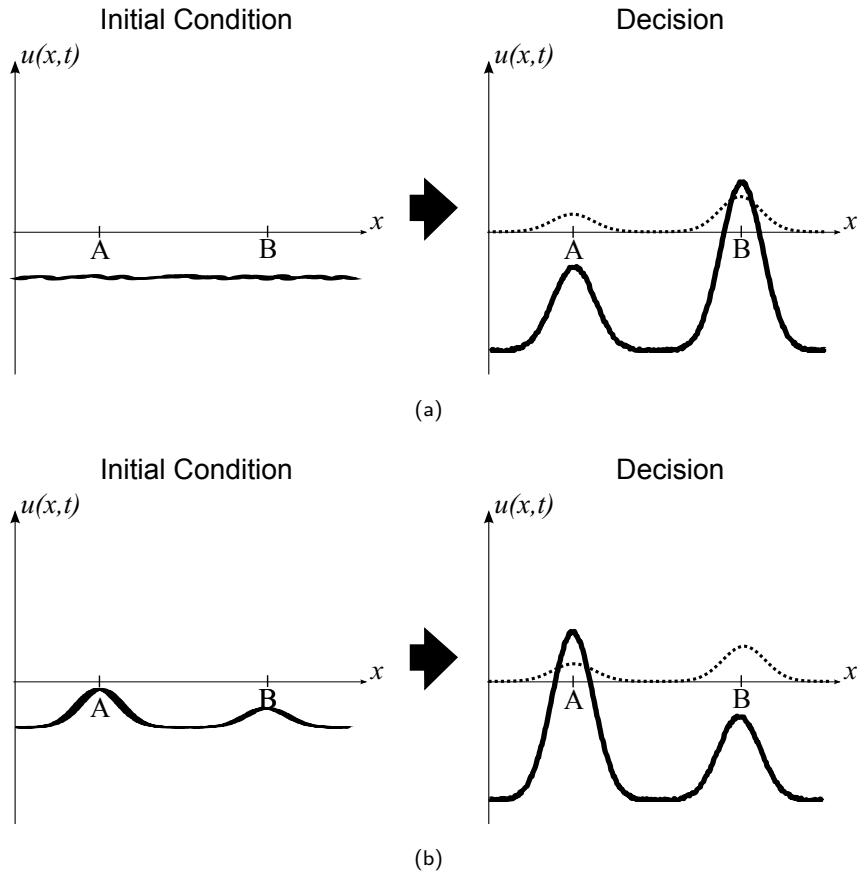


Figure 3.7: **Example of field decisions**. In (a) there was no preshape in the field's initial condition. Thus, a decision was made favouring the population with highest input. In (b) there was an initial preshape. When the input was set, the activation appeared in peak location A, although the input was stronger in B.

Chapter 4

Artificial Neural Networks

The previous chapter demonstrated how Dynamical Neural Fields can be used to model cognitive abilities like memory, forgetting or decision making. In this chapter it is shown how different fields can be interconnected providing the path for modelling another fundamental cognitive ability: Learning. For that, we rely on the framework of Artificial Neural Networks (ANN) that can be integrated rather effortlessly with the DNF theory (in fact, a Dynamic Neural Field can also be regarded as a neural network). The chapter begins with a motivation for the use of neural networks. It continues with a presentation of basic concepts and then addresses some learning principles that are used in this thesis. The chapter ends with a description of how the neural networks concepts can be integrated with the Dynamical Neural Field Theory.

4.1 Why neural networks

The study of neural networks has come a long way since McCulloch and Pitts (1943) first showed how ANN can be used to compute any arithmetic or logical function. Today, ANN is a well established research field with applications in domains as diverse as the aerospace industry, banking or medical applications, to name a few. The idea of studying neural networks and replicate its functioning in artificial models has always

attracted researchers for rather obvious reasons. For a computer engineer, having models of neural mechanisms represents the opportunity to develop processing tools capable of doing what the brain does with extreme efficiency: compute huge amounts of sensory data, extracting meaningful information from it, learn to make accurate predictions based on that information, and as a response produce fine tuned motor control. Computational models of neural mechanisms also raised interest among neuroscientists, since they have the potential to facilitate the understanding of the brain functioning and eventually help in the development of more efficient treatments for mental illnesses or in improvements in the teaching methods used in schools (Haykin, 2009).

The power of neural networks as a computational tool derives essentially from two factors: The parallel distributed processing they perform and their ability to learn. The first factor is intrinsic to ANN. They consist simply of processing units called neurons, each one individually doing only basic calculations on relative small portions of data. However, these simple computations are massively parallelized, strongly reducing the time necessary for computing large data sets. These simple computations can, in the right network configuration, implement highly complex non-linear functions. Also, because the computation is distributed, ANN are inherently fault tolerant, since damage on part of the network does not necessarily translate on complete degradation of the network response. The same can be said if the supplied input data is somewhat incomplete or degraded. In most cases, the network can still produce valid output. The second factor that makes them a powerful tool is their learning ability. The connections between neurons can be adapted as a function of input data, changing the overall response of the network. For a human brain this means that it can learn from the environment and easily adapt to changes. For an artificial neural network it means that it is possible to reproduce an input-output mapping function by learning from examples without the need of explicit knowledge about the function. This fundamental aspect makes ANN extremely useful in fields like pattern recognition and other fields that deal with highly non-linear functions, that are often very difficult or impossible to formalize.

Given that neural networks have this intrinsic ability to learn from input data,

their use is also appealing in the field of service robotics. Robots present in human environments must deal with highly variable and noisy sensory information. Neural networks can help robots to cope with this problem by serving as the substrate for control architectures, allowing them to learn to interpret these data and the proper way to deal with them. This is specially important when interacting with human beings (Breazeal et al., 2005). Being able to interpret human actions and behaviours and anticipating their needs, is one of the most valuable assets a social robot can have (Bicho et al., 2010, 2011a). With neural networks, robots can learn to make this interpretation and to anticipate actions outcomes. They can also be taught how to properly respond to a prediction, producing adequate complementary behaviours.

4.2 Basic functioning

In the previous chapter, it was referred, that a DNF can be seen as the formalization of a neuronal layer with an infinitely dense distribution of neurons. The statement is reflex of the close relation between the two frameworks, DNFs and ANNs. To understand this proximity and explore the basic principles of ANNs, one can begin by addressing the basic unit of any neural network, the neuron, using the same nomenclature employed in DNFs. To follow this idea, one must first state that a neuron has an activity level u that depends on an external input s and input from neighbouring neurons (like in the dynamic fields). Each source provides a contribution to the total input, that can be labelled q_i, where $i = 1, 2, ..., m$ identifies the source and m corresponds to the total number of input sources. Each of these inputs propagates to the neuron through a synaptic connection. This propagation is formalized by the product of q_i by a synaptic weight a_i that characterizes the synapse. The sum of all contributions forms the input s. Mathematically, this is translated in the equation

$$s = \sum_{i=1}^{m} q_i a_i \ . \tag{4.1}$$

Figure 4.1 depicts a diagram of a neuron like the one just described. Each input

source is represented by a node, connected to the neuron by the corresponding synaptic connection with weight a_i. There is an additional contribution labelled b which is typically used in neuronal networks to represent an external bias applied to the input. This bias can be either positive or negative and introduces an overall shift to s. In the *Amari* field equation (3.2) the b is replaced by the resting level $h < 0$, that applies a negative shift to the input s.

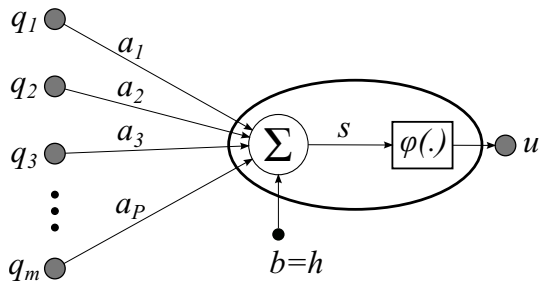


Figure 4.1: **Diagram of a single neuron.** The sum integrates the input to each input node q_i mediated by the synapses a_i and the bias b which is in this case, made equal to the resting level h.

The s signal resulting from the sum of all external inputs and the bias h is processed by the activation function $\varphi(.)$. This function limits the amplitude of the output of a neuron. The overall response of a single neuron can be mathematically translated in the equation,

$$u = \varphi\left(\sum_{i=1}^{m} q_i a_i + h\right) \ . \tag{4.2}$$

The activation function $\varphi(.)$ can have several formulations, the most common ones being the *Heaviside* and sigmoid functions. The *Heaviside* function is used in the DNF formulation presented in the previous chapter (3.5). It limits the activity levels of a neuron to values equal to zero or one. A neuron using the *Heaviside* function works as a binary classifier, dividing the input space by an *hyperplane*. It was first proposed by McCulloch and Pitts (1943) based on the "all-or-none" property of neuronal activity, being historically known as the *McCulloch-Pitts model*. Figure 4.2a depicts a neuron with two input nodes. This classifier divides a bi-dimensional input space represented

by two sources q_1 and q_2 by a line. Each point represents a input pair $\{q_1, q_2\}$. The line defined by the equation $q_1a_1 + q_2a_2 = h$ divides the space into two regions (Figure 4.2b). Points above the line will produce an activation $u = 1$, while points bellow line will have no activation $u = 0$.

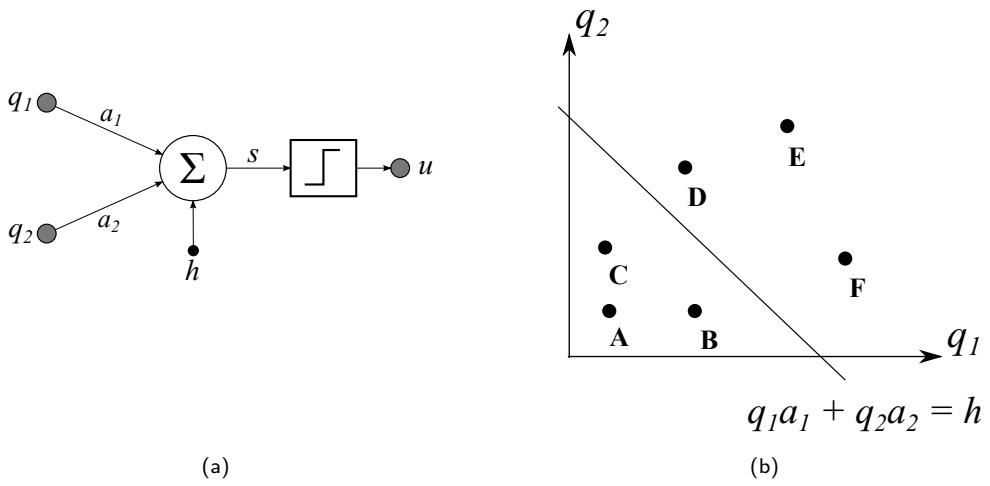


Figure 4.2: **Linear separation of input space.** (a) depicts a neuron with two input nodes, a bias h and an Heaviside as activation function. This neuron operates as binary classifier separating the input space into two regions. A graphical representation of the classification is shown in (b). Each point represents an input set $\{q_1, q_2\}$. The line defined by the $q_1a_1 + q_2a_2 = h$ divides the space into two regions. The neuron response will be $u = 1$ for points above the line and $u = 0$ for points below.

Sigmoid functions are also widely used, specially the logistic function

$$\varphi(s) = \frac{1}{1 + e^{-\sigma s}} \ , \tag{4.3}$$

where σ is the slope parameter. A sigmoid function like (4.3) is attractive as activation function mainly because it presents a smoother behaviour than the Heaviside with output values ranging continuously from 0 to 1. It is is also differentiable, facilitating mathematical analysis of the networks. At last, the *x-over-x-plus-1* can also be employed:

$$\varphi(s) = \frac{[s]^+}{1/\sigma + [s]^+} \ , \tag{4.4}$$

where $[s]^+$ is the positive part of s and σ is the slope parameter. This function has a similar behaviour as function (4.3) but has the particularity of trunking negative values.

4.3 Network organization

Having analysed the basic functioning of an individual neuron, one can now consider networks with several neurons organized into layers. These neurons share some or all input nodes. Beyond that, they can have a complete independent functioning or have interactions between them like in a dynamic field. Let us for now consider only neuronal layers with no interconnections. A network of this type is depicted in Figure 4.3. In this case, each neuron j receives an input s_j which is the sum of contributions from all the input nodes biased by h_j.

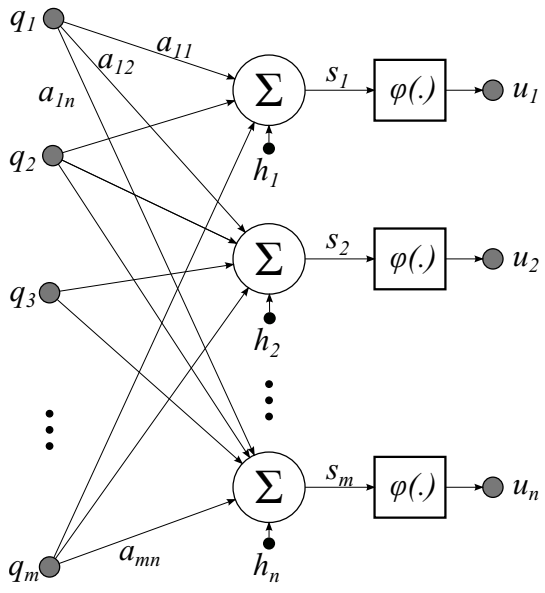


Figure 4.3: **Example of network with multiple output neurons.** In this particular example, each output neuron integrates input from all the input neurons.

The mathematical formalization of a single neuron 4.1 can be extended to address the complete set of neurons within a layer:

$$u_j = \varphi\left(s_j + h_j\right) \ , \tag{4.5}$$

where u_j is the activation of neuron j resulting from the input s_j, which in turn is given by

$$s_j = \sum_{i=1}^{m} q_i a_{i,j} \ , \tag{4.6}$$

where $a_{i,j}$ is the weight of the synapse linking the input node i to the pos-synaptic neuron j. In the neural networks literature, it is rather common to use the matrix form of the previous equations. Following that formalism, the weights $a_{i,j}$ are grouped in the matrix $\mathbf{A}$:

$$\mathbf{A} = \begin{bmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,n} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m,1} & a_{m,2} & \cdots & a_{m,n} \end{bmatrix} \ , \tag{4.7}$$

and 4.6 is replaced by

$$\mathbf{s} = \mathbf{A}\mathbf{q} \ , \tag{4.8}$$

where $\mathbf{q}$ is the vector formed by all input units q_i and $\mathbf{s}$ is the vector containing the output of each sum.

With respect to topological organization, networks can be classified according to two factors: the number of layers and to the direction of data projection.

In terms of the number of layers, networks are categorized as *single-layer* or *multi-layer* networks. In *single-layer* networks, the source or input nodes propagate directly to the output layer of neurons through a single set of synapses (Figure 4.4a). Each neuron functions as a linear classifier, firing in the presence of a specific feature of the input

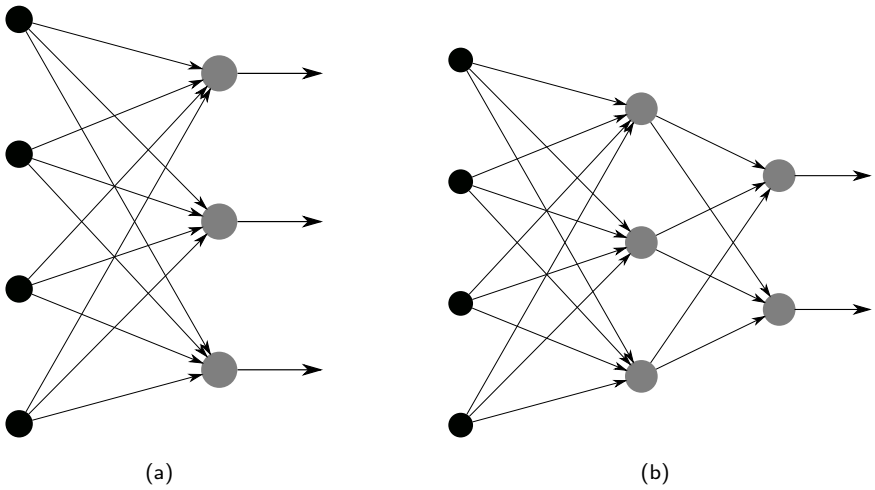


Figure 4.4: **Single layer and multi-layer networks.** The diagrams present examples of a (a) single layer network and (b) multi-layer network

space. *Multi-layer* networks have one or more layers of neurons between the input nodes and the output layer (Figure 4.4b). These intermediate neurons are normally known as *hidden neurons*. In a *multi-layer* network, the input nodes project onto the first *hidden layer*. The output of the first *hidden layer* is the input of the next layer which can be the output or another *hidden layer*. The use of *multi-layer* networks in many applications is due to a major limitation of single layer networks: they are unable to correctly classify inputs that are not linearly separable. This is the case for instance of the XOR problem, where the two classes of inputs are linearly dependent.

Regarding the direction of data projection, networks can be either *feedforward* or *recurrent*. The examples of Figure 4.4 are both feedforward, meaning that input projection is always in the same direction, with each layer providing input solely into the next one. In the case of recurrent networks there is at least one feedback loop, thus the output of one or more layers is also used as input of a previous layer (Figure 4.5).

Depending on the network topology the rules for network analysis and weight adaptation vary greatly. The networks used in this work can, in some perspective, be classified as *multi-layer* networks. However, the synaptic weight adaptations are applied between adjacent pairs of layers in a *single layer* approach. Hence, the learning rules

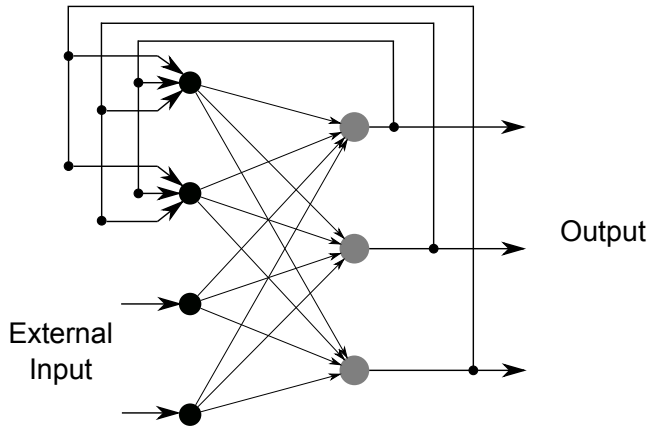


Figure 4.5: **Example of recurrent neural network.** The output network signals are feed back to the input nodes.

and considerations discussed in the next sections are mostly directed to this particular sub-class of networks (although they might also be valid to the other types).

4.4 Types of learning

Overall, what makes neural networks a distinctive computational framework is their ability to learn. There is a huge body of methods and approaches for adapting the weights of a network and achieving a proper response to an input or set of inputs. These methods are grouped into three major classes, each one complying with a particular learning paradigm.

1. **Supervised Learning**. Supervised Learning implies the existence of some sort of external agent that assumes a teaching role towards the network. This teaching agent possesses a complete knowledge of the environment where the network is framed and what should be the network response to each particular environment state. More concretely, the teacher knows, for each input, what the desired output is. Typically, the input is supplied to the network which will produce an output response. This actual output is compared with the desired output and an error signal is produced. The weights are then iteratively adapted in order to minimize the error. The *Delta Rule* (Widrow and Hoff, 1960) which we will address further

ahead is an example of a supervised learning rule.

2. **Unsupervised Learning**. This learning paradigm contrasts with the previous one since the learning is made without any external supervision from a teacher. The weights are adjusted to capture statistical regularities of the input data, forming internal representations of such regularities and therefore autonomously organizing the input data into classes. Usually this is achieved with learning rules that introduce competition between the output neurons of a network. This competition means that, when the weights of the synapses leading to a specific neuron are strengthened, the synaptic connections leading to other neurons are weakened. The consequent weight distribution results in a network behaviour characterized by having each output neuron or group of neurons representing a specific class of inputs. A classical example of such networks are the SOMs introduced by Kohonen (1990).

3. **Reinforcement Learning**. In Reinforcement Learning it is assumed that the network is working in closed-loop with the environment. Therefore, the output of the network affects the environment state which is then feed in the network. The learning control is carried out by a tutor that performs an evaluation on the environmental state providing a reinforcement signal to control the weight adaptation. Normally, rather than classifying the current environmental state as being influenced solely by immediate network output, the tutor considers it as a consequence of the several outputs verified during a certain amount of time. This happens because in several situations, earlier network outputs are more accountable for the current environmental state than the immediate ones. In fact one of the main challenges in the reinforcement learning paradigm is precisely the temporal credit assignment problem (Sutton and Barto, 1998) which deals with the challenge of determining the contribution to the current environment state of each past network state.

4.5 Hebbian learning

Hebbian Learning is one of the most famous learning paradigms applied in ANN. It is based on the postulate defining the dynamics for synaptic adaptation between two cells, proposed by Donald Hebb in his 1949's book, "The organization of behavior" (Hebb, 1949) . It states the following:

> *"When an axon of cell A is near enough to excite a cell B and repeatedly or persistently takes part in firing it, some growth process or metabolic change takes place in one or both cells such that A's efficiency, as one of the cells firing B, is increased."*

The postulate defines a neural substrate for the notion of associative learning. If two cells are active at the same time, then the connection between them is strengthened. If one consider each cell as being part of a neural representation of an event, then the events become associated in such a way that the occurrence of event A will contribute to the activation of the representation of event B. The principle of Hebbian learning can be mathematically formulated by the equation

$$\Delta a_{i,j} = \gamma(q_i, u_j) \ , \tag{4.9}$$

where $a_{i,j}$ is the synaptic connection linking the pre-synaptic neuron i to the post-synaptic neuron j and $\gamma(.,.)$ defines the weight variation as a function of the input value q_i and the activation u_j. The simplest formulation of the previous rule is given by the multiplication of the input and output signals, with a *rate* parameter η:

$$\Delta a_{i,j} = \eta q_i u_j \ . \tag{4.10}$$

The equation obeys the Hebbian principle since the weights can only grow if both the pre and post-synaptic neurons are active. However, its continuous application makes the weights grow indefinitely and any sort of representation is lost. Due to this and other

limitations the simplest Hebbian learning rule is rarely used. Nevertheless, its principle is behind a large number of a widely used weight adaptation rules, including the ones used in this work.

4.6 Delta rule

It was mentioned in the previous paragraph that with continuous application of the simple Hebbian learning rule (Equation 4.10), the weights will grow continuously. This is explained by the fact that the learning rule does not take into consideration the network current state. The weights grow regardless of their current value. A different approach is to make the growth depend, not on the network desired output, but on the difference between the desired and the actual output. The weights evolve in order to minimize the output error and eventually stabilize when the error is near zero. When using this method, setting the weights can be seen as an optimization problem where the goal is to minimize a cost function $\epsilon(a)$ of the weight parameters a. One of the most well known solutions to this problem is the Least-Mean Square (LMS) Algorithm proposed by Widrow and Hoff (1960) which makes use of the *Delta Rule* formalized by the equation

$$\Delta a_{i,j} = \eta q_i e_j \ , \tag{4.11}$$

where e_j is the difference between the desired output u_j^{tar} and the total amount of input. Thus, it is given by

$$e_j = u_j^{tar} - \sum_{i=1}^{m} q_i a_{i,j} \ . \tag{4.12}$$

The LMS algorithm has been widely used as an optimization and function approximation tool, specially in the fields of signal processing and communications applications, due to its simplicity and computational efficiency that comes at no expense of its performance.

4.7 The Self-Organizing Map

The Self-Organizing Map (SOM) is an important class of non-supervised algorithms that typically maps a set of input patterns of arbitrary dimension into an array or a bi-dimensional discrete surface of neurons. This mapping is deemed topological because the distance between neurons responding to different input patterns codes a measure of statistical similarity between those inputs. Similar patterns are mapped closer than dissimilar ones.

The development of the SOM has its roots on known neuro-physiological principles. Concretely, it is known that many classes of sensory inputs are mapped in the brain in a topologically order fashion (Haykin, 2009). This has inspired the development of a class of algorithms from which the most well known, the *Kohonen Map* (Kohonen, 1990), is described next.

Following the basic principles of the SOM, the *Kohonen Map* is formed by a set of neurons organized as an array or a matrix. Each neuron receives activation from input through a set of synaptic connections whose number is equal to the dimensionality of the input space. These connections are normally initialized with small random values. In every iteration of the map, one different pattern from the input space is presented to the network. The learning mechanism is based in three principles:

1. **Competition**. In every iteration of the SOM a winning neuron will be selected. This neuron is selected according to a competing function that performs some comparison between the input pattern and the weights of the synaptic connections associated with the neuron. The neuron with the highest value of such function is the winner.

2. **Cooperation**. The winning neuron defines the spatial location of the map considered more representative of the input pattern. The neurons within a region of the map centred on the winning neuron will have its weights adapted. The SOM map implements a **winner-takes-most** approach. This means the winning neuron

will undergo the strongest adaptation but the surrounding neurons will also have their weights updated to a lesser extent.

3. **Synaptic Adaptation**. The synaptic adaptation rule applied to the winning and surroundings neurons enhances their capability to encode the input pattern and thus, of winning the competition in future presentation of the same or similar inputs. Such is achieved through the adjustment of the weights in such a way that the competing function will produce higher outputs.

The *Kohonen Map* is now described in greater detail for the case of a feature map formed by a one dimensional array of neurons, since this will be the configuration used in the work reported in this thesis (although extension to two dimensions is straightforward).

The first step of the algorithm consists in the competition process. Let n identify a neuron from an output layer. Let $\mathbf{q}$ identify a vector from the input space at iteration k. To determine the winning neuron, the array of synaptic weights of every output neuron n must be compared with the input vector $\mathbf{q}$ by a measure of similarity. Two of such measures are the euclidean distance and the dot product.

If the euclidean distance is used, then the winning neuron is the one whose weights $\mathbf{a}_j$ have the minor difference to the input vector $\mathbf{q}$. The equation formalizing this measure is the following:

$$j^{win}(\mathbf{q}) = \underset{j}{\operatorname{argmin}} \|\mathbf{q} - \mathbf{a}_j\| \ , \ j = 1, 2, ..., l \ , \tag{4.13}$$

where l defines the total number of neurons. If the dot product is used, then the selection is achieved by computing the dot product between the input vector $\mathbf{q}$ and the vector of weights of each output neuron $\mathbf{a}_j$. The winner will be the one with the highest product, described as

$$j^{win}(\mathbf{q}) = \underset{j}{\operatorname{argmax}} \left\{\mathbf{q}^{\intercal}\mathbf{a}_j\right\} \ , \ j = 1, 2, ..., l \ . \tag{4.14}$$

Having selected the winning neuron the next step is to update the weights. This process reflects the cooperation principle, since not only the winning neuron will be updated, but also, to a lesser extent, its neighbours. Thus, the update of the neurons must depend on some kernel function, centred on the winning neuron. Also, the update must take into account the metric that is used in the similarity measure employed in the competition process and improve that similarity (Kohonen, 1990).

If the euclidean distance is used for the competition, then the following update rule is normally employed:

$$\Delta \mathbf{a}_j = \eta(k) w\big(j, j^{win}(\mathbf{q}), k\big)(\mathbf{q} - \mathbf{a}_j) \ , \ j = 1, 2, ..., l \ . \tag{4.15}$$

In (4.15) the term $\Delta \mathbf{a}_j$ represents the weight variation of the array of synaptic weights $\mathbf{a}_j$ of neuron j. The weight variation depends on a variation rate $\eta(k)$ and on a kernel function $w\big(j, j^{win}(\mathbf{q}), k\big)$ centred on the winning neuron $j^{win}(\mathbf{q})$. The last term $(\mathbf{q} - \mathbf{a}_j)$ is responsible for ensuring that the weights $\mathbf{a}_j$ are updated to approximate the value of the input $\mathbf{q}$.

If the dot product is used, then a simpler rule, more "Hebbian friendly" can be employed:

$$\Delta \mathbf{a}_j = w\big(j, j^{win}(\mathbf{q}), k\big)\mathbf{q} \ , \ j = 1, 2, ..., l \ . \tag{4.16}$$

This last rule however, requires an additional step of normalization to maintain the stability of the weights:

$$\mathbf{a}_j(k) = \frac{\mathbf{a}_j(k-1) + \Delta \mathbf{a}_j}{\|\mathbf{a}_j(k-1) + \Delta \mathbf{a}_j\|} \ , \ j = 1, 2, ..., l \ . \tag{4.17}$$

In both (4.15) and (4.16) the kernel function $w\big(j, j^{win}(\mathbf{q}), k\big)$ is defined as an exponential equation with a variable width. This width can decrease with time as the SOM becomes increasingly ordered and thus, there is no need to adapt the weights of distant neurons. The function is given by

$$w\big(j, j^{win}(\mathbf{q}), k\big) = \exp\left(-\frac{\|j - j^{win}(\mathbf{q})\|}{2\sigma^2(k)}\right) \ , \ j = 1, 2, ..., l \ , \tag{4.18}$$

where $\sigma(j)$ defines the variable kernel width formalized by

$$\sigma(k) = \sigma_0 \exp\left(-\frac{k}{\tau_\sigma}\right) \ , \tag{4.19}$$

where σ_0 defines the initial width and τ_σ is the decay rate.

One important result of using such a learning rule is related with the direction of convergence of each vector $\mathbf{a}_j$. It was shown by Oja (1982) that under adequate stability conditions the weights $\mathbf{a}_j$ of a neuron converge to the first principal component of the input set. In a SOM each $\mathbf{a}_j$ is only selected for update in a specific set of input presentations. Thus it will tend to converge to the first principal component of such reduced input set. This result plays a important role in Chapter 9.

4.8 Neural Networks and Dynamic Neural Fields

So far, the fundamentals of ANNs have been explained using the typical approach of analysing neurons as discrete units. It is possible however, to make a smooth transition between this formalism and the one commonly used in the DNF framework, where neuronal activity is modelled as field activation over a continuous spatial dimension. This is required if one intends to use some of the basic neuronal adaptation rules from typical neural network approaches to establish adaptable associations between DNFs. Concretely, the adaptation in formalism must occur in two aspects: the equations defining the propagation of activity from pre to post-synaptic neurons and the adaptation rules that govern the weights of the synaptic connections.

4.8.1 Activity propagation

Equation (4.6) defines the propagation of activity from a layer of input nodes to an array of neurons. This equation can be adapted to model activity propagation between two fields. To do so, the first step is to make an equivalence between the behavioural dimensions of DNFs and the discrete arrays of output neurons. This can be achieved by thinking of a behavioural dimension x^{out} as a set of discrete locations x_i, with a fixed spacing between them. If a neuron is present at every x_i^{out}, then, the following equivalences can be made:

$$u_j^{out} \equiv u\big(x_j^{out}\big) \ , \tag{4.20}$$

$$s_j^{out} \equiv s^{out}\big(x_j^{out}\big) \ , \tag{4.21}$$

where $u^{out}\big(x_j^{out}\big)$ and $s^{out}\big(x_j^{out}\big)$ are respectively the activation and the input of a neuron at location x_j^{out}. For the sake of coherence, let us consider a similar discrete space for the inputs q_i. To do so, the input nodes must also be considered discrete locations of a dimension, that can be for instance x^{in}. This new equivalence is formalized by

$$q_i \equiv q(x_i^{in}) \ . \tag{4.22}$$

Equation 4.6 can be rewritten to accommodate (4.21) and (4.22):

$$s\big(x_j^{out}\big) = \sum_{i=1}^{m} q\big(x_i^{in}\big) a\big(x_i^{in}, x_j^{out}\big) \ , \tag{4.23}$$

where $a\big(x_i^{in}, x_j^{out}\big)$ corresponds to the weight of the synaptic connection between input node located at x_i^{in} and the output neuron located at x_j^{out}. Since the main purpose of using neuronal networks is to couple DNFs, (4.23) needs to be adapted to reflect the propagation of activity between two fields. Let $u^{in}\big(x^{in}\big)$ and $u^{out}\big(x_j^{out}\big)$ represent the activations of two coupled fields. Equation 4.23 can be adapted to reflect propagation

from $u^{in}(x_i^{in})$ to $u^{out}(x_j^{out})$ by replacing $q(x_i^{in})$ with $u^{in}(x_i^{in})$. However, the models presented in the next chapter require that only neurons with activity above a certain threshold ($u > \lambda^{in} > 0$) are allowed to influence behaviour of the other fields. Thus a threshold function $f_{\lambda^{in}}(\cdot)$ must be applied to the input $u^{in}(x_i^{in})$ as gating mechanism. The input $s^{out}(x_j^{out})$ will then be equal to

$$s^{out}\left(x_j^{out}\right) = \sum_{i=1}^{m} f_{\lambda^{in}}\left[u^{in}\left(x_i^{in}\right)\right] a\left(x_i^{in}, x_j^{out}\right) . \tag{4.24}$$

The final step is to make the cross from the discrete domain to the continuous one. This is achieved by reducing the distance between the spatial locations of the neurons until an infinitely dense distribution is achieved. Then, the summation is replaced by the integral on the domain of the dimension x^{in}. The equation must also reflect the fact that the *Amari* field equation is time dependent, meaning that s^{out} and u^{in} must have time as a parameter. Thus, the function that gives the s^{out} as a result of the propagation from another field is given by

$$s^{out}(x^{out}, t) = \int_{x^{in}} f_{\lambda^{in}}\left[u^{in}(x^{in}, t)\right] \underset{in \to out}{a}\left(x^{in}, x^{out}, t\right)\left(x^{in}, x^{out}, t\right) dx^{in} . \tag{4.25}$$

The matrix of synaptic weights $a\left(x_i^{in}, x_j^{out}\right)$ is replaced in the previous equation by the continuous surface $\underset{in \to out}{a}\left(x^{in}, x^{out}, t\right)\left(x^{in}, x^{out}, t\right)$. The function is also time dependent, since the connections weights are adaptable, varying in time according with the learning rule.

4.8.2 Adaptation rules

The models reported in this work make use of the two learning rules explored in sections 4.6 and 4.7: The *Delta rule* and the SOM rule.

The Delta Rule

The *Delta Rule* is formalized by (4.11) and (4.12). Making the transition to the continuous domain, requires an approach similar to the one employed in the propagation formalization. First of all, the rule is intended to adapt the connections between two fields. Let x_i^{in} and x_j^{out} represent the respective locations of input neuron i and output neuron j. A new equivalence can be made for the term defining the weight variation:

$$\Delta a_{i,j} \equiv \Delta a\left(x_i^{in}, x_j^{out}\right) \ . \tag{4.26}$$

Secondly, since the input to the network is a field activation, the term q_i in (4.11) is replaced by $f_\lambda\big[u^{in}(x_i^{in})\big]$. At last, one must look to the error term $e(u_j)$ which is given as a function of the activation of the neuron j. u_j has already been replaced by its "field version" $u^{out}(x_j^{out})$, making u^{out} a function of the spatial location of neuron j. Thus, the error is now given by

$$e\big(u_j\big) \equiv e\big(x_j^{out}\big) \ . \tag{4.27}$$

The *Delta Rule* (4.11) can then be rewritten as

$$\Delta a\left(x_i^{in}, x_j^{out}\right) = \eta f_{\lambda^{in}}\big[u^{in}(x_i^{in})\big] e\big(x_j^{out}\big) \ . \tag{4.28}$$

The equation must now be adapted from discrete to continuous time version. To do so, the *difference* term $\Delta a\left(x_i^{in}, x_j^{out}\right)$ is replaced by the time derivative. Also, since the spacial dimensions x^{in} and x^{out} are continuous, the indexes can be discarded. The following equivalence can be made:

$$\Delta a\left(x_i^{in}, x_j^{out}\right) \equiv \tau_a \frac{\partial}{\partial t} \underset{in \to out}{a}\big(x^{in}, x^{out}, t\big) \ , \tag{4.29}$$

where τ_a is the time constant defining the variation rate in the weights. (4.28) can then be rewritten as

$$\tau_a \frac{\partial}{\partial t} \underset{in \to out}{a}\big(x^{in}, x^{out}, t\big) = f_{\lambda^{in}}\big[u^{in}(x^{in}, t)\big] e\big(x^{out}, t\big) \ . \tag{4.30}$$

The rate term η, is no longer required since its purpose is suited by the time constant τ_a.

The error term in the *Delta Rule* is the difference between the desired output and the total amount of input to a neuron. The last is given by (4.25). The first be can represented as $g_{\lambda^{out}}\big[u^{tar}\big(x^{out}, t\big)\big]$, where u^{tar} is the target activation. The function $g_{\lambda^{out}}[\cdot]$ is defined by (4.4) with a threshold value λ^{out}

$$g_{\lambda^{out}}\big[u\big] = \frac{[u - \lambda^{pr}]^+}{1 + [u - \lambda^{pr}]^+} \ . \tag{4.31}$$

The function ensures that the weight growth does not depend on the amplitude of u^{tar}. Like in (4.25), the threshold value λ^{out} is applied so that only activations above a specific value produce weight variation. Although the Heaviside step could also have been used, (4.31) produces a smoother bell-shaped target function. The weights will evolve to produce excitation with a similar smooth profile.

Having defined its components one can now write the error equation:

$$e\Big(x^{out}, t\Big) = g_{\lambda^{out}}\big(u^{tar}(x^{out}, t)\big) - \int\limits_{x^{in}} f_{\lambda^{in}}\big[\ u^{in}\big(x^{in}, t\big)\big]\ \underset{in \to out}{a}\big(x^{in}, x^{out}, t\big) dx^{in} \ . \tag{4.32}$$

The SOM Rule

The mechanisms of the SOMs can also be easily integrated with the DNF framework, because DNFs already implement two of the three basic principles of the SOM, namely the competition and the cooperation between output neurons.

To realize this integration, one can begin by defining a DNF equation $u^{som}\big(x^{som}, t\big)$ applied over a dimension x^{som} where the map is to be established. Let $q(x^{in})$ (form-

alized by (4.22)) be a continuous input pattern defined over the dimension x^{in}. Let $\underset{in\to som}{a}(x^{in}, x^{som}, t)$ be the surface function defining the weights of the synaptic connections, initialized with random values. The first step is to compare the input pattern with the weights of each output neuron. (4.25) already performs a calculation similar to the dot product in continuous space. Thus it can be adapted to compute the similarity measure between the input and the synaptic weights:

$$s^{som}(x^{som}, t) = \int_{x^{in}} q(x^{in}) \underset{in\to som}{a}(x^{in}, x^{som}, t) dx^{in} \ . \tag{4.33}$$

If the interaction kernel of the field is Gaussian-shaped with lateral inhibition, (see Section 3.3) it will force a competition and produce a localized activation. This activation will have a bell-shaped profile as result of the cooperation between neurons, even with the initial randomly valued weights. Thus it can replace the kernel function $w(n, n^{win})$ that was used to gate the weight adaptation. A Hebbian based rule can be employed to update the synaptic connections, that makes use of the field activation for controlling the range and degree of adaptation suffered by the winning neurons of the field:

$$\tau_a \frac{\partial}{\partial t} \underset{in\to som}{a}(x^{in}, x^{som}, t) = \left[u^{som}(x^{som}, t)\right]_0^+ q(x^{in}) \ , \tag{4.34}$$

where $[\cdot]_0^+$ is a ramp function with threshold zero.

Like in the previously described SOM algorithm, a normalization process must be performed for weight stabilization. Since the weight function is continuous, the applied norm function must be a continuous version of the euclidean norm,

$$\left\| \underset{in\to som}{a}(x^{in}, x^{som}, t) \right\|_{x^{in}} = \sqrt{\int_{x^{in}} \underset{in\to som}{a}(x^{in}, x^{som}, t)^2 dx^{in}} \ . \tag{4.35}$$

Finally the normalization function is defined by,

$$\underset{in\rightarrow som}{a}\left(x^{in}, x^{som}, t\right) = \frac{\underset{in\rightarrow som}{a}\left(x^{in}, x^{som}, t\right)}{\left\| \underset{in\rightarrow som}{a}\left(x^{in}, x^{som}, t\right) \right\|_{x^{in}}} \ . \tag{4.36}$$

4.9 Conclusion

This chapter and Chapter 3 form the mathematical backbone of the work described in this thesis. The next two chapters present respectively current state-of-the-art in the LfD paradigm of robotics and an overview on cognitive and computational models of sequential behaviour in humans. In Parts IV and V it is shown how the tools described here were used in the development of cognitive models for sequencing and generalizing neuronal representations and how these models can be applied to HRI.

Part III

State of the art

Chapter 5

Learning sequential tasks from demonstration in robotics

Within the LfD paradigm of robotics, the acquisition of serial order plays a fundamental role in one particular sub-domain: the learning of tasks at the plan level. This type of learning is focused on developing task representations formed as sequences of sub-tasks that must be completed through a specific order. This chapter presents a short review about methods normally used to encode the sequential relations between those sub-tasks. The chapter begins with a general overview about the problem of acquiring task/plan level knowledge from demonstration. It then focuses on the different approaches that are most commonly used to extract and encode the sequential path of the tasks.

5.1 General overview

As an approach to robot control LfD has been used to tackle a wide range of problems that go from learning arm trajectories through kinesthetic demonstrations (Billard et al., 2006) to learning routines for airplane control (van Lent and Laird, 2001) or the acquisition of complex behaviours like the ones required for household tasks (Zöllner et al., 2005). Given the wide range of applications, the scope of the LfD framework can

sometimes be difficult to define. One often made distinction that is particularly relevant for this thesis is between *skill learning* and *task learning* (Dillmann, 2004). While the first refers to precise imitation at the trajectory level, the later has the purpose of extracting knowledge at the plan level typically by representing a task as a sequence of sub-routines.

The challenge of developing a system for learning a task representation at the plan level can be broadly divided into three main problems. The first is concerned with the gathering of information from a demonstration. More concretely, it addresses the questions which method is used by the tutor for providing the learning examples and which sensory inputs and variables provide a relevant task description and therefore need to be collected by the learning system. The second subject deals with the segmentation of a task into meaningful units of action, recognizable across multiple demonstrations and reproducible by the robot. The learning system may learn just the goal state (e.g. object inside the box) or extract also a control strategy in order the achieve the goal (e.g. a motor primitive). The third subject regards the acquisition and encoding of the sequential order through which the several units of action can be encoded. This last point is the focus of the review presented here.

Although there are cases where the tasks require the execution through a single sequence, the majority of task learning applications deals with more flexible tasks that can be achieved through different sequential orders (Nicolescu and Matarić, 2003). Normally, this means that the demonstrator provides not one but several demonstrations of the same task. Learning algorithms capable of exploring this variability can increase adaptability by providing multiple action options, allowing the robot to choose the one that better suits current environmental constraints. Take the example of a navigation task. A robot furnished with multiple navigation paths can potentially deal with blocked paths by choosing alternative routes.

Learning a multi-path task plan requires a structure capable of encoding the relational rules defining the precedence relations between the units of action and a proper way to extract them from the multiple observations. Most models for task learning em-

ploys one or both of the following techniques: 1) precedence graphs and 2) hierarchical representations. Both alternatives are reviewed next.

5.2 Precedence graphs

The most commonly used structures to encode multiple action paths are precedence graphs. These structures are formed by nodes linked through edges (Figure 5.1). Nodes normally represent either actions that must be conducted or goals that must be achieved while the edges represent the precedence relations that must be respected. If the nodes represent goals the robot must engage on a behaviour that will lead to the achievement of that goal. This behaviour is the outcome of a control policy that is either pre-programmed (Ekvall and Kragic, 2006) or the result of a learning algorithm that extracts a policy from demonstrations (Aksoy et al., 2011). Goals can be represented in terms of exteroceptive states (Ekvall and Kragic, 2006) (states of variables external to the robot, e.g. objects positions) or proprioceptive states (states of the internal variables of the robot, e.g. the robots position in a navigation task) (Nicolescu and Matarić, 2003).

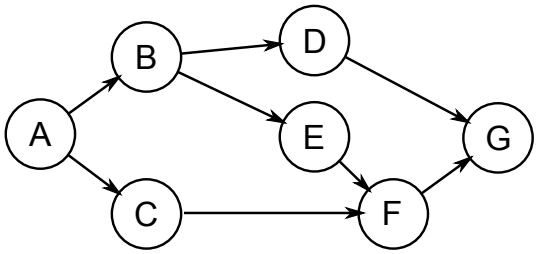


Figure 5.1: **Graph depicting a task with seven subtasks.**

Graph structures are a standard tool for dealing with applications involving artificial intelligence and optimization and many algorithms have been developed over the last years regarding the construction and optimization of these structures (West, 2001).

One particular case of such structure is the one where a single sequence is encoded. While this sacrifices flexibility it can improve the reliability of the extracted plan. Ogawara et al. (2002) proposed a method for learning a manipulation task that uses multiple demonstrations, not to learn a flexible task plan, but rather to extract a se-

quence of essential interactions between the hand and the object. The algorithm, based on Dynamic Programming (Bertsekas, 1995), assumes that the essential interactions are the ones present in all demonstrations, while non-essential result from errors and variability in the demonstrations. It also assumes that these essential interactions always follow the same order in all demonstrations.

5.2.1 Precedence analysis

Contrarily to Ogawara et al. (2002) most of the graphs structures in robot task learning assume the existence of multiple sequential paths of execution. The graphs are drawn through analysis of pre and post-conditions of each unit of action. Pre-conditions can be the memory of the execution of the previous action or the environmental state that results from it. Post-conditions result from execution of the unit of action.

Several applications have been proposed using precedence analysis. In Friedrich et al. (1996) a robot had to learn a sequence of manipulations performed on a set of objects. Pre-conditions were a combination of the outcome of the previous manipulation together with environmental context. Nicolescu and Matarić (2003) used a similar approach for a task where a robot had to navigate through a room while picking and placing objects. In the work of Bentivegna and colleagues (Bentivegna and Atkeson, 2003; Bentivegna et al., 2004) the precedences were encoded through a lookup table where current task state is used as a cue to retrieve the next primitive. Zöllner et al. (2005) proposed an algorithm to build a graph from multiple demonstrations. Beyond the precedence relations, their algorithm also allows the execution of actions in parallel in cases where the order is not relevant. In Ekvall and Kragic (2006) the learning algorithm assumed the existence of a precedence constraint for each two goal-states that are executed one after the other. If an order constraint is not observed in all demonstrations it is removed. The execution plan is built during execution through a planner algorithm that respects all the task constraints. The algorithm was inspired on the STRIPS (Stanford Research Institute Problem Solver) (Fikes and Nilsson, 1971). Veeraraghavan and Veloso (2008) use precedence analysis but make a distinction between

actions and objects to which they refer, allowing actions to be reused for different objects; Kulic et al. (2012) proposed a graph structure for encoding different sequences of motion primitives. Each primitive is encoded on a node while the edges represent the probability of transition between primitives. The probability of a transition is calculated by counting the number of transition occurrences in a block of demonstrations.

A rather different approach is used by Browne and Nicolescu (2012). In their algorithm each unit of action constitutes the root node of a different graph, and has as branches all sequences that led to it in the demonstrations. Recall of a task is made by finding, at each point of the execution, which graph has the currently observed sequence as a branch. The root node of the chosen graph is the next action to take.

Konidaris et al. (2011) assumes the existence of multiple starting points for a robot navigation task but a single goal location. The algorithm builds a graph shaped as a tree where each edge corresponds to a trajectory segment. The several courses of the tree are built by adding branches from the end to the beginning. If a newly observed trajectory segment is similar to the one encoded on the current branch, the two trajectories are merged. If not, the branch is divided and the graph continues to be constructed from the new branch.

5.2.2 Petri-Nets

Petri-nets are a particular type of graph structure commonly used to optimize processes that has also been employed in LfD. These graphs represent both the states and the transitions leading from one state to the other. As a modelling tool, they became popular mostly due to a graphical look that depicts processes very clearly and because of the existence of well-defined mathematical rules for optimization (Murata, 1989). In the LfD domain they were used by Chang and Kulic (2013) to encode a representation of a block piling task, where the task states was represented in the nodes and the transitions corresponded to the object trajectories leading from one state to another. Recall of the task was made by seeking the shortest execution path between an initial and a final task state. Zöllner et al. (2004) also employed Petri-nets to coordinate the arms of an

anthropomorphic robot in a task involving dual-arm object manipulation.

5.2.3 Hidden Markov Models

Hidden Markov Models (HMM) are stochastic models applied to processes following two principles (Elliott et al., 1995): 1) The probability of transition between two process states depends solely on the current state, disregarding past events, and 2) the current and past process states are always unknown and instead what is known is an "observation" which depends on distribution probability specific to each state. HMM can be used as task representations by defining each state as unit of action. Learning within an HMM is then the process of determining the values of the transition probabilities between the states.

HMM have been employed for learning tasks where a clear demonstration is difficult to provide and thus the demonstration does not correspond to the actual intentions of demonstrator (Chen, 1994). Because they assume a difference between an observation (demonstration) and the actual state (demonstrator's intention) they can extract the most likely sequence of intentions corresponding to the observations, even if the demonstration has errors. An early implementation was proposed by Hovland et al. (1996) for learning a task where a robot manipulation had to insert a peg into a whole. In their implementation the hidden states were the contact states of the peg with the walls of the hole. It was assumed that the sequences of observed contact states could result of a faulty demonstration not corresponding to the intended sequence. The transition probabilities were calculated through a classic HMM algorithm, the *Baum-Welch* method (Rabiner, 1989). The same algorithm was also employed by Kulic and Nakamura (2010); Kulic et al. (2012). In this case the purpose was to learn a sequence of full body motion primitives. In their implementation each HMM state correspond to a motion primitive. Whenever a new motion primitive is detected a new state is added to the HMM and the matrix containing all transition probabilities is extended. The *Baum-Welch* algorithm is then applied to incorporate the new demonstration.

Amit and Matarić (2002) proposed a different approach to learning sequences of

movements, where each sequence was formalized by a different HMM called a "learner". Whenever a new demonstration was provided, the observed sequence was compared against all the learners to discover which one was the most likely to produce that movement. The winner had its probability distribution adapted to better incorporate the new demonstration. After several demonstrations the probability distribution of each sequence learner converged to an optimal state representing a specific sequence.

Butterfield et al. (2010) also developed a HMM model of a sequence of motions. In their implementation each primitive was modelled as an expert function that depended on the state. To define the transition probability they simply counted the number of transitions between expert functions.

5.3 Hierarchical representations

Hierarchical models implement task representations whose structure is built to mirror the typical hierarchical organization of complex tasks (Figure 5.2). Normally, these representations are organized into different abstraction levels, with the bottom levels of the structure being filled with primitive-level actions like "reach" and "grasp" gestures (Friedrich et al., 1996), while the top levels encompass more abstract components, deemed as sub-tasks (e.g. "enter the elevator" in Miura et al., 2005). These sub-tasks are formed either by groups of motor level actions or by other lower-level sub-tasks.

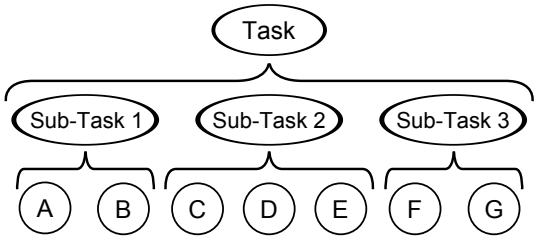


Figure 5.2: **Hierarchical organization in three levels.**

The singular fact about hierarchical task representations is that the position of a task component normally reflects its contribution towards an end-goal instead of the temporal or precedence relations observed in the demonstrations. Because of that they

tend to allow more flexible executions allowing order variations and parallel execution of sub-tasks (Cooper and Shallice, 2000).

However, hierarchical structures can be difficult to extract from demonstrations. Applications using hierarchical task representations often rely on predefined background knowledge about the task or additional information provided by a tutor to scaffold the development of the hierarchy. Both cases are addressed next.

5.3.1 Hierarchy based on task specific knowledge

Kuniyoshi et al. (1994) proposed a hierarchical task model for an application where a robot had to learn to stack a set of objects using a gripper. The hierarchical plan was derived from the sequence of demonstrated actions, form the contact relations between the objects and from background knowledge about sub-task dependencies. Aleotti et al. (2004) also proposed an algorithm for manipulating objects in a pick and place task. In this case the structure of the lower levels of each sub-task was predefined as a sequence of motor operations (e.g. move, rotate).

A rather similar approach was used by Miura et al. (2005). Their implementation assumed that the learning system must be pre-defined with hierarchical models for common tasks. The purpose of learning was to provide environment specific information that could not be pre-programmed. The model was organized into higher-level macro-operations that are divided into lower-level sub-operations. As an implementation example they choose a task where the robot had to learn to navigate towards an elevator and getting inside it. An example of macro-operations was "move inside the elevator" while sub-operations were of type "reach the elevator" or "press elevator button". To obtain environment specific knowledge the robot verbalized questions to the human, asking for instance the position of the elevator button, which the human demonstrated by pointing towards it.

5.3.2 Guidance from a tutor

Van Lent and Laird (2001) defined an airplane control task as a sequence of operators with specific pre-conditions and goal states. Each operator was constituted by a set of lower level sub-operators. In a first stage, the hierarchical structure was drawn by the demonstrator on a computer console. In a second phase, the demonstrator performed the task while the learning systems gathered information about context and actuation. At the same time, he annotated the data indicating which of the predefined task operators he was addressing. Annotations were also used by Garland and Lesh (2003) in a simulated cooking task, where multiple annotated demonstrations were provided. In this case the annotations provided structural information about the hierarchy, marking sub-tasks that could executed in parallel or which sub-sequences across demonstrations achieved the same result.

In the work of Saunders et al. (2006, 2007) tutor indications were used to scaffold the development of a hierarchical representation of a task where the robot had to navigate through an environment, grasping objects and placing them into containers. The learning followed a bottom-up approach. The teacher first demonstrated sequences of predefined motor level actions independent of environmental constraints (e.g. a "grab" sequence formed by the primitives "lower the gripper" and "close it"). The low level sequences were then labelled by the teacher. To build higher-level sub-tasks the tutor could use the labelled sequences (he could instantiate them through the labels) together with low-level primitives and/or other sub-tasks.

5.3.3 Graphs and hierarchies

Zöllner et al. (2005), used a combination of both hierarchies and graphs, in different abstraction levels. In the higher levels the task representation is constituted by a graph of sub-tasks. Each of the sub-tasks is formed by a sequence of basic operations that begins with a "pick", ends with a "place" and can include transport or tool handling operations. Each of the basic operations was in turn formed by a sequence of elementary

operators of type “approach” or “grasp”.

5.4 Final remarks

As it was shown, serial order plays a fundamental role in many robotic applications focused on task learning through demonstration. However, few have paid attention to the wide bibliography of cognitive models of serial order that has been produced in the last years. In the next chapter, a review is presented on the most important cognitive models of sequential action and the principles which support them.

Chapter 6

Models of sequential behavior: an overview

Since the seminal talk given by Lashley (1951), the mechanisms behind the production of sequential behaviour in humans have been the subject of wide discussion, encompassing the areas of cognitive psychology, neuroscience, computation and robotics. This led to a wide body of work with multiple models and dozens of studies conducted about several aspects of serial order. A detailed survey of this literature is beyond the scope of this thesis (see Hurlstone et al., 2013, for a deeper review). This chapter intends to provide a short overview, addressing the main classes of models and the relevance of their conceptual differences. Connectionist models are addressed as well as models that take into account the hierarchical organization of sequential behaviour.

6.1 Main conceptual approaches

Sequential behaviour has been a central research topic for the cognitive psychology community. During the last sixty years, several cognitive models for sequence learning and retrieval have been proposed. Henson (1998) presented a classification framework of these models into three major classes, according with the conceptual approach they

follow, namely, chaining, ordinal and positional.

6.1.1 The chaining approach

According with the chaining approach, the sequential order is stored through associations between consecutive elements of the sequence. When an element is retrieved, it triggers the recall of its successor, which will then trigger the next one, following a chaining process. Being a simple and intuitive mechanism, chaining has been the most prevalent theory of serial order and is the basis of multiple cognitive models (Lewandowsky and Murdock, 1989; Murdock, 1993). But despite its popularity pure chaining mechanisms (pairwise associations between contiguous elements) have been largely criticized (Lashley, 1951; Henson, 1998; Brown et al., 2000) since they fail to address some fundamental questions, namely:

i) How is the recall of the sequence started? If each element of the sequence is cued by its predecessor, how is the first element recalled, since it is not, in principle, cued by anyone?

ii) How can chaining encode repetitions? Since each item in the sequence is cued by its predecessor, repeated elements will cue two different successors. How is then possible to disambiguate between them?

iii) How can the model recover from an error? If one element is wrongly retrieved, it will cue a different successor, and the remainder of the sequence will probably be different from the correct one.

A solution for the first problem presented by several models is the use of external cues (Murdock, 1995). As for the second and third objections, the use of Compound Chaining (Elman, 1990; Murdock, 1993; Botvinick and Plaut, 2004) can be a fitting solution. According to this paradigm, each element of a sequence is cued not just by one single predecessor but by several ones. These compound cues allow to disambiguate

between elements when there is a common cue. Also, since the cueing function is distributed between more than one element, an erroneous retrieval does not necessarily lead to a chain of successive errors, since the items retrieved before the error can contribute to a correct selection of the elements following the error.

The issue of repetitions is also the basis of another important criticism, namely the lack of fitting between empirical data and behaviour predictions based on chaining theory. Specifically, it seems logic that in sequences with repetitions there would be a tendency for items sharing the same cue to be confused in recall, even when assuming the existence of a compound chaining mechanism. However there seems to be no evidence of such effect (Henson et al., 1996; Kahana and Jacobs, 2000).

On the other hand, a recent work by Solway et al. (2012) provided evidences that chaining models may well account for another empirical phenomenon, the temporal clustering effect. This effect was observed in experiments where participants were tested in sequence recall tasks. After committing an order error, participants were most likely to recall the item that followed the erroneously recalled item instead of the one belonging to that position. This is in accordance with other studies providing evidences for associative effects in serial recall (Hulme, 2003; Howard et al., 2009).

6.1.2 Ordinal theory

According to the ordinal theory, each element of a sequence is associated with a specific value of a common dimension. The position of each element within the sequence is defined by the position of its associated value relative to the values associated with the other elements. The typical implementation of this theory consists of a set of nodes, each representing one sequence item and with an activation value corresponding to its relative position within the sequence. The most popular method for retrieving the order is called Competitive Queueing (CQ) (Houghton, 1990). It implements a competition between the nodes and a subsequent suppression of activated nodes. The recall starts with the element with the highest activation and ends with the element with the lowest activation (Grossberg, 1978). Classical examples of these models are the the primacy model (Page

and Norris, 1998) and the phonological loop (Burgess, 1992; Burgess and Hitch, 1999; Hartley and Houghton, 1996). The hierarchical model of Cooper and Shallice (2000) also relies heavily on competitive queueing mechanisms.

The attractiveness of ordinal models as a serial order theory derives from how naturally these models can replicate empirical phenomena of serial behaviour in humans, specially some classes of errors. Anticipation errors for instance (recall of a sequence element ahead of its correct position) can be easily induced by introducing noise in the competitive process between nodes, allowing for nodes with less activation to "win" the competition (Page and Norris, 1998; Cooper and Shallice, 2000). The primacy effect, which states that errors are less frequent in the beginning of the sequence is also presented by some models like the phonological loop (Burgess and Hitch, 1999). In this case each sequence node is queued by a context/temporal signal. Since in the beginning of a list there are fewer queued items to compete, errors are less frequent.

Other types of errors, are more hardly explained. Botvinick and Plaut (2004) point the example of the slips-of-action. This type of error can be described as the intrusion in a sequence of not just a single erroneously element, but an entire subsequence belonging to another sequence. Empirical observations show that this error is often the result of shared context or common preceding elements (see Norman, 1981). However, it is not clear how the competition process would allow the intrusion of an entire set of actions.

6.1.3 Positional theory

According to this theory, sequential order is stored by associating each item of a sequence with a position within an ordered list. To recall the sequence, the list must be activated and the item associated with each position is retrieved. A simple implementation is represented by a predefined chain of connected nodes, each one associated (through learning) with a sequence item. By stepping through the chain of nodes and retrieving the item associated with each node, the sequence is recovered (Shiffrin and Cook, 1978). This theory is somewhat similar to chaining, with the important difference that associations are not established between sequence items, but between positional rep-

resentations and items. Functionally, this has deep implications (Henson et al., 1996). Repetitions for instance are effortlessly implemented, since sequence items can be associated with more than one position. Also, retrieval errors are less problematic, because an error in retrieving the item associated with one node in the chain has no implications in the retrieval of the element corresponding to the next node.

However, classic positional models are often criticized since they do not answer the question of how the positional list itself is established (Brown et al., 2000). A comprehensive example of this problem is the "box" model from Conrad (1965), one of the earliest accounts of a positional model. Its core assumption is the basilar of positional theory: sequential items are stored within a set of ordered "boxes", so that the first element of the sequence goes to the first box, the second goes to the second box and so on. However, the model does not answer the question of how, the order of the boxes itself is established. The same is true for another classical example, the perturbation model of Lee and Estes (1977, 1981); Estes (1997) which despite addressing several empirical observations like the occurrence of transposition errors, leaves unattended the question of how positional representations are set in their correct position.

More recent positional models focused on this issue. Henson (1998) proposed the Start-End Model (SEM) where the position of each element within a sequence is coded as a function of the distance to both the start and end of the list and stored together with the information about the item. Brown et al. (2000) presented the Oscillator-based Associative Recall (OSCAR), a connectionist model, where sequence items are associated with positional codes generated by a set of oscillators actuating at different frequencies, working as a sort of a clock. Since this codes are time-dependent, the model is capable of encoding not just the sequential order, but also the time relations between occurrences of the respective items. In fact, the use of time dependent context representations proved to be a sufficiently attractive feature to be included in several models (Burgess and Hitch, 1999; Howard and Kahana, 2002), although experiments conducted by Nimmo and Lewandowsky (2005); Lewandowsky et al. (2006, 2008) have challenged time-based theories, by questioning one of their fundamental empirical predictions, namely the one

stating that temporal isolation of sequential items during the presentation of a sequence, benefits memorization.

An important note

It is important to mention that the classification of models into these three categories is not categorical, in the sense that some of them can be framed in several theories. For instance, the phonological loop (Burgess and Hitch, 1999, 2006) uses a competitive queuing mechanism in its "item layer" while also making use of a context/timing information, therefore falling under the positional label. And since the output of recall also becomes part of a composite input that will cue the retrieval of the next item, it is also comparable to chaining mechanisms (Brown et al., 2000). Another example is the OSCAR model. Due to its use of a time dimension as a contextual cue it can be considered as a positional model (Henson, 1998). However, the time context is translated into an activation gradient whose items compete for activation and in that sense it accounts as a Competitive Queuing model (Brown et al., 2000).

The importance of this three-folded classification is that it gives a good account of the main conceptual mechanisms behind the majority of the sequential behaviour models. Each of these theories seems to account well specific classes of empirical phenomena. Therefore, it seems likely that a comprehensive theory of serial order (still not reached) could incorporate all three mechanisms.

6.2 Connectionism: a major trend

One important trend in the development of serial order models has been the use of Recurrent Neural Network (RNN). As a modelling tool, neural networks possess two main advantages:

- They implement a biologically plausible representation of sequential states (through neural activation patterns) and of the relations between them (through the syn-

aptic connections), thus providing an embodied account of a mechanism for serial order. This view contrasts to many of the models mentioned thus far that are described on the behavioural level, with no concerns about the neuro-physiological substrate that implements them.

- Learning is implemented as the outcome of a simple (and biologically plausible) weight adaptation process. Again, this constrasts with other models that rely on more or less complex and logical mathematical processes to perform sequence encoding.

Typically, RNN models of serial order are constituted by three layers of neurons, an input, an output and a hidden layer, with recurrent interactions between them. The distribution of the the recurrent connections is the distinguishing element between most of the implementations. For instance, one of the earliest accounts of a RNN for serial order Jordan (1986) had connections linking the output directly to the input. The Simple Recurrent Network (SRN) implementation of Elman (1990) directly mapped the hidden layer to a contextual input layer, allowing the network internal representation to depend not only on the input but also on its own previous state, thus making them sensitive to temporal context.

Because the network states heavily depend on its preceding state, RNN are commonly referred as implementing chaining-like mechanisms and thus suffer from the same limitations (Henson, 1998; Brown et al., 2000; Cooper and Shallice, 2000). Despite this objection already addressed in subsection 6.1.1, RNN models of serial order remain influential, mostly due to the work of Botvinick and Plaut (2002, 2004) who implemented a SRN model for learning and recall of a hierarchical organized task. They obtained significant results consistent with empirical data, like the occurrence of slips-of-action, omission, repetition and anticipation errors, among others.

A distinctive feature of Botvinick and Plaut's model is their approach to the hierarchical nature of many sequential tasks. The SRN does not have a specific representation of hierarchical structure. Botvinick et Plaut consider that the hierarchical

character of the sequence execution is an emergent property of the recurrent network. This view directly contrasts with many approaches to the problem of sequential order, as it is shown next.

6.3 Hierarchical models

The hierarchical organization of sequential behaviour into nested subroutines is a known fact already referred by Lashley (1951) and has been a concern for modellers ever since. Several models have been proposed to account for this phenomenon, including some of those already addressed in the previous sections. One of the earliest was the "test-operate-test-exit" (TOTE) from Miller et al. (1960). According to this model, each TOTE is a goal-directed unit of action. In each of these units the achievement of the goal is constantly being tested and in case it has not been achieved, a corresponding operation is called. The hierarchical character comes from the fact that each TOTE can call other TOTEs, thus forming a tree of goal directed actions. Another model proposed by Estes (1972) implemented a hierarchy of "control elements" where an element of an above unit activated a set of sub-elements below, while an ordinal mechanism dictated the order between them. Several later models built on the work of Estes adding contributions on specific elements of the model (see Botvinick, 2008, for a review).

One of the most recent and comprehensive examples of this tree-like approach is the model proposed by Cooper and Shallice (2000). It is organized around three levels of goal-directed segments of action named as "schemas": A lower level that includes the execution of body primitives such as "reach", "preshape" or "enclose", a higher level fitting more abstract and general activities like "preparing coffee" and an intermediate level accounting for more simple activities like put "sugar into coffee" or "milk into coffee". On each level, lateral inhibition ensures competition between schemas that cannot be activated in parallel. Cooper and Shallice (2000) provided extensive results on a coffee preparation task successfully replicating a considerable amount of empirical data, specially in relation to errors.

The key feature of Cooper and Shallice model as well as the ones following the approach of Estes (1972) is that the architecture is in itself hierarchically structured, directly mapping the task structure. Action elements or schemas have localist representations, being represented by individual nodes who receive input from nodes in higher levels and supply it to nodes in lower levels. However, this approach raises the fundamental question of how exactly this type of structure can emerge (Botvinick and Plaut, 2004). Concretely, the problem is unsolved which process would lead to the emergence of a unit at a higher level as a representation of a composite structure of units in a lower level.

The connectionist model of Botvinick and Plaut (2004) that builds on the work of Jordan (1986) and Elman (1990) follows a different paradigm. Rather than having an explicitly hierarchical architecture the internal representations and the dynamics evolve, to implicitly encode the hierarchical structure of the task. Representations are not localist but rather distributed as activation states of the network. Since the learning process is intrinsic, it does not suffer from the same limitations as the hierarchical structured models. However, the weight adaptation mechanism that originates the emergence of the hierarchical task structure can be very slow (O'Reilly and Frank, 2006) and require a very large and varied set of demonstrations (Cooper and Shallice, 2006) in order to reach an acceptable degree of structure [1].

Recent evidences of topographical organization of brain structures coding different levels of behavioural structure (Fuster, 2004) have raised a particular interest on hierarchically structured architectures and possible mechanisms underlying its emergence. In an effort to address this issue, Hierarchical Reinforcement Learning (HRL) techniques have recently been explored as a functional tool for creating hierarchical structured task representations (O'Reilly and Frank, 2006; Botvinick et al., 2009). According to this paradigm, the policies derived from the Reinforcement Learning rules depend on which specific high level representation is activated. An example would be to have an agent

[1] The differences between the two approaches were the subject of an interesting exchange of arguments between Botvinick/Plaut and Cooper/Shallice that can be followed in their papers (Cooper and Shallice, 2000; Botvinick and Plaut, 2004; Cooper and Shallice, 2006; Botvinick and Plaut, 2006).

navigating on a set of rooms, following a high level policy, and keeping different sub-policies for navigating in each specific room.

Although HRL seems to be a promising framework for addressing the hierarchical structure of sequential tasks, the neural substrate (and a subsequent model) behind a possible HRL-like behaviour remains unclear (Botvinick, 2012).

6.4 Application to robotics

As it was mentioned, the majority of current task learning robotic applications, pays no attention to existing neuro-cognitive models when it comes to encode the sequential order of task events. One exception is the work of Sandamirskaya and Schöner (2010). In their task, a mobile robot navigates in an arena approaching coloured markers in a sequential order learned from demonstration. The sequence of colours is encoded through a DNF-based positional model. Another exception is a study conducted at the MARLab (Ferreira et al., 2014). A DNF model that combines an activation gradient and a competition process to encode both the serial order and the timing of events has been applied with ARoS to learn a musical sequence.

6.5 Final remarks

The purpose of this chapter was to summarize some of the most important theoretical and empirical concepts in the sequential behaviour research as well as the most relevant psychological and computational models. They form the theoretical background for the development of the DNF model for learning sequential tasks presented in the next two chapters.

Part IV

Sequence Learning

Chapter 7

Model of learning sequential tasks - Design

This chapter describes the model of sequential actions developed for the integration in the cognitive architecture for Human-Robot joint action. The chapter begins with an overview of the basic neuro-scientific concepts underlying the model, followed by a more detailed description of each of the modules and an explanation of its functioning principles. Transversely, the relations between its several components are described as well as the implications for its overall functioning. The chapter finishes with a detailed description of the mathematical implementation of the model.

As mentioned in the Introduction (Section 1.4), the work presented in this thesis applies the model of serial behaviour at two different levels of abstraction: i) learning representations of sequential tasks and ii) encoding sequences of motor primitives for anticipating the outcome of motor actions. For the sake of clarity, the model's description in this chapter focuses solely on the learning and retrieval of sequential tasks. This application is the most complex one and implies the usage of all model properties. In Chapter 9, it is shown how the model can be easily adapted to encode sequences of motor primitives.

7.1 General overview

The first question to address in the development of a sequence model is what constitutes an item of the sequence. In ARoS cognitive architecture (Figure 1.1), the task is represented within the shared-task knowledge module (or "common sub-goals layer" in Figure 1.1) as a set of sub-goals that need to be achieved sequentially. Maintaining coherence with this approach, the model for task learning represents the task in a similar fashion, making each item of the sequence a sub-goal representation. This design choice is inspired by experimental findings by Bekkering et al. (2000) showing that gesture imitation in children between 3 and 5 years old is goal directed, favouring the purpose of the gestures rather than the concrete movements employed to achieve it (this was later backed up with neuro-imaging data from Koski et al. (2002)). More recent findings in cognitive neuroscience, support the notion that specific areas of brain encode the goals of actions, disregarding the means to achieve it. For instance, a study by Tanji et al. (2007), demonstrated that areas of the Prefrontal Cortex (PFC), specifically concerned with behavioural planning, encode the end-result of object-directed behaviour (e.g. a change in the state of an object) and not the movement required to achieve such change. A Functional Magnetic Resonance Imaging (fMRI) study by Newman-Norlund et al. (2007) showed that a particular area in the left anterior inferior parietal sulcus responds to functional goals of grasping actions independent of specific movements.

7.1.1 Goal representations

In the LfD literature, learning of a task as a sequence of end-states of goal-directed actions, is often considered as an alternative to the imitation of body movements, since it avoids the problem of matching movements from a demonstrator to the ones of a robot (Calinon et al., 2005; Saunders et al., 2007). This "correspondence problem" (Nehaniv and Dautenhahn, 1999) is particularly significant if "demonstrator" and "learner" have different embodiments (Alissandrakis et al., 2002). Goal representations also suit well

within the joint-action approach. Given that the sub-goals of a collaborative task can, in principle be achieved by any of the partners, the updating of the internal task representation of the robot in response to changes in the world must be agent independent (Decety and Sommerville, 2003; Newman-Norlund et al., 2007).

Another important question is how to distinguish already achieved and future sub-goals. At this point one must take into account that the model needs to deal with the real time constraints of a LfD paradigm. This implies that the model must be adapted in synchrony with the state of the task at hand. To this end, a dual representation approach was adopted, separating representations of sub-goals already completed from the ones still to be achieved. Concretely, the two types of representations are formalized as neuronal pools in two different neural fields, designated as "past" and "present" fields (Figure 7.1). This allows the model to maintain a working memory in the "past" field required during task execution and at the same time to make predictions in the "present" field about the next step to take. It also permits the learning of associations between successively achieved sub-goals during the observation (this will be further explained in Section 7.2).

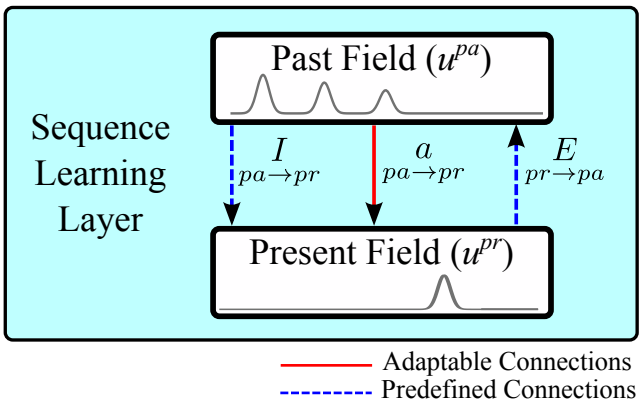


Figure 7.1: **Diagram of the connections between the fields u^{pa} and u^{pr}.**

The implementation of separate representations for "achieved" and still to be accomplished goals is consistent with neurophysiological findings. For example, studies from Genovesio et al. (2006) on the firing patterns of neuronal populations in PFC of monkeys suggest that past and future sub-goals have indeed separate representations.

A subsequent study showed significant correlations between the firing patterns of these neuronal populations, suggesting the existence of strong functional relations between these representations (Tsujimoto et al., 2008). The model presented here mimics these experimental findings, since the population dynamics of the "past" and "present" fields are strongly related through a dense network of connections responsible for defining the behavioural constraints of the model.

7.1.2 The sequence model paradigm

From the cognitive modelling point of view, a compound chaining approach is applied since the activation of a specific sub-goal representation is the result of excitations from multiple previously achieved sub-goals. Although usually compound chaining, as opposed to simple chaining, is viewed as a solution for the problem of repetitions within a sequence (Murdock, 1993) here it assumes a different role. It allows the model to deal with tasks where the order of execution of the sub-goals is not unique. Take as example the assembly task of a toy vehicle depicted on Figure 7.2. The task consists of mounting a pair of wheels fixed by nuts on a base and placing four columns vertically on the base with a platform on top. The columns can be inserted in any order whereas the top can be mounted only when all columns are in place. A compound chaining approach allows one to learn both flexibility in task execution and long-term dependencies.

For the learning a social guidance paradigm is applied. The tutors guide the observational learning with additional verbal cues. The typical learning procedure is as follows (Figure 7.3): one or more tutors demonstrate possible sequences of execution of the task sub-goals. The robot then tries to reproduce the task. At every step, it verbally communicates a prediction about the subsequent sub-goal to the human tutor who then provides immediate verbal feedback. A negative feedback signalling a prediction error automatically triggers an adaptation process that increases the time window for associative learning in subsequent demonstration trials. As a result, a larger number of executed sub-goals becomes associated with a certain prediction. The increased dependence on execution history in multiple demonstration/execution cycles will ultimately eliminate

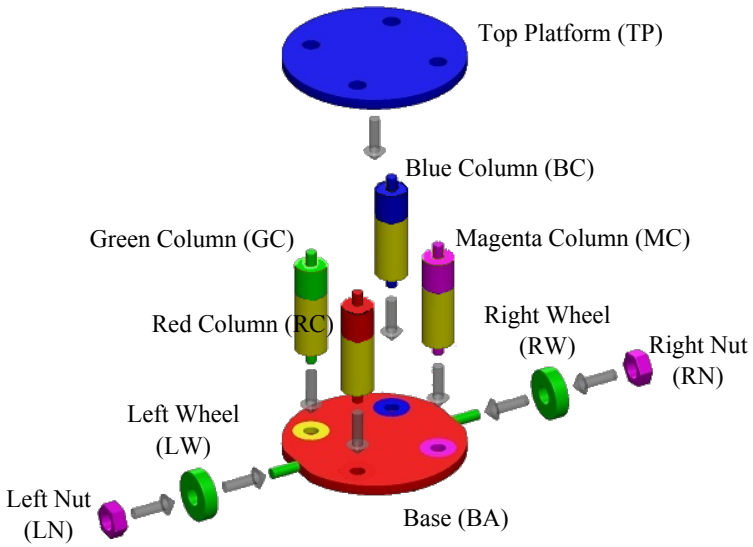


Figure 7.2: **Diagram of the Toy Vehicle.** Each wheel must be placed before the corresponding nut and the columns must all be in place before the top platform can be mounted.

the prediction error.

7.1.3 The rehearsal mechanism

The formation of associations between the neuronal representations is made through adaptation of synaptic connections between the DNFs. In this sense, the model can be seen as a modular neural network, organized into layers. Since the synaptic learning usually must be slow to be robust (Botvinick and Plaut, 2004) multiple demonstrations of the same sequence are required. This can be boring and time consuming for a human user (Ehrenmann et al., 2002). To avoid this problem, the sequence learning ability was extended through a short-term memory module, capable of autonomously recalling a demonstrated sequence from memory, thus avoiding a large number of demonstrations by the user.

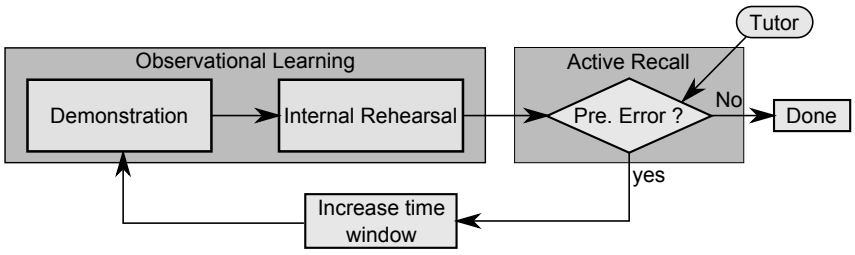


Figure 7.3: **Diagram of the learning process.** The process begins with a period of Observational Learning followed by an Active Recall of the sequence. If a prediction error occurs, the window for associative learning during a new demonstration is automatically increased. Correctly memorized sequences are internally recalled during a rehearsal period which drives the learning of synaptic connections between field representations.

7.1.4 Chunking representations

The model was further extended to form higher-level sub-task representations (chunks) corresponding to groups of sub-goals that must be completed before a subsequent sub-goal can be completed (Sakai et al., 2003; Hard et al., 2006a). Specifically, it is assumed that prediction errors are salient events whose occurrence is associated with boundaries between groups of sub-goals (Reynolds et al., 2007; Zacks et al., 2011). Thus, prediction errors constitute natural breakpoints in the linear processing of a sequential task and are used in the model as event boundaries to segment the flow of sub-goals into chunks.

In the model extension described here, these sub-task are represented through the same dual representation approach employed for the individual sub-goals, with "past" and "present" sub-tasks being coded by populations in different DNFs.

Figure 7.4 presents a diagram of the complete model. It is constituted by four modules: The Sequence Learning Layer (SL), where the sub-goals are represented, the Short-Term Memory Layer (STM) where the last observed sequence is encoded, the Task Monitoring Layer (TM) where a memory of an error is kept and the High-level Cognitive Memory Layer (HCM) where the sub-tasks are represented.

In the next sections, the several modules of the model are described in more detail and an explanation of their functioning is given.

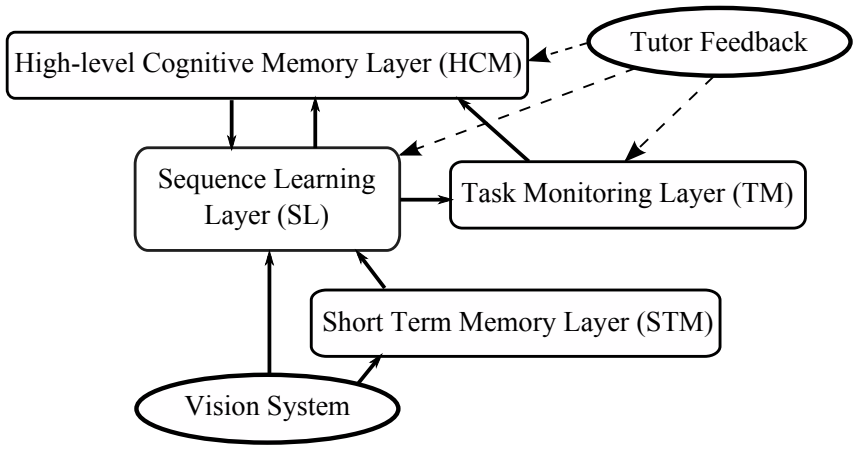


Figure 7.4: **Schematic view of the DNF model for learning sequential tasks.** The model is formed by four layers: The Sequence Learning Layer (SL), where the sequence of sub-goals is encoded, the Short-Term Memory Layer (STM) that maintains a representation of the last observed sequence, the Task Monitoring Layer (TM) where a memory of errors is maintained and the High-level Cognitive Memory Layer (HCM) where chunk representations are formed. Inputs from the vision system and input connections between layers are depicted in solid lines while the external control signal resulting from verbal feedback by the human tutor is represented by dashed lines.

7.2 Learning sub-goal sequences

This section describes the SL Layer where the sequences of sub-goals are encoded. It also details the dynamics and the interaction process that allows the learning of the serial order of a task.

7.2.1 The vision input

The vision system provides information about the current state of the assembly work to the DNF model. A perceived change in the state of the task (the execution of a sub-goal) is modelled as a Gaussian pattern centred on a sub-population in the dynamic field representing the specific sub-goal (Figure 7.5). The input from the vision system is formalized by $v^k\left(x^k,t\right)$ where k is the label identifying the field to which the contribution refers to.

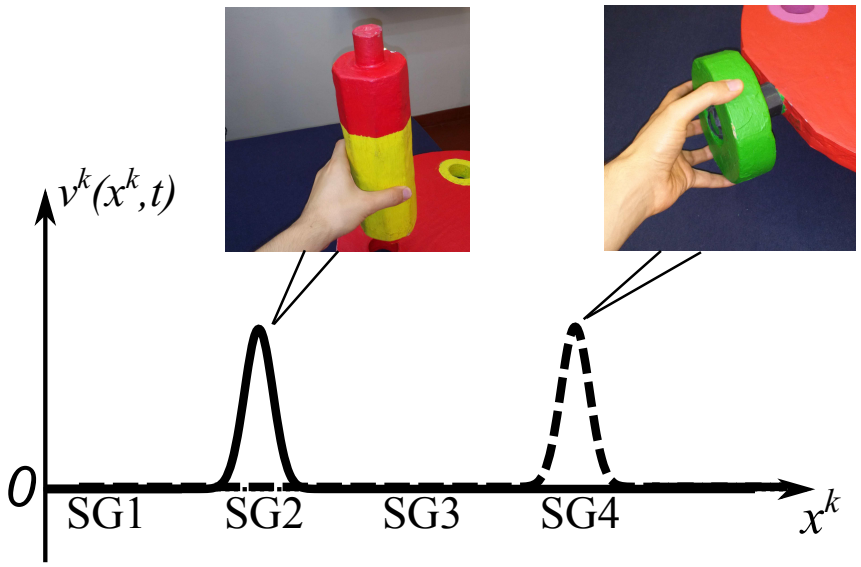


Figure 7.5: **Example of the visual input**. The completion of a sub-goal triggers a Gaussian input pattern $v^k(x^k,t)$ to the neural populations in the fields encoding the specific sub-goal (SG2 and SG4)

7.2.2 The Sequence Learning Layer (SL)

The Sequence Learning Layer represents the core part of the learning model (Figure 7.6). Its dynamics assumes the existence of two different operation modes: 1) a learning mode, in which the model learns a task from demonstration; 2) a recall mode, in which the model makes predictions about the next sub-goal based on the ones already achieved.

The model is constituted by two coupled dynamic neural fields, representing all possible sub-goals of the task. As mentioned previously, these DNFs are referred to as "past" and "present" fields respectively. The "past" field, $u^{pa}(x^{pa},t)$, maintains a working memory of the state of the task, representing already completed sub-goals. The "present" field, $u^{pr}(x^{pr},t)$, represents sub-goals that are currently being observed or whose accomplishment is predicted by the model. The difference between a prediction and the actual observation is established in terms of the amplitude of the peaks. The field dynamics establishes localized activation patterns corresponding to predictions with an activation strength below a threshold λ^{pr} (see SG2 in Figure 7.6).

The inputs $s^{pa}(x^{pa},t)$ and $s^{pr}(x^{pr},t)$ to fields $u^{pa}(x^{pa},t)$ and $u^{pr}(x^{pr},t)$, re-

spectively, represent the summed contributions from the vision system $v^{pa}\left(x^{pa},t\right)$ and $v^{pr}\left(x^{pr},t\right)$ and contributions from connected model layers.

The two DNFs interact through three sets of synaptic connections (Figure 7.6):

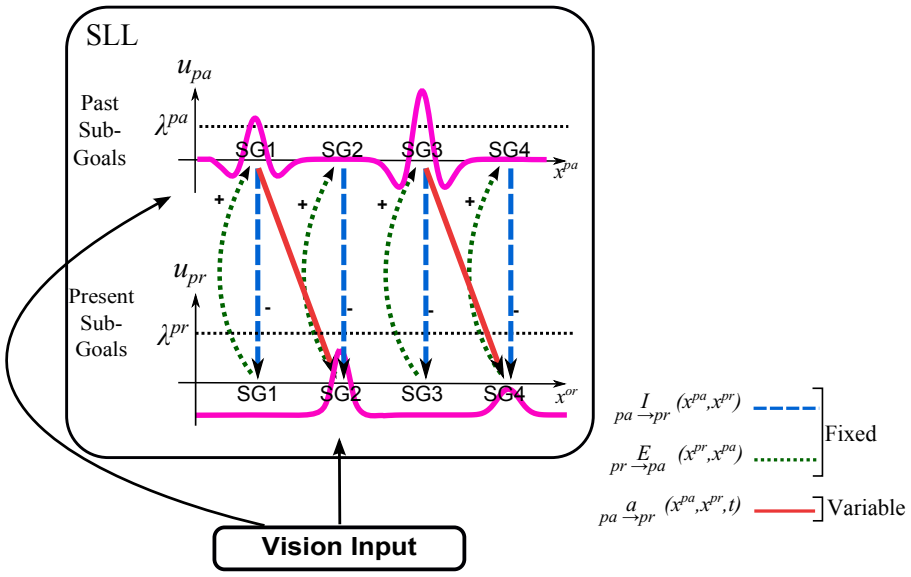


Figure 7.6: Diagram of the Sequence Learning Layer (SL).

i) $\underset{pa\rightarrow pr}{I}\left(x^{pa},x^{pr}\right)$: Predefined inhibitory synapses connecting u^{pa} to u^{pr};

ii) $\underset{pr\rightarrow pa}{E}\left(x^{pr},x^{pa}\right)$: Predefined excitatory synapses connecting u^{pr} to u^{pa};

iii) $\underset{pa\rightarrow pr}{a}\left(x^{pa},x^{pr},t\right)$: Adaptable excitatory synapses connecting u^{pa} to u^{pr};

The roles of the predefined and of the adaptable connections are explained next. For facilitating the reading, some simplifications are made in the mathematical labelling of the model within the text. Functions representing field activity will be represented solely by the letter "u" with the label in superscript. Thus, for instance $u^{pa}\left(x^{pa},t\right)$ is replaced with u^{pa}. Likewise, the functions representing the input to a field will be represented by the letter "s" with the label in superscript. Thus, for instance $s^{pa}\left(x^{pa},t\right)$ is replaced by s^{pa}. The sets of synaptic connections linking dynamic fields are simplified in a similar manner, for instance $\underset{pa\rightarrow pr}{a}\left(x^{pa},x^{pr},t\right)$ is replaced by $\underset{pa\rightarrow pr}{a}$. These

simplifications applied only to plain text. In the mathematical equations the complete notation is kept.

Predefined connections

The inhibitory and excitatory synaptic connections, $\underset{pa\to pr}{I}$ and $\underset{pr\to pa}{E}$ respectively, couple neuronal populations from u^{pa} and u^{pr} coding the same sub-goal. They produce an inter-field dynamics reflecting the following two assumptions:

A.1 An accomplished sub-goal represented in the past field should not be activated again in the present field. This is guaranteed by the inhibitory connections $\underset{pa\to pr}{I}$, through which above threshold populations in u^{pa} ($u^{pa} > 0$) inhibit their counterparts in u^{pr} (connections coloured in blue and marked with a minus sign in Figure 7.6).

A.2 A sub-goal currently being observed and therefore active in u^{pr}, must activate the corresponding representation in u^{pa}. This is ensured by excitatory connections $\underset{pr\to pa}{E}$ that propagate activity above threshold ($u^{pr} > \lambda^{pr} > 0$) to u^{pa} (connections coloured in green and marked with a plus sign in Figure 7.6).

Interaction between populations coding the same sub-goal in the past and present fields

When observing the achievement of a certain sub-goal, the corresponding populations in u^{pa} and u^{pr} both receive visual input v^{pa} and v^{pr} respectively. However, depending on the resting state of the fields the populations response is different. While in u^{pr} the resting level is fixed and chosen to guarantee that visual input alone may drive population activity beyond the threshold λ^{pr}, the resting level function $h^{pa}\big(x^{pa},t\big)$ in u^{pa} is assumed to have two different global baseline values for the two operating modes of the model. In the learning mode, the baseline level $h^{pa}_{bas} = H^{pa}_{low} < 0$ is lower than during task recall ($h^{pa}_{bas} = H^{pa}_{high} < 0$). This implies a difference in the time course of the activations in

the two DNFs. With a low baseline ($h^{pa}_{bas} = H^{pa}_{low}$), the field u^{pa} takes longer to respond to the visual input (Figure 7.7). This gives a processing advantage to the corresponding population in u^{pr} which reaches the threshold λ^{pr} before a supra-threshold pattern in u^{pa} evolves.

In the recall mode (Figure 7.8), when the baseline value is high ($h^{pa}_{bas} = H^{pa}_{high}$), the field u^{pa} responds faster to the visual input and the formed peak immediately inhibits the corresponding population in u^{pr} preventing the formation of a peak.

In both case, the development of a peak in u^{pa} representing a newly achieved sub-goal causes the adaptation of the resting level h^{pa} of the supra-threshold neurons. This resting dynamics leads to continuous decay of population activity which, however, remains supra-threshold to implement the memory function.

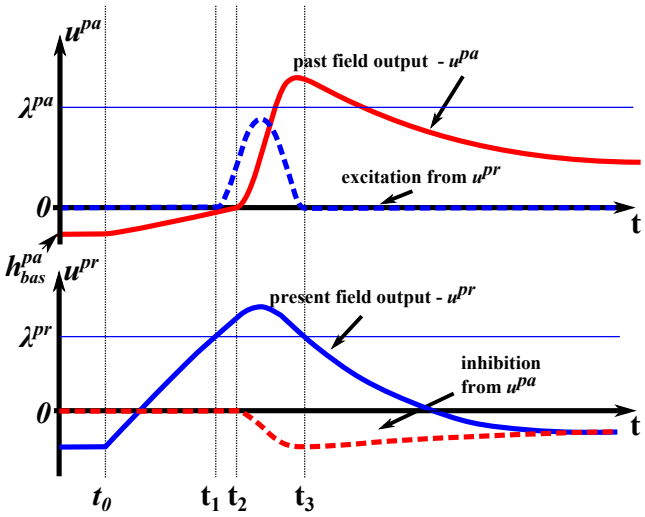


Figure 7.7: **Interaction process between neurons in u^{pa} and u^{pr} for $h^{pa}_{bas} = H^{pa}_{low}$.** In (t_0), an input from the vision system representing the achievement of a sub-goal starts to drive the population activity in u^{pr}. In (t_1), the activity in u^{pr} reaches λ^{pr} and starts to excite the corresponding population in u^{pa}; In (t_2), the activity in u^{pa} reaches the threshold and begins to inhibit u^{pr} through the predefined inhibitory connections $I_{pa \to pr}$; In (t_3), the inhibition from u^{pa} has suppressed activity in u^{pr} below λ^{pr}. The activity peak in u^{pa} decays as a result of the localized adaptation of the resting level h^{pa}.

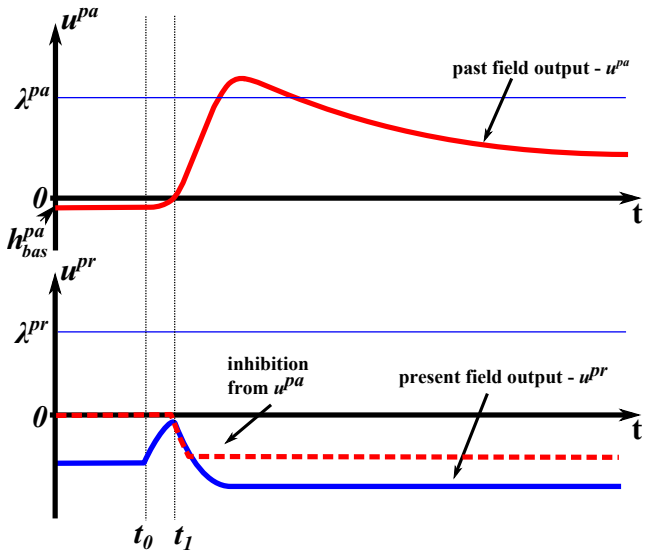


Figure 7.8: **Interaction process between neurons in u^{pa} and u^{pr} for $h^{pa}_{bas} = H^{pa}_{high}$.** In (t_0), the input from the vision system is presented. In (t_1), the activation in u^{pa} reaches a positive level and starts to inhibit the population coding the same sub-goal in u^{pr}.

Adaptable connections

A Hebbian learning rule is applied to establish synaptic links $\underset{pa \to pr}{a}$ between population representations in u^{pa} and u^{pr}. Learning takes place whenever population activity is above the learning thresholds (Seitz and Dinse, 2007), λ^{pa} and λ^{pr}, respectively (Figures 7.9a and 7.9b). Initially, this happens to occur only when, during task observation, vision input v^{pr} directly drives the population in u^{pr}. After learning, the newly established links $\underset{pa \to pr}{a}$ from one or more sub-goal representations in u^{pa} are sufficiently strong to trigger a population response above-zero in u^{pr}. During recall, this corresponds to a prediction of the next sub-goal.

The threshold λ^{pr} that is here used as a learning threshold is the same employed to distinguish between a prediction peak and a peak caused by the actual observation of the corresponding sub-goal. The same value was used for the sake of simplicity.

Since memory representation in u^{pa} decays over time, only those sub-goals that have been recently achieved and whose peaks are still above the learning threshold λ^{pa} will become linked to active sub-goal representations in u^{pr} during observation. By

changing the decay rate $\tau_{h^{pa}}$, one can effectively control the time window for associative learning. In the robotics application, a prediction error decreases the decay rate, resulting in learning longer term dependencies between past and present events.

Adapting the decay rate of the memory representations is not the only way to increase the time window for learning. For instance, increasing the gain of the population response to input could have been chosen a well, since a larger response amplitude with a constant decay rate would also result in a longer time window for learning. Controlling the decay rate has been chosen for the present robot implementation for simplicity.

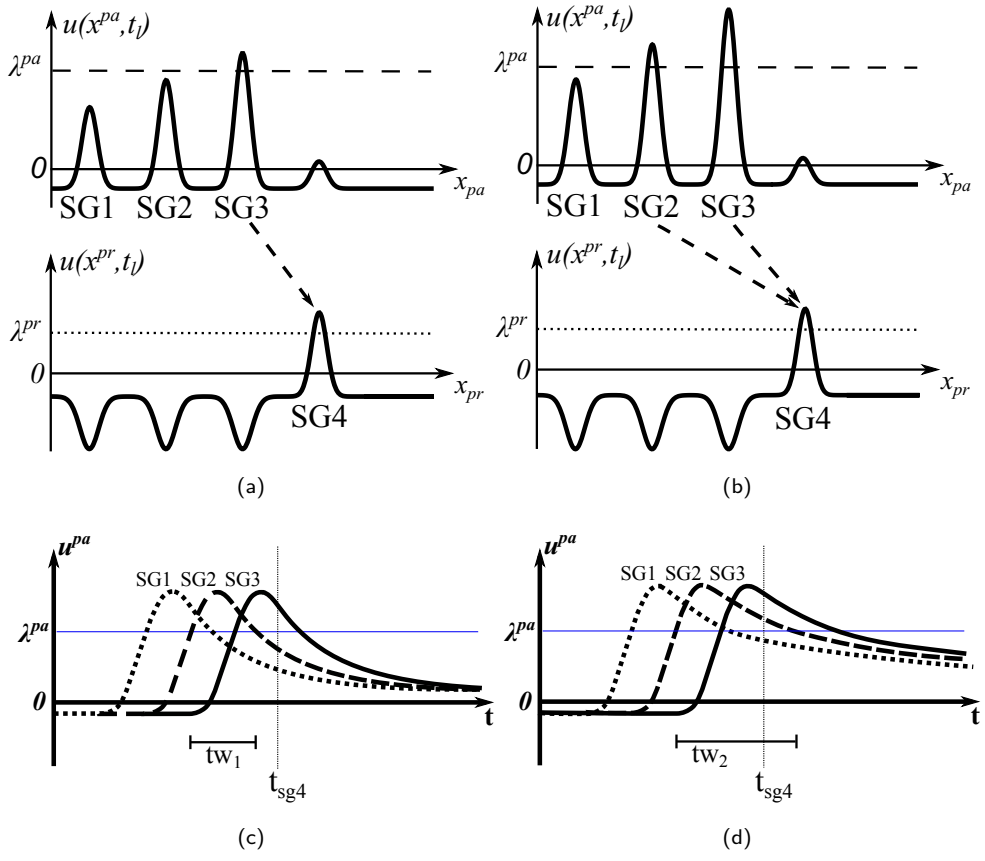


Figure 7.9: **Learning with different** $\tau_{h^{pa}}$. a) and b) depict the field activations when a sub-goal SG4 is observed at time t_{sg4} with $\tau_{h^{pa}} = T_1$ and $\tau_{h^{pa}} = T_2 > T_1$ respectively. c) and d) show the time course of the population activities encoding past sub-goals in u^{pa} for the same two situations. With $\tau_{h^{pa}} = T_1$ the time window tw_1 supports simple chaining (SG3 $\rightarrow$ SG4). With $\tau_{h^{pa}} = T_2 > T_1$ the time window is increased ($tw_2 > tw_1$) and associations are also established to SG2 (compound chaining).

7.2.3 The social learning paradigm

The learning process occurs through social interactions between the robot and the human tutor. It begins with an "Observational learning" period during which the model is in its learning mode and the tutor demonstrates the sequential task. This is followed by an "Active Recall" period in which the model is in the recall mode. The recall of the sequence starts with the tutor presenting the first object (e.g. the Base BA). The robot then verbalizes its prediction about the next sub-goal and waits for verbal feedback from the tutor. If positive feedback is provided robot and human will interact to execute the specific sub-goal, which in turn will lead to a new prediction. This interplay between verbal communication and joint execution will continue until the end of the sequence is reached.

If a prediction error occurs the human provides negative feedback. The time scale parameter $\tau_{h^{pa}}$ is updated and a new "Observational learning" period follows, during which the tutor demonstrates the same sequence again. Because of the increased time window, more past sub-goals in u^{pa} now become associated with a specific prediction.

The alternation between periods of "Observational learning" and "Active Recall" occurs until no prediction errors occur anymore .

7.2.4 Mathematical formulation of the model

Field equation

Each layer of the model is formalized by one or more DNFs. The activity of the neurons within each field is defined by an equation adapted from Amari (1977):

$$\tau^k \frac{\partial\, u^k(x^k,t)}{\partial t} = -\, u^k(x^k,t) + h^k(x^k,t) + \zeta^k_{stoch}(x^k,t) + \\ + f_\beta\Big[\, u^k(x^k,t)\Big]\left(\int_{x^k} w_k(x^k - y^k) f_0\Big[\, u^k(x^k,t)\Big] dy^k + \, s^k(x^k,t)\right), \tag{7.1}$$

where u^k is the activity of neuron x^k of the field k at time instant t. h^k is the resting level towards which the field activity relaxes in the absence of external input, that in some fields is defined as a function of time. The parameter $\tau^k > 0$ is the relaxation constant. Different from the original Amari equation, a gating term $f_\beta\Big[\, u^k(x^k,t)\Big]$ is applied to the input s^k and the interaction term. This gating term ensures that the field activity does not reach strongly negative values (below β) even with a strongly inhibitory input, which would significantly increase the field response time to future positive inputs. The stochastic noise term ζ^k_{stoch} forces a decision when two very similar inputs are competing.

As explained before, each field has a different input term, s^k, defined as a sum of input from the vision system and from other fields, mediated by predefined interconnections. The resting level h^{pa} of the past field is subject to both local and global adaptation during the learning process. Next, the equations defining the inputs s^{pa} and s^{pr}, the resting level h^{pa} and the synaptic adaptation of $\underset{pa\to pr}{a}$ are described.

Input to field u^{pa}

The input s^{pa} to field u^{pa} is defined by

$$s^{pa}(x^{pa},t) = C^{pa} v^{pa}(x^{pa},t) + \int f_{\lambda^{pr}}\Big[\, u^{pr}(x^{pr},t)\Big] \underset{pr\to pa}{E}(x^{pr},x^{pa}) dx^{pr}\ . \tag{7.2}$$

The input s^{pa} is the sum of Gaussian patterns supplied by the vision system (v^{pa}) mediated by the gain constant C^{pa}, which is hand-coded to regulate the strength of the visual component. The integral term describes the propagation of activity u^{pr} through

the predefined excitatory connections $\underset{pr \to pa}{E}$ (see assumption A.2 in subsection 7.2.2) between corresponding populations. Only the activity in u^{pr} above the threshold λ^{pr} is propagated to u^{pa}. This is ensured by the threshold function $f_{\lambda^{pr}}[\cdot]$.

Resting level adaptation in u^{pa}

The variation of the resting level h^{pa} of field u^{pa} is controlled by the equation:

$$\begin{aligned} \tau_{h^{pa}} \frac{\partial h^{pa}(x^{pa},t)}{\partial t} = \Big(1 - f_0\big[\, u^{pa}(x^{pa},t)\big]\Big)\Big(h^{pa}_{bas} - h^{pa}(x^{pa},t)\Big) + \\ + f_0\big[\, u^{pa}(x^{pa},t)\big]\Big(H^{pa}_{dec} - h^{pa}(x^{pa},t)\Big)\,, \end{aligned} \tag{7.3}$$

where the baseline value of the resting state, h^{pa}_{bas}, defines the operation mode (learning or recall) of the model. For field locations with $u^{pa} > 0$ the value $h^{pa}(x^{pa},t)$ decays from the baseline h^{pa}_{bas} to H^{pa}_{dec}. Reversely, for field locations with $u^{pa} \leq 0$ the function h^{pa} converges to h^{pa}_{bas}. The switching between learning and recall is realized by setting

$$h^{pa}_{bas} = \begin{cases} H^{pa}_{high}, & \text{if mode} = \text{recall} \\ H^{pa}_{low}, & \text{if mode} = \text{learning} \end{cases}, \tag{7.4}$$

respectively. It is assumed that an utterance or sentence like for instance "Let us see if you have learned" by the tutor triggers the switch. The adaptation rate of the localized decay of h^{pa} for neurons with a supra-threshold activity is controlled by the time scale parameter $\tau_{h^{pa}}$. This time constant is updated to a higher value every time the tutor gives negative feedback:

$$\tau_{h^{pa}} = \begin{cases} T_{n+1}, & \text{if (feedback} = \text{negative)} \\ \tau_{h^{pa}}, & \text{otherwise} \end{cases}. \tag{7.5}$$

The value n defines the number of times that negative feedback is given. The values T_n are hand-coded such that $T_n < T_{n+1}$. The effect is slowing down the memory decay in the past event field u^{pa}.

Input to field u^{pr}

For the field u^{pr}, the input s^{pr} is given by the equation:

$$s^{pr}\big(x^{pr},t\big) = s^{SL}\big(x^{pr},t\big) \tag{7.6}$$

where

$$\begin{aligned} s^{SL}\big(x^{pr},t\big) = C^{pr}v^{pr}\big(x^{pr},t\big) + \\ + \int f_0\Big[\ u^{pa}\big(x^{pa},t\big)\Big] \underset{pa\to pr}{I}\big(x^{pa},x^{pr}\big)dx^{pa} + \\ + C^{rec}\int f_0\Big[\ u^{pa}\big(x^{pa},t\big)\Big] \underset{pa\to pr}{a}\big(x^{pa},x^{pr},t\big)dx^{pa}\ . \end{aligned} \tag{7.7}$$

This equation described three input contributions:

1. The vision system contributes, v^{pr} with a strength parameter C^{pr};

2. The projection of activity from the past field, u^{pa}, through inhibitory connections, $\underset{pa\to pr}{I}$, to corresponding populations (see assumption A.1 in subsection 7.2.2);

3. The "recall" contribution given by the propagation of activity u^{pa}, through the adaptive connections, $\underset{pa\to pr}{a}$, established during the learning process.

Synaptic adaptation rule

The set of synaptic weights, $\underset{pa\to pr}{a}$, connects active neurons x^{pa} to x^{pr}. These connections are established during the "Observational Learning" period when a sequence of field activations occurs during a sequence demonstration. The weight adaptation occurs between sufficiently active neurons, that have activity above the learning threshold $u^{pa} > \lambda_{pa}$ and $u^{pr} > \lambda_{pr}$, and is defined as a variation of the *Delta Rule* (4.30):

$$\tau_a \frac{\partial \underset{pa\to pr}{a}\left(x^{pa}, x^{pr}, t\right)}{\partial t} = f_{\lambda^{pr}}\Big[\ u^{pr}\left(x^{pr}, t\right)\Big] f_{\lambda^{pa}}\Big[\ u^{pa}\left(x^{pa}, t\right)\Big] \times \\ \times \left[e^{pr}\left(x^{pr}, t\right) - \eta \underset{pa\to pr}{a}\left(x^{pa}, x^{pr}, t\right)\right] , \tag{7.8}$$

where the error term, $e^{pr}\left(x^{pr}, t\right)$ is given by,

$$e^{pr}\left(x^{pr}, t\right) = g_{\lambda^{pr}}\big[\ u^{pr}\left(x^{pr}, t\right)\big] - \int f_{\lambda^{pa}}\big[\ u^{pa}\left(x^{pa}, t\right)\big] \underset{pa\to pr}{a}\left(x^{pa}, x^{pr}, t\right) dx^{pa} \ . \tag{7.9}$$

Equation 7.8 reflects the Hebbian principle stating that when a neuron continually participates in the activation of another neuron, the synaptic connection between them is strengthened. Parameter τ_a defines the time scale of the learning.

The term e^{pr} corresponds to the difference between activity above the threshold $u^{pr} > \lambda^{pr}$ and the total amount of input given through the connections $\underset{pa\to pr}{a}$. The activity $u^{pr} > \lambda^{pr}$ is the target of the *Delta Rule* since it reflects the excitation from the vision system, v^{pr} which in turn reflects the target of the learning.

The decaying term in (7.8), with strength parameter η is added to ensure that the learning dynamics avoids local minima (Krogh and Hertz, 1992). For instance, after the first training period, the model is basically functioning as a simple chaining mechanism, where a single population in u^{pa} is capable of fully activating a population in u^{pr}. If a prediction error occurs it will lead to an new "Observational learning" period with an increased time window for associative learning. In other words, more past sub-populations in u^{pa} will be above the threshold λ^{pa}. However, the error term, e^{pr}, remains equal to zero, due to the connections established in the previous period. The decaying term which forces the decrease of previously established weights allows the redistribution of weights to include also the synaptic connections whose pre-synaptic neurons in u^{pa} have not been above the learning threshold before.

7.3 Short-Term Memory Layer (STM) - The rehearsal mechanism

While the mechanism described so far is fully capable of learning multiple task sequences through social interactions it has a major disadvantage. The synaptic adaptation rule described requires multiple demonstrations of the task sequence in order to robustly set the correct connection weights $\underset{pa \to pr}{a}$. In many neural network applications this does not come as a problem, since the whole training process is performed off-line. However, in LfD robotic applications like the one described here, it is normally a human being that supplies the examples in an on-line process. Learning a sequential task on-line thus represents a major challenge since a user cannot be expected to repeat the same demonstration more than a small amount of times (Ehrenmann et al., 2002).

It is known that human beings have the faculty of learning sequences and even extracting hierarchical dependencies very efficiently (Hard et al., 2006a). Neurological evidences suggest the existence of a fast but transient encoding memory mechanism capable of memorizing linear sequences with only few demonstrations. These memories seem to be later encoded in more permanent structures through some internal learning process (Preston and Eichenbaum, 2013). The sequence learning model is inspired by the idea of two complementary learning systems (McClelland et al., 1995).

An additional layer, the Short-Term Memory Layer (STM) works as a fast encoding mechanism capable of acquiring a sequence in a single demonstration. The implementation relies on the cognitive model of short-term memory proposed in (Ferreira et al., 2014). It follows a similar approach to the SL Layer in terms of representations since sequence elements are represented as populations in a DNF. It uses the multi-bump kernel described in Section 3.3, that ensures the stable coexistence of multiple active populations. The model is coupled with a control dynamics for the resting level $h^{stm}\big(x^{stm}, t\big)$ that makes the activity of each active population rise continuously over time.

The operation of the model as a short-term memory can be described in the following way: Each sub-goal observed by the vision system is translated into a localized

Gaussian input pattern $s^{stm}\big(x^{stm}, t\big)$ in the Short-Term Memory Layer (STM) layer. This Gaussian pattern triggers a localized above-zero activation that rises continuously due to the dynamics controlling the resting level. Given that the rising of the activation is linear in time, the height of the corresponding peak codes the elapsed time since the visual event. The complete demonstration is thus encoded as a field activation profile formed by multiple separate bell-shaped peaks with different weights, with the tallest one corresponding to the first observed sub-goal, the second tallest one corresponding to the second sub-goal and so on (Figure 7.10). This self-stabilized activation pattern works as short-term memory of the sequence.

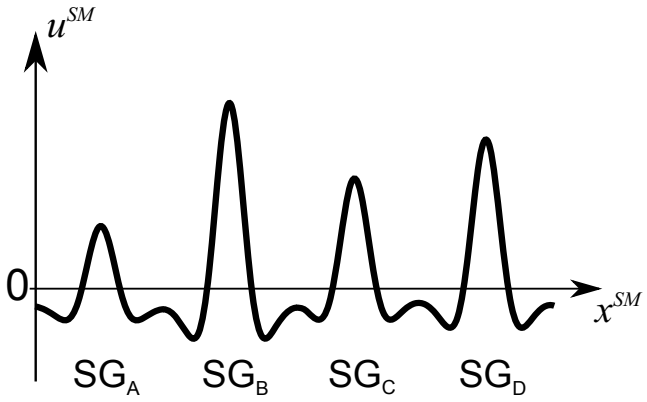


Figure 7.10: **Example of STM pattern.** The field dynamics establishes an activation gradient in response to a sequential input. The highest peak corresponds to the first observed sub-goal, the second highest corresponds to the second observed sub-goal and so on.

Because it is capable of encoding a sequence in a single demonstration, the model of Ferreira et al. (2014) provides a useful implementation of a short-term memory for the model described here. During the "Observational Learning" stage, following the demonstration from the tutor, an "Internal Rehearsal" period is added during which the memorized sequence is used as input to the SL Layer, substituting the demonstration by the tutor. In the presence of multiple input peaks, the lateral inhibition-type interaction in u^{pr} granted by the kernel implements a competition dynamics (similar to the competitive cuing mechanism in Houghton, 1990). The strongest input (corresponding to the highest peak in u^{stm} Figure 7.10) will be the first to form an above-zero activation and due to the lateral inhibition will prevent the others from rising. This newly

formed peak reaches the threshold λ^{pr} and propagates its activation to u^{pa} through the excitatory connections $\underset{pr \to pa}{E}$. When u^{pr} is inhibited by its u^{pa} counterpart, the second strongest input from the STM Layer will win the competition and so on until all sub-goals representations have been activated. Thus, the successive activations follow the serial order through which the sub-goals were demonstrated by the tutor. When for a period of time no supra-threshold activity emerges in u^{pr}, meaning that all sub-goals have been rehearsed, the fields u^{pa} and u^{pr} are reset to resting level. The process is then repeated. The number of sequences rehearsals, N^{reh}, is set heuristically.

It is important to notice here that using the activation gradient as a direct input to u^{pa} would not solve the sequential learning problem. Due to the lack of strong lateral competition in u^{pa}, the multi-peak input from u^{stm} would produce peaks in all sub-goal populations almost simultaneously.

7.3.1 Mathematical formulation of the rehearsal mechanism

Input to field u^{pr}

The inclusion of the rehearsal mechanism implies a change in the equation defining the input s^{pr}, that must now count with the excitation from the STM Layer. (7.6) is now written as

$$s^{pr}\left(x^{pr}, t\right) = s^{SL}\left(x^{pr}, t\right) + s^{SM}\left(x^{pr}, t\right) \tag{7.10}$$

where s^{SL} is given by (7.7) and s^{SM} is given by

$$s^{SM}\left(x^{pr}, t\right) = \left(b^{lea} C^{lea} + C^{rec}\right)\, u^{stm}\left(x^{stm}, t\right)\ . \tag{7.11}$$

In the above equation, the excitation from the STM Layer, u^{stm}, is added with hand-coded gain parameters $C^{rec} > 0$ (for active recall) and $C^{lea} > 0$ (added to C^{rec} during the learning period). The parameter C^{lea} is gated by a flag b^{lea} so that it is only active

when the model is learning through "Internal Rehearsal"'. In this case, the stronger input compensates the lack of external visual feedback so that the input from u^{stm} alone may drive activation in u^{pr} to values above the learning threshold λ^{pr}. Thus, b^{lea} is given by,

$$b^{lea} = \begin{cases} 1, & \text{if state} = \text{Internal Rehearsal} \\ 0, & \text{otherwise} \end{cases} . \tag{7.12}$$

The "Internal Rehearsal" stage immediately follows the "Observational Learning" stage. The tutor finishes each demonstration with an utterance such as "I have finished", which sets $b^{lea} = 1$.

Input to field u^{stm}

The STM layer receives excitation only from the vision system

$$s^{stm}\big(x^{stm}, t\big) = C^{stm} v^{stm}\big(x^{stm}, t\big) \ , \tag{7.13}$$

where the gain parameter C^{stm} regulates the input strength.

Resting level adaptation in u^{stm}

The dynamics controlling the resting level h^{stm} establishes, in a single demonstration, the activation gradient in the STM layer which encodes serial order (Ferreira et al., 2014). The dynamics of h^{stm} is governed by the following equation:

$$\begin{aligned} \tau_{h^{stm}} \frac{\partial h^{stm}\big(x^{stm}, t\big)}{\partial t} = \Big(1 - f_0\Big[u^{stm}(x^{stm}, t)\Big]\Big)\Big(- h^{stm}\big(x^{stm}, t\big) + H^{stm}_{ini}\Big) + \\ f_0\Big[\ u^{stm}\big(x^{stm}, t\big)\Big] \ , \end{aligned} \tag{7.14}$$

where H^{stm}_{ini} is the initial value of h^{stm} and $\tau_{h^{stm}}$ controls the growth rate of h^{stm}. When

$u^{stm} < 0$ the resting level h^{stm} simply converges to the initial value H^{stm}_{ini}. If $u^{stm} > 0$, the derivative is equal to one and h^{stm} grows linearly.

7.4 Extending the model for chunking sub-goals into a sub-task representation

This section describes how the model for learning of sequential tasks can be extended to address the chunking of sub-goals into higher level sub-task representations. The sub-goals of a sub-task must be completed (not necessarily by a fixed order) before a subsequent sub-task can be addressed.

The development of higher-level sub-task representations is in line with a hierarchical learning approach in cognitive neuroscience (Fuster, 2006; Botvinick, 2008). According to the hierarchical view, distinct areas code different levels of abstraction in sequential actions, with basic movements forming at the lower the "lower levels" and more abstract representations of actions and goals in the "higher" ones. It is believed that this hierarchical representation favours temporal abstraction of task plans, since high-level representations remain active during longer periods of time.

The development of these new representations is achieved by adding two new layers to the model (Figure 7.11): The Task Monitoring Layer (TM) that controls the occurrence of prediction errors and the High-level Cognitive Memory Layer (HCM) where the sub-task representations are actually formed.

7.4.1 The Task Monitoring Layer (TM)

The TM Layer (Figure 7.4) serves as a supervising mechanism that memorizes the occurrence of prediction errors and controls the formation of sub-task representations. It is formalized by a DNF which gets excitatory input $s^{tm}\big(x^{tm}, t\big)$ solely from u^{pr}. Its resting level, h^{tm}, varies between two values, H^{tm}_{low} and H^{tm}_{high}, with $H^{tm}_{low} < H^{tm}_{high} < 0$, and is controlled by the verbal feedback from the tutor.

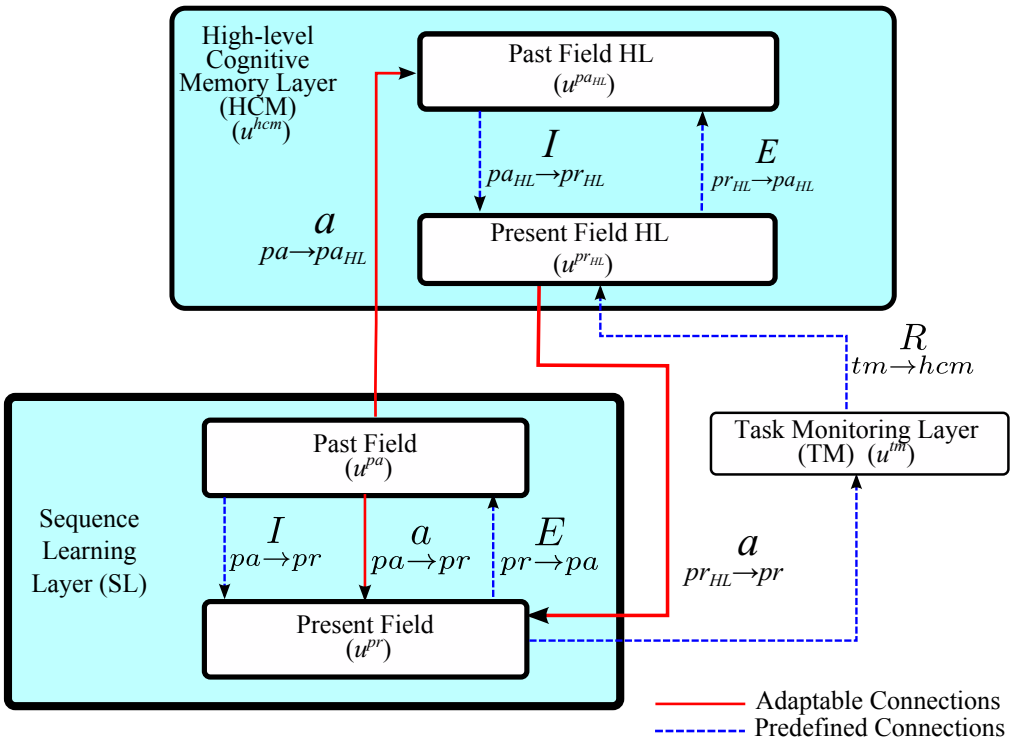


Figure 7.11: **Diagram of the extended model with the Task Monitoring Layer (TM) and the HCM Layers.** The inter-field connections that are adapted during the learning process are shown as solid lines.

It is assumed that prediction errors are the events that guide the segmentation of the task into the chunks that constitute the sub-tasks. These chunks must be completed before a subsequent sub-task can be addressed. For instance, in the example of the assembly task depicted in Figure 7.2, placing the top platform on the four columns terminates the sub-task or chunk "top part of the vehicle". Recalling the platform before all four columns are in place would constitute a prediction error that may guide the emergence of an explicit representation of that chunk in the HCM layer.

The TM Layer is a copy of u^{pr} representing all possible sub-goals. In the absence of errors, $h^{tm} = H^{tm}_{low}$ and the input s^{tm} from u^{pr} only pre-shapes the field. However, during the "Active Recall" period if a prediction error occurs and negative feedback is given by the tutor, the resting level h^{tm} is raised to a higher value H^{tm}_{high} (see subsection 7.4.3) and the localized input from u^{pr} will trigger a supra-threshold activation peak that serves as memory of the wrong prediction.

After the correction of the error, as described in the last section the human tutor provides positive feedback for an activation in u^{pr} which matches the location of the existing activation peak in TM. The comparison between positions of field activations in u^{pr} and u^{tm} mediated by positive feedback will cause the lowering of the resting level h^{tm} back to its initial level, H^{tm}_{low}. As a result, the representation of the prediction error in TM will disappear.

7.4.2 High-level Cognitive Memory Layer (HCM)

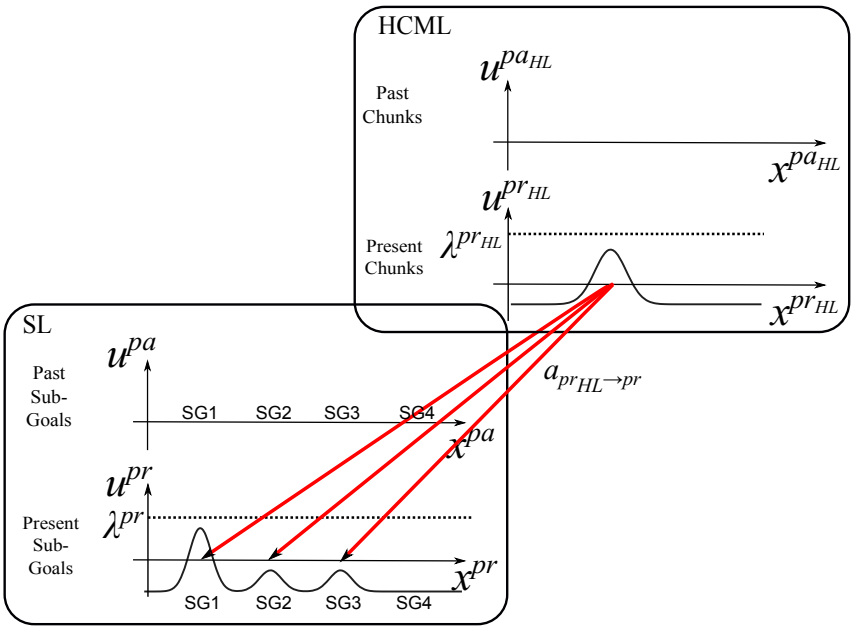
The HCM Layer contains representations of sub-tasks. It follows a double representation scheme similar to the one of the SL Layer. Past and present sub-tasks are represented in two separate fields, $u^{pa_{HL}}\left(x^{pa_{HL}}, t\right)$ and $u^{pr_{HL}}\left(x^{pr_{HL}}, t\right)$, respectively. The fields are interconnected through two sets of fixed connections, one excitatory, $\underset{pr_{HL}\rightarrow pa_{HL}}{E}\left(x^{pr_{HL}}, x^{pa_{HL}}, t\right)$ and the other inhibitory $\underset{pa_{HL}\rightarrow pr_{HL}}{I}\left(x^{pa_{HL}}, x^{pr_{HL}}, t\right)$.

In order for a population in the HCM layer to represent a whole group of sub-goals, two types of associations must be established (Figure 7.12):

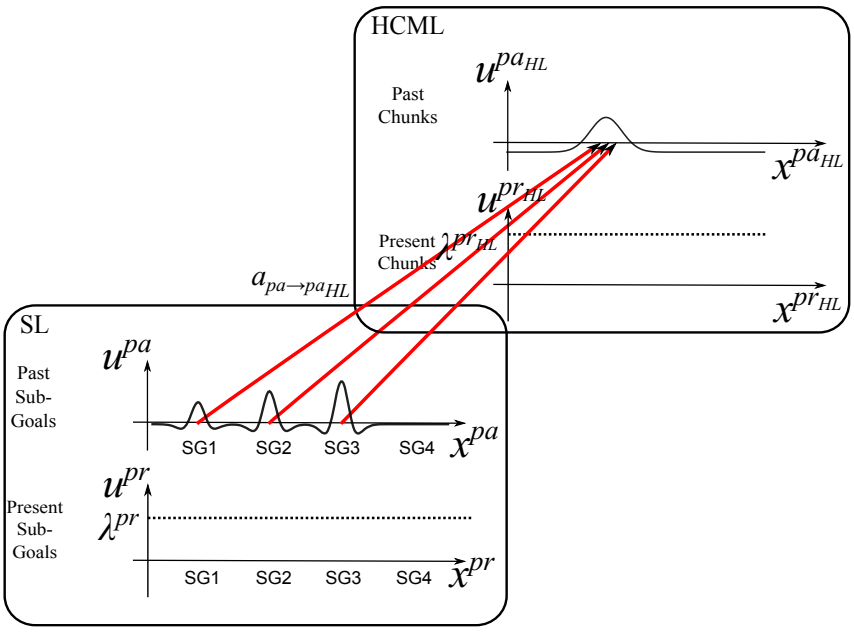
1. An active population representing a sub-task in $u^{pr_{HL}}$ must provide excitation to all populations in u^{pr} representing the sub-goals belonging to the sub-task. This is provided through a set of adaptable excitatory connections $\underset{pr_{HL}\rightarrow pr}{a}\left(x^{pr_{HL}}, x^{pr}, t\right)$ linking $u^{pr_{HL}}$ to u^{pr} (Figure 7.12a).

2. When all the sub-goals forming a sub-task are achieved and their populations in u^{pa} are active, they must provide excitation to the population in $u^{pa_{HL}}$ representing the sub-task. The total excitation must be strong enough to drive a supra-threshold peak representing the achievement of the sub-task. This association is implemented as a set of adaptable excitatory connections $\underset{pa\rightarrow pa_{HL}}{a}\left(x^{pa}, x^{pa_{HL}}, t\right)$ linking u^{pa} to $u^{pa_{HL}}$ (Figure 7.12b).

To establish the two sets of connections, $\underset{pr_{HL}\rightarrow pr}{a}$ and $\underset{pa\rightarrow pa_{HL}}{a}$, a two stage adaptation process is proposed.



(a)



(b)

Figure 7.12: **Associations between the SL and HCM Layers.** (a) depicts the associations established between the sub-task population in $u^{pr_{HL}}$ and the sub-goal populations that constitute the sub-task in u^{pr}. (b) illustrates the associations established between the populations in u^{pa} representing a completed set of sub-goals and the chunk population in $u^{pa_{HL}}$.

The first stage begins when during the "Active Recall" period a prediction error occurs and negative feedback is given by the tutor (instant t_{st1} Figure 7.13a). The error triggers an above-zero activation in the TM Layer (see Section 7.4.1). Through a set of random connections $\underset{tm \to pr_{HL}}{R}\left(x^{tm}, x^{pr_{HL}}\right)$, the peak in TM excites $u^{pr_{HL}}$. Due to the strong recurrent interactions of lateral inhibitory-type in this field, a localized peak evolves at a random position. Its strength is below $\lambda^{pr_{HL}}$, the threshold for propagating excitation to the past field $u^{pa_{HL}}$. During the subsequent "Observational Learning" period, associations $\underset{pa \to pa_{HL}}{a}$ will be established between the representation in $u^{pr_{HL}}$ and the activations above the learning threshold λ^{pr} emerging sequentially in u^{pr}. The new population in $u^{pr_{HL}}$ will, from this point on represent the chunk of sub-goals forming a sub-task. The population representation which has been initially triggered spontaneously by the prediction error, now represents a sub-task.

The second learning stage begins when the once wrongly predicted sub-goal is now predicted in its correct sequential position and positive feedback is issued by the tutor (instant t_{st2} in Figure 7.13b). When this happens, the threshold $\lambda^{pr_{HL}}$ is decreased back to its original value and as a consequence of the existing excitatory connections to $u^{pa_{HL}}$, a peak evolves in this field representing the completion of the particular sub-task. Like in the SL Layer, the population activity reaches a learning threshold $\lambda^{pa_{HL}}$ and then decays by a local adaptation of the resting level $h^{pr_{HL}}$. The synaptic connections $\underset{pa \to pa_{HL}}{a}$ are then adapted to connect currently active populations (activity above zero) in u^{pa} and the sub-task population of $u^{pa_{HL}}$ during the time when its activity is above the learning threshold $\lambda^{pa_{HL}}$. The Hebbian rule ensures that the weight distribution is balanced, meaning that after the weights are set, only the summed input of all sub-goal populations belonging to the specific sub-task will be strong enough to drive the new sub-task representation in $u^{pa_{HL}}$ above threshold.

The learning mechanism described in this section focuses on the formation of sub-task representations in the HCM Layer. The formation of a sequence representation on this higher level is not considered. Like on the lower level of sub-goals, this should happen through the learning of excitatory connections $\underset{pa_{HL} \to pr_{HL}}{a}$ linking $u^{pa_{HL}}$ to $u^{pr_{HL}}$ (see

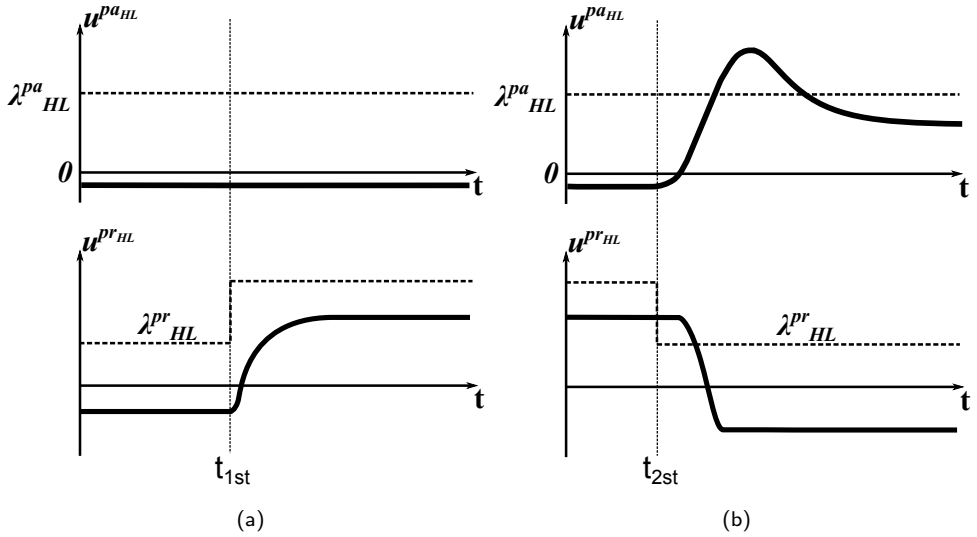


Figure 7.13: **Time course of the activation in the HCM Layer**. a) First stage of the chunking process: When the error peak appears in the TM Layer, it propagates its activity to $u^{pr_{HL}}$ (time t_{1st}). As a result, a localized peak evolves at a random position. At the same time, the threshold $\lambda^{pr_{HL}}$ is raised preventing propagation to $u^{pa_{HL}}$. b) Second stage of the chunking process: When the error is corrected, the excitation caused by the TM Layer disappears. The threshold $\lambda^{pr_{HL}}$ is lowered back to its initial value (time t_{2st}) causing the emergence of a peak in $u^{pa_{HL}}$. Connections between active populations in u^{pa} and $u^{pa_{HL}}$ are formed during the period corresponding to $u^{pa_{HL}} > \lambda^{pa_{HL}}$.

the discussion, Subsection 8.3). This open issue is left for future work.

7.4.3 Mathematical formulation of the chunking mechanism

Sequence Learning Layer (SL)

Input to field u^{pr}

The mechanism for developing sub-task representations requires the re-writing of the equation defining the input to field u^{pr} so it incorporates the excitation from $u^{pr_{HL}}$. The new input function is thus given by

$$s^{pr}\big(x^{pr},t\big) = s^{SLL}\big(x^{pr},t\big) + s^{REA}\big(x^{pr},t\big) + s^{HL}\big(x^{pr},t\big) \tag{7.15}$$

where s^{SLL} and s^{REA} are defined by equations (7.7) and (7.11) respectively and s^{HL} is given by

$$s^{HL}(x^{pr},t) = \int f_0\Big[\, u^{pr_{HL}}(x^{pr_{HL}},t)\Big] \underset{pr_{HL}\to pr}{a}(x^{pr_{HL}},x^{pr},t)dx^{pr_{HL}} \ . \tag{7.16}$$

The new term describes the excitation provided by the present field of the HCM layer, $u^{pr_{HL}}$, to the present field of SL, u^{pr}, through the connections $\underset{pr_{HL}\to pr}{a}$ established during the learning process.

Task Monitoring Layer (TM)

Input to field u^{tm}

The TM layer receives an excitatory input solely from field u^{pr}. The input term, $s^{tm}(x^{tm},t)$, can then be described as

$$s^{tm}(x^{tm},t) = C^{tm}\Big[\, u^{pr}(x^{pr},t)\Big]^{+} \ , \tag{7.17}$$

where the gain parameter C^{tm} regulates the strength of the contribution and $[u]_0^+$ represents a ramp function with threshold zero.

Resting level adaptation in u^{tm}

The resting level h^{tm} of field u^{tm} is controlled by the verbal feedback from the human. Thus, h^{tm} is given by:

$$h^{tm}(t) = \begin{cases} H^{tm}_{high}, & \text{if feedack} = \text{negative} \\ H^{tm}_{low}, & \big(\text{if feedack} = \text{positive}\big) \wedge \big(\underset{x}{\operatorname{argmin}}(\,u^{pr}) = \underset{x}{\operatorname{argmin}}(\,u^{tm})\big) \ . \end{cases} \tag{7.18}$$

Negative feedback sets $h^{tm} = H^{tm}_{high}$ allowing the formation of activity peaks representing wrongly predicted sub-goals. Positive feedback for a prediction in u^{pr} which matches the position of the memory peak in TM, sets $h^{tm} = H^{tm}_{low}$, leading to the suppression of the supra-threshold activity.

High-level Cognitive Memory Layer (HCM)

The HCM layer is constituted by the fields $u^{pr_{HL}}$ and $u^{pa_{HL}}$ each with its own input $s^{pr_{HL}}$ and $s^{pa_{HL}}$, respectively.

Input to field $u^{pr_{HL}}$

The input $s^{pa_{HL}}$ is described by the equation

$$\begin{aligned} s^{pa_{HL}}\big(x^{pa_{HL}},t\big) = &\int f_{\lambda^{pr_{HL}}}\Big[\, u^{pr_{HL}}\big(x^{pr_{HL}},t\big)\Big] \underset{pr_{HL}\to pa_{HL}}{E}\big(x^{pr_{HL}},x^{pa_{HL}}\big)dx^{pr_{HL}} \\ &+\int f_0\Big[\, u^{pa}\big(x^{pa},t\big)\Big] \underset{pa\to pa_{HL}}{a}\big(x^{pa},x^{pa_{HL}},t\big)\ , \end{aligned} \tag{7.19}$$

where the first term corresponds to the propagation of activity above threshold ($u^{pr_{HL}} > \lambda^{pr_{HL}}$) to $u^{pa_{HL}}$ through the invariant connections $\underset{pr_{HL}\to pa_{HL}}{E}$. The second term formalizes the propagation of activity from u^{pa} through the adaptable excitatory connections $\underset{pa\to pa_{HL}}{a}$. This term ensures that a population in $u^{pr_{HL}}$ representing a sub-task becomes activated when all its sub-goals are executed (see Subsection 7.4.2).

The input $s^{pr_{HL}}$ is given by

$$\begin{aligned} s^{pr_{HL}}\big(x^{pr_{HL}},t\big) = &\int f_0\Big[\, u^{pa_{HL}}\big(x^{pa_{HL}},t\big)\Big] \underset{pa_{HL}\to pr_{HL}}{I}\big(x^{pa_{HL}},x^{pr_{HL}}\big)dx^{pa_{HL}} \\ &+\int f_0\Big[\, u^{tm}\big(x^{tm},t\big)\Big] \underset{tm\to pr_{HL}}{R}\Big(x^{tm},x^{pr_{HL}}\Big)dx^{pa_{HL}}\ , \end{aligned} \tag{7.20}$$

corresponding to the propagation of activity from $u^{pa_{HL}}$ to $u^{pr_{HL}}$ through the fixed

inhibitory connections $\underset{pa_{HL}\rightarrow pr_{HL}}{I}$ and the random input provided by field u^{tm} (see Subsection 7.4.2).

Adaptation of the learning threshold $\lambda^{pr_{HL}}$

The learning threshold $\lambda^{pr_{HL}}$ is controlled by the feedback from the "teacher". When the robot makes a prediction error and the tutor gives negative feedback, the threshold is set to its higher value $\Lambda_{high}^{pr_{HL}}$. This ensures that the activations in $u^{pr_{HL}}$ do not propagate to $u^{pa_{HL}}$. When the learning process triggered by the error finally leads to a correct prediction, the teacher gives positive feedback. This lowers the threshold back to its initial value $\lambda^{pr_{HL}} = \Lambda_{high}^{pr_{HL}}$:

$$\lambda^{pr_{HL}} = \begin{cases} \Lambda_{low}^{pr_{HL}}, & \text{if (feedback = positive)} \wedge \Big(\underset{x}{\operatorname{argmin}}(\, u^{pr}) = \underset{x}{\operatorname{argmin}}(\, u^{tm}) \Big) \\ \Lambda_{high}^{pr_{HL}}, & \text{otherwise} \end{cases} \tag{7.21}$$

Synaptic adaptation rules

The synaptic connections $\underset{pr_{HL}\rightarrow pr}{a}$, propagate activity from a sub-task representation in $u^{pr_{HL}}$, to all the sub-goals belonging to this sub-task, represented in u^{pr}. This set of connection weights has a learning rule similar to (7.8):

$$\begin{aligned} \tau_a^{pr_{HL}} \frac{\partial \underset{pr_{HL}\rightarrow pr}{a}}{\partial t} = f_0\Big[\, u^{pr_{HL}}\big(x^{pr_{HL}}, t\big)\Big] f_{\lambda^{pr}}\Big[\, u^{pr}\big(x^{pr}, t\big)\Big] \times \\ \times \Big[e^{pr}\,(x^{pr}, t) - \eta \underset{pr_{HL}\rightarrow pr}{a}\big(x^{pr_{HL}}, x^{pr}, t\big)\Big] \end{aligned}, \tag{7.22}$$

with the following error term:

$$\begin{aligned} e^{pr}\,(x^{pr}, t) = g_{\lambda^{pr}}\Big[\, u^{pr}\big(x^{pr}, t\big)\Big] - \\ - \int f_0\left[\, u^{pr_{HL}}\big(x^{pr_{HL}}, t\big)\right] \underset{pr_{HL}\rightarrow pr}{a}\big(x^{pr_{HL}}, x^{pr}, t\big) dx^{pr} \end{aligned}. \tag{7.23}$$

The connections, $\underset{pa \to pa_{HL}}{a}$, ensure that when all sub-goals of a particular chunk are achieved, that is, all its constituting populations are supra-threshold in u^{pa}, the corresponding high-level representation in $u^{pa_{HL}}$ also becomes active. The learning rule adapting the weights is similar to (7.22):

$$\begin{aligned} \tau_a^{pa_{HL}} \frac{\partial \underset{pa \to pa_{HL}}{a}}{\partial t} = f_{\lambda^{pa_{HL}}}\Big[\, u^{pa_{HL}}\big(x^{pa_{HL}}, t\big)\Big] f_0\Big[\, u^{pa}\big(x^{pa}, t\big)\Big] \times \\ \times \Big[e^{pa_{HL}}\left(x^{pa_{HL}}, t\right) - \eta \underset{pa \to pa_{HL}}{a}\big(x^{pa}, x^{pa_{HL}}, t\big)\Big] \end{aligned}, \tag{7.24}$$

with the error term $e^{pa_{HL}}$ given by,

$$\begin{aligned} e^{pa_{HL}}\big(x^{pa_{HL}}, t\big) = g_{\lambda^{pa_{HL}}}\Big[\, u^{pa_{HL}}\big(x^{pa_{HL}}, t\big)\Big] - \\ - \int f_0\left[\, u^{pa}\big(x^{pa}, t\big)\right] \underset{pa \to pa_{HL}}{a}\big(x^{pa}, x^{pa_{HL}}, t\big) dx^{pa} \end{aligned}. \tag{7.25}$$

Chapter 8

Model of learning sequential tasks - Implementation and results

The model of learning sequential tasks described in Chapter 7 was implemented in the robotic platform ARoS. The goal of the implementation was to test the model performance in a real world scenario where a robot learned a task in close interactions with a tutor. This chapter reports results on a set of experiments conducted to demonstrate the functioning of the several components of the model and its features. The chapter is divided into three sections. The first section presents results demonstrating how the model furnishes the robot with the ability to learn a multi-path sequential plan of a task. It also shows how the rehearsal mechanism avoided the need of repeated demonstrations by the user. The second section described the development of high-level sub-task representations. The last section presents a discussion of the results.

8.1 Learning of a sequential task and the rehearsal mechanism

The robotics experiments with ARoS focus on the highl-level cognitive capacity to learn the sequential order of the different sub-goals defining a certain task. It is assumed that ARoS already has the necessary perceptual and motor capacities to execute each sub-goal.

The purpose of the model for sequence learning is to endow ARoS with the ability to learn a sequential representation of the different sub-goals defining a construction task. An LfD paradigm is applied in which the robot first observes a human teacher executing the sequential task and then tries to recall it from memory. The teacher actively guides the learning process by providing verbal feedback about serial order errors that may occur at each step and provides additional demonstrations if necessary. Importantly, to learn not only a single linear sequence of task execution, the tutor demonstrates the task using different serial orders. This allows the robot to acquire general task knowledge about the logical dependence of sub-goals.

The model integrates a fast encoding mechanism that memorizes the last sequence of events of a single demonstration. This "sequence memory" is then applied to adapt the synaptic connections of the generalized sequence model through an internal rehearsal process.

8.1.1 Experimental setup

Based on previous work conducted at the Mobile and Anthropomorphic Robotics Laboratory at the University of Minho, an experimental paradigm was developed in which a human demonstrator had to teach the robot ARoS how to construct a toy vehicle from its components. The physical setup is similar to the one used in Bicho et al. (2011a). Human and robot are placed on opposite sides of a table, facing each other (Figure 8.1). Each partner has a defined workspace and is supposed to manipulate objects. If neces-

sary, human and robot are able to coordinate to transfer objects from one partner to the other.

During the "Observational Learning" phase, the human has all parts within reach to perform the task. During "Active Recall", human and robot cooperate with each other to complete the task. This is intended to show how the learned sequence can be used as shared task knowledge for joint action. To enforce cooperation, objects are distributed between both workspaces.

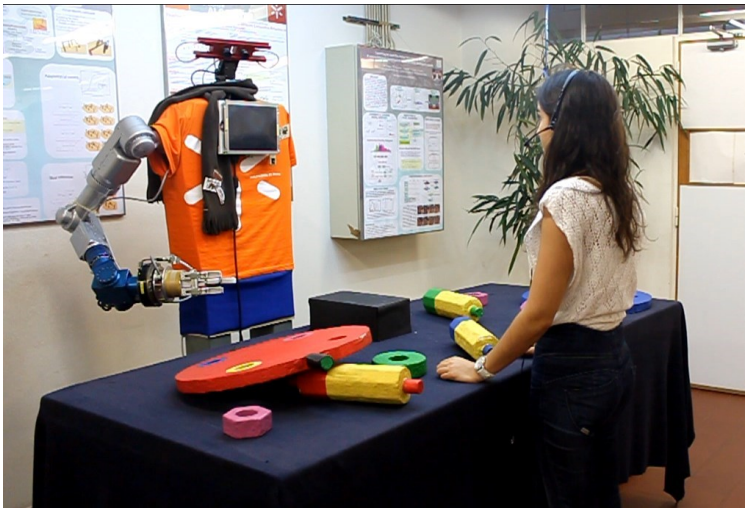


Figure 8.1: **Experimental setup.** The figure depicts the setup used in the experiments. ARoS and the human demonstrator are working on opposite sides of the table.

Three experiments are reported in this section in which different tutors demonstrated different serial orders of the construction. In the last experiment, an error occurred and was eventually corrected following additional demonstrations.

The assembly task

The robot has to learn the possible serial orders of the assembly of a toy vehicle depicted in Figure 7.2 which is composed of the following parts:

- A red base (BA) with two black pegs on opposite sides and four vertical holes;
- Two wheels, left (LW) and right (RW);

- Two nuts, left (LN) and right (RN), fixing the wheels;
- Four columns (GC, RC, BC, MC) half yellow, half other color (green, red, blue, magenta);
- A top platform (TP).

The wheels are placed first on the pegs of the base followed by the nuts that fix the wheels. The columns are inserted in the vertical holes of the base whose contours have distinctive colors that match the colors of the corresponding columns. The top platform is placed when all columns are in place. In principle the columns can be inserted in any order. The task thus allows to test the learning of flexibility in task execution as well as causal constraints on the serial order of sub-goals.

Implementation details of the experiments

The motor system already has the motor primitives required for the task, namely, reaching objects from table, reaching objects from human hand, insert objects (e.g. wheel on the base peg) and handing over objects to partner.

The vision system is responsible for recognizing the achievement of a specific sub-goal of the task. This recognition is based on the positioning of the objects relative to each other. If, for instance, a wheel is perceived near the base peg it is assumed that it is attached to the base. This simplifies the work of the vision system since no explicit recognition of the assembly state has to be performed. However, the assumption is not completely unrealistic since very often occlusions prevent the vision system from directly observing the end state of a specific assembly step. When the achievement of a specific sub-goal is recognized (e.g. GC inserted) the information supplied by the vision system is translated into Gaussian inputs v^{pa}, v^{pr} and v^{stm}, to the fields u^{pa}, u^{pr} and u^{stm}.

Human feedback is given verbally through a set of predefined sentences recognized by the speech recognition module. The sentence can be for instance "that is correct" for positive and "that is wrong" for negative feedback. When the robot vocalizes a

prediction (e.g. "next we can place the blue column") it always waits for a reply before performing any action. If the feedback is positive, the robot will engage in the decision process about the adequate motor behaviour that must be employed to achieve the recalled sub-goal. The decision process takes into account contextual information about the location of the pieces in the two workspaces of the robot and the human user. For instance, if a column is placed on the table in the robot's workspace but has to be inserted on the opposite side of the base, the robot will transfer it to the user (Bicho et al., 2011b).

Switching between the learning and demonstration modes is also achieved by verbal communication. Before starting a demonstration, the tutor says "I will show you". He/she finishes the demonstration with the phrase "I have finished", triggering the "Internal Rehearsal" of the demonstrated sequence. The robot warns the tutor before starting the execution of the task by saying "I think I am ready".

Next the three experiments are described. Model parameters are detailed in Appendix A.

8.1.2 First experiment: learning a sequence

In the first experiment, the tutor demonstrates the assembly of the top of the toy vehicle, using the following serial order:

$$\text{BA} \rightarrow \text{MC} \rightarrow \text{GC} \rightarrow \text{RC} \rightarrow \text{BC} \rightarrow \text{TP}$$

The demonstration took approximately 28 seconds [1]. Figure 8.2 depicts the resulting pattern in the STM Layer. One can clearly see that the different heights of the peaks represent the temporal order in which the sub-tasks were demonstrated.

A total of 30 rehearsal trials were made. During each one, the pattern stored in u^{stm} provides the input contribution to u^{pa} and u^{pr} that drives the rehearsal. Each rehearsal trial has a fixed duration T^{reah}, which is heuristically chosen to allow all peaks

[1] Video can be seen at `http://marl.dei.uminho.pt/public/videos/TaskLearningVideo1.html`

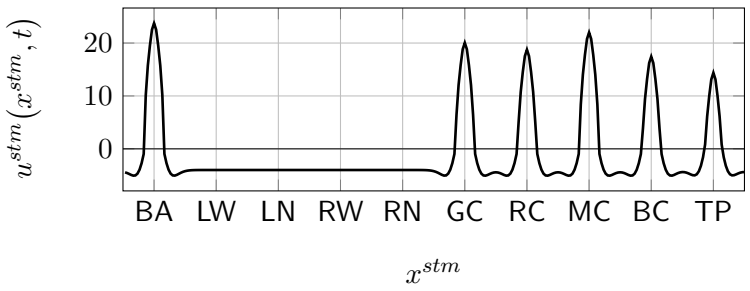


Figure 8.2: **Activation profile of the STM layer after the first demonstration.** The height of the peaks reflects the temporal order of the sub-goal demonstrations.

in u^{stm} to trigger supra-threshold activations at the corresponding site in u^{pr}. After T^{reah}, the activations in u^{pa} and u^{pr} are set back to resting level. Figure 8.3 depicts the time evolution of the maximally excited neuron of each population in u^{pa} and u^{pr}, during "Observational Learning" (Figure 8.3a) and "Internal Rehearsal" (Figure 8.3b). Despite the temporal differences (see subsection 8.1.5), the populations are activated according to the order through which the sub-goals were demonstrated. As can be clearly seen, the serial order appears to be preserved during the sequence activation from memory. Only the time course of activation differs slightly since during rehearsal it is defined solely by the dynamics of the coupled field and is thus independent of the exact timing of the inputs from the vision system.

The "Internal Rehearsal" is followed by an "Active Recall" stage. To start the task, the tutor placed the base BA (Figure 8.4), leading to the emergence of an activation in the BA population in field u^{pa}. This activation propagated through the adaptive connections $\underset{pa \to pr}{a}$ leading to a peak in u^{pr} at the location of the MC population (Figure 8.5). This peak corresponds to a prediction of which sub-goal must be achieved next. The prediction is deemed as correct, since it is consistent with the demonstration.

The activation in u^{pr} triggered a vocalization by the robot that pronounced the sentence "Now we can place the magenta column". The tutor provided positive feedback. Because the magenta column was in the robot's workspace, but had to be inserted on the tutor's side, the robot held out the magenta column for the tutor, while issuing the sentence "'Please take the magenta column". The tutor reached the column from

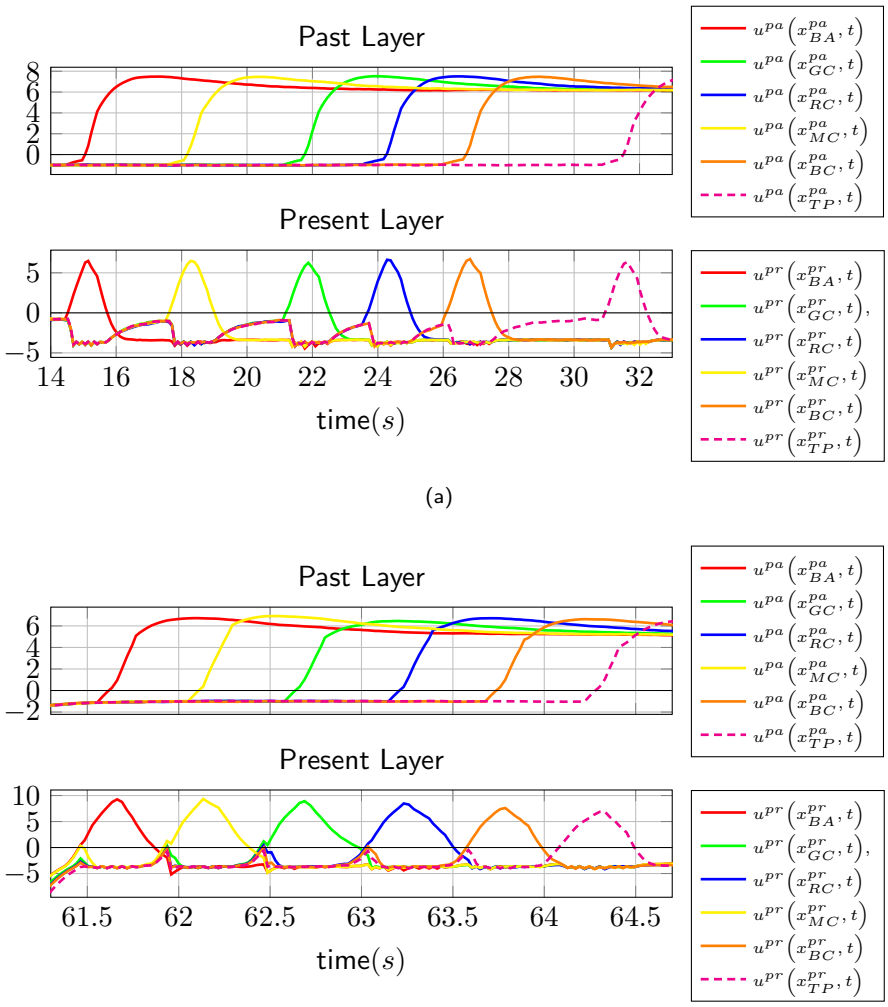


Figure 8.3: **Time course of population activity during "Observational Learning" (a) and "Internal Rehearsal" (b).**

the robot hand, and inserted it. The memory of the accomplished sub-goal in u^{pa} drove a new prediction represented by the GC population in u^{pr}. Following the positive feedback given by the tutor, ARoS raised its empty hand towards the tutor and verbalized the sentence "Please give me the green column" since the column was located in the user's workspace. The tutor then handed the column to ARoS for inserting the column on the platform. The process continued until all sub-goals were completed.



Figure 8.4: **Snapshots of the first experiment: Active Recall.** The tutor has just placed the base. The robot then recalls the magenta column (MC) as the next sub-goal.

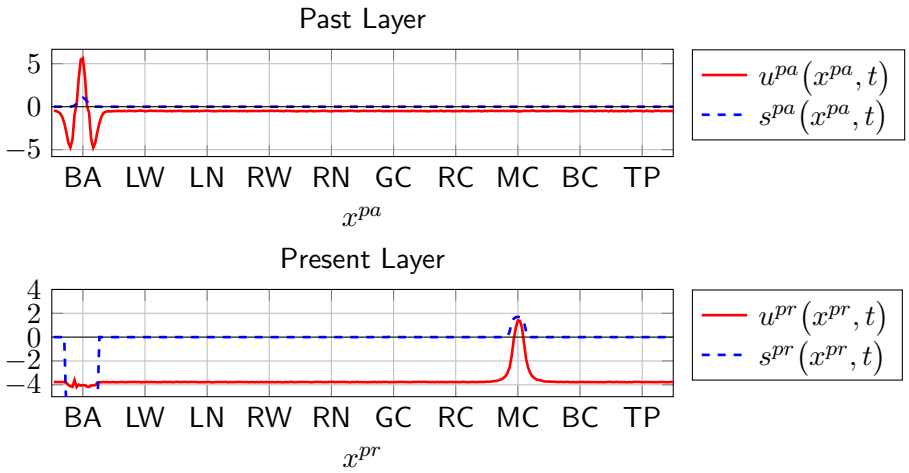


Figure 8.5: **Activation profile of the SL - Recall after the first demonstration.** When the base is inserted, the model recalls MC as the next sub-goal to be achieved.

8.1.3 Second experiment: learning an alternative sequence

In the second experiment, another tutor demonstrated the same task, but using a different order [2]. She began with BA and finished with the TP but the order of two columns RC and BC was reversed:

[2]Video is available at `http://marl.dei.uminho.pt/public/videos/TaskLearningVideo2.htm`

$$\text{BA} \rightarrow \text{MC} \rightarrow \text{GC} \rightarrow \text{BC} \rightarrow \text{RC} \rightarrow \text{TP}$$

This second demonstration took approximately 30 seconds. The significant difference from this recall attempt to the previous one, is that after the placing of GC, the input s^{pr} triggers supra-threshold activity in both RC and BC, meaning that both possible sub-goals are now encoded by the learned connection weights (Figure 8.6). The decision of which sub-goal population becomes active depends on the noise in u^{pr} introduced by the stochastic term $\zeta^{pr_{stoch}}$ in (7.1). In the experiment, the competition in u^{pr} mediated by the lateral connections in the field led to a decision favouring BC. Since BC was in the tutor's workspace, she grasped and inserted the column (Figure 8.7). After BC was placed, RC was chosen as the next sub-goal.

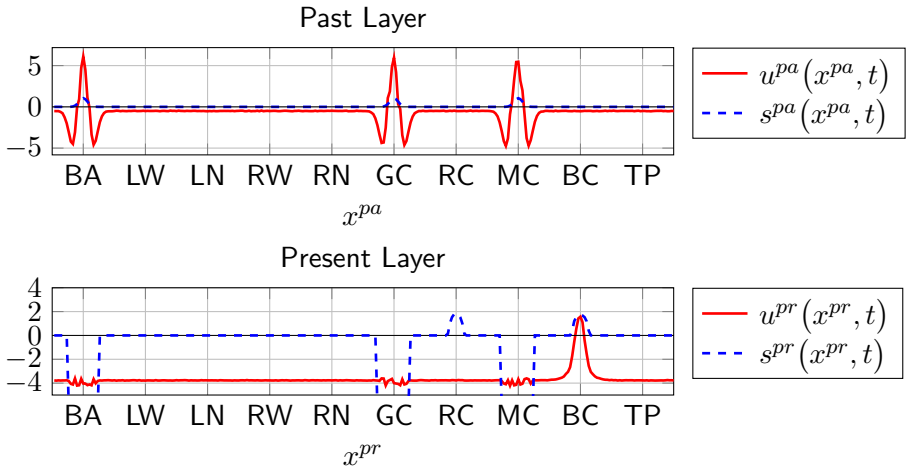


Figure 8.6: **Activation profile of the SL: recall after the second demonstration.** After insertion of GC and MC, there is localized input to both RC and BC, resulting from the connections established during the two demonstrations. In this trial, BC won the demonstration.

To test the probability of choosing each of the two possible sub-goals, the weights $\underset{pa\rightarrow pr}{a}$ were stored and an off-line test was run a posteriori. The simulated situation was the same as during the real experiments, with BA, MC and GC already in place. The decision process in u^{pr} was repeated 1000 times. The percentage of choices for BC and RC was 47.6% and 52.4% respectively. This is reflected in the balanced connections strengths that link the populations BA, GC and MC in u^{pa} to RC and BC in u^{pr}, depicted in Figure 8.8.



Figure 8.7: **Snapshot of the second experiment: Active Recall.** Both the Green and the Magenta columns (GC and MC) are inserted. The robot recalls the insertion of the blue column BC as being the next sub-goal. The user grasps and inserts it.

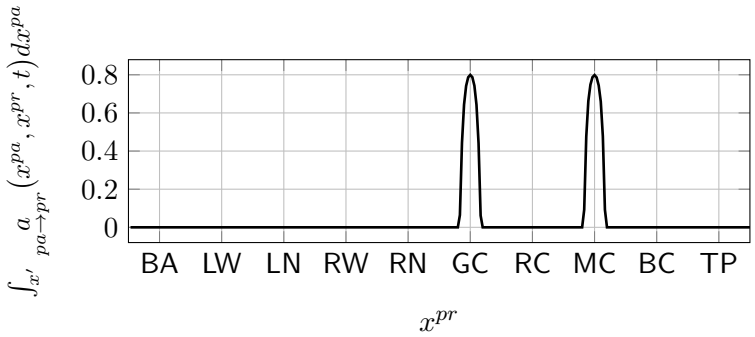


Figure 8.8: The integrals of all newly established connection weights that link active populations in u^{pa} to the two populations GC and MC in u^{pr} are compared. The total input is approximately equal for both sub-goals.

8.1.4 Third experiment: errors occurrence and correction

A third experiment was made with the purpose of demonstrating the model's handling of error events. To this end, the task was re-demonstrated by a third tutor with the following sequence:

$$\text{BA} \rightarrow \text{BC} \rightarrow \text{RC} \rightarrow \text{GC} \rightarrow \text{MC} \rightarrow \text{TP}$$

This new demonstration took approximately 34 seconds [3]. After the demonstration and "Internal Rehearsal" the tutor places the base leading to a prediction by the robot. In Figure 8.9 it is visible that the input s^{pr} excites two locations, MC and and BC, mediated by the connections established in the three preceding demonstrations. The competition in u^{pr} resulted in a peak at location MC.

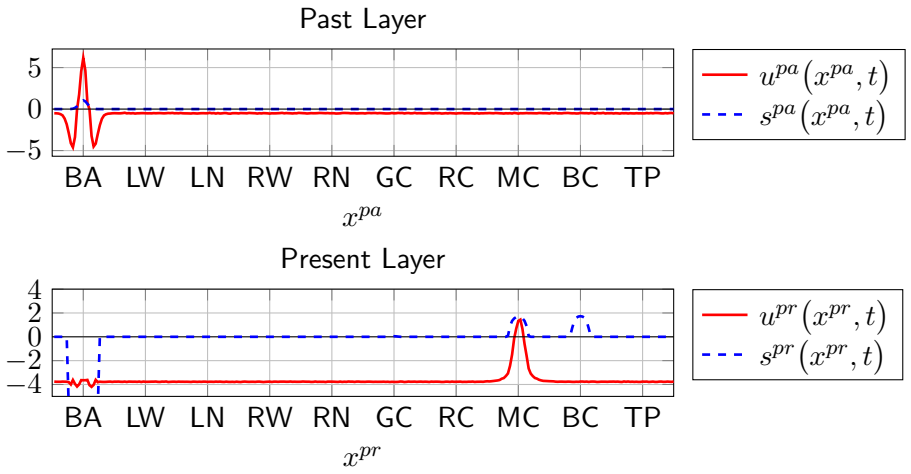


Figure 8.9: **Activation profiles of the SL Layer: recall after the third demonstration.** Following the placement of the base (BA) the input s^{pr} to field u^{pr} has two bumps at MC and BC. The decision process favours MC.

After placing MC, the input from u^{pa} triggered through the learned connections the evolution of excitation at locations GC, BC and TP. A peak is eventually formed at TP, corresponding to a wrong prediction (Figure 8.10).

Looking on the individual connection strengths (Figure 8.11) it becomes clear that the choice of TP at this stage was in fact the less likely one. However, even if GC or BC were chosen, the probability of a sequence error in subsequent stages remained high. This was verified through an a posteriori statistical test. The value of the weights $\underset{pa\to pr}{a}$ were stored and the decision process was simulated for the multiple choices at different states of the assembly work. For each possible state, the decision process in u^{pr} was repeated 1000 times for measuring the probability of each possible outcome. This allowed the building of the probability tree depicted in Figure 8.12 which shows that the total probability of an error occurrence was 89.6%.

[3] Video available at `http://marl.dei.uminho.pt/public/videos/TaskLearningVideo3.htm`

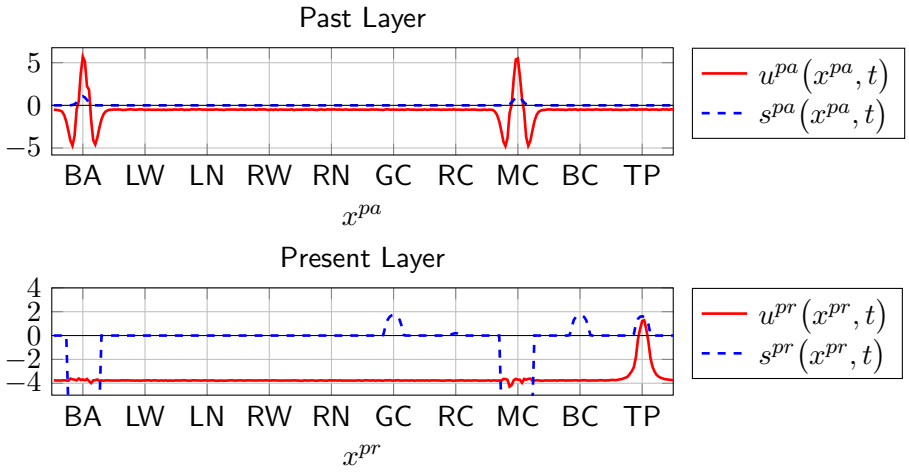


Figure 8.10: **Activation profile of SL: first error.** When both the base and the magenta column are placed the input has three bumps. In this trial, the top platform is prematurely chosen as the next sub-goal.

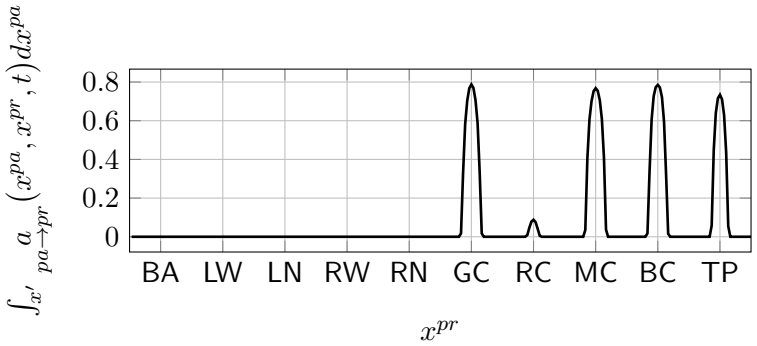


Figure 8.11: **Comparison of the integrals of learned connection weights from active populations in u^{ps} to populations GC, MC, BC and TP in u^{pr}.** The summed input to all four populations is very similar. Since MC has been already inserted this choice is inhibited.

Error correction through adaptation of the decay rate of h^{pa}

Negative feedback about a prediction (Figure 8.13) decreases the decay rate of h^{pa}, augmenting the time window for past sub-goals to become associated with a specific prediction in u^{pr}.

After a new demonstration of the same serial order, the adapted weights led to a decision to insert BC following MC. However, TP was again prematurely chosen as the next sub-goal (Figure 8.14). This second error occurred because the time window for learning still did not include enough past sub-goals to disambiguate the situation.

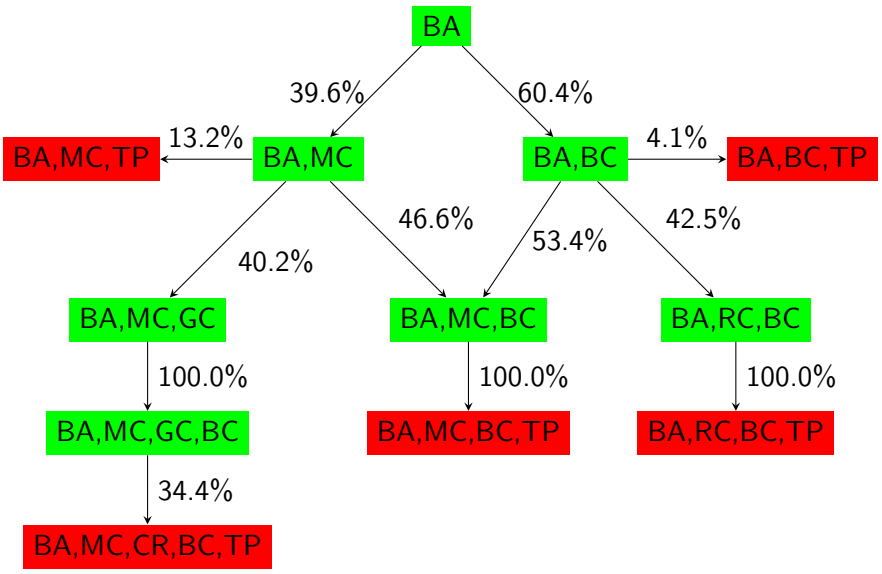


Figure 8.12: **Probability graph of the decision process in the SL Layer following the first demonstration of the third sequence.** The coloured marks represent possible states of the task and the numbers in the lines indicate the probability of a decision taking to the next state. Only the paths that lead to an error are represented. By summing the probabilities of each error, one can verify that the overall probability of an error occurrence is equal to 89.6%.



Figure 8.13: **Snapshot of the third experiment: first error.** The robot recalls the top platform after the magenta column was placed.

Figure 8.15 depicts the probability tree of the decision process for this second recall attempt, calculated from a posteriori simulation, like in the previous experiment. The total probability of an error occurrence was equal to 66.7%.

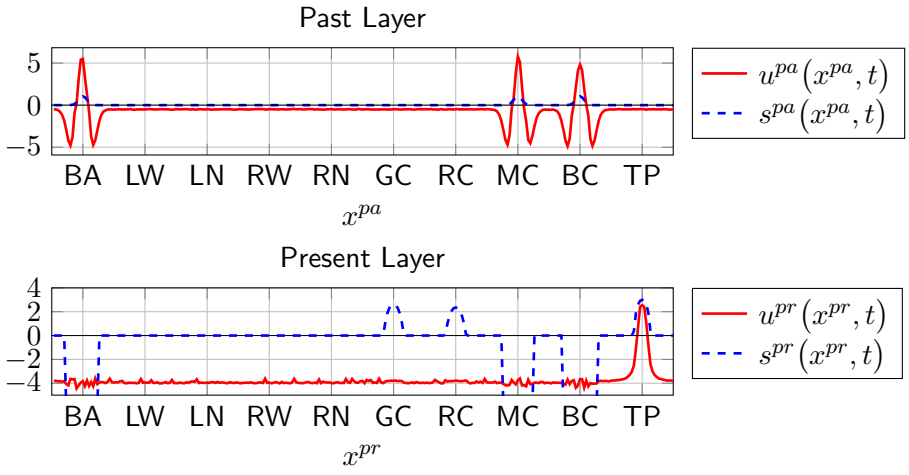


Figure 8.14: **Activation profile of the SL: second error.** With the base in place and both the magenta and blue columns inserted (MC and BC) the decision was taken again for the top platform (TP).

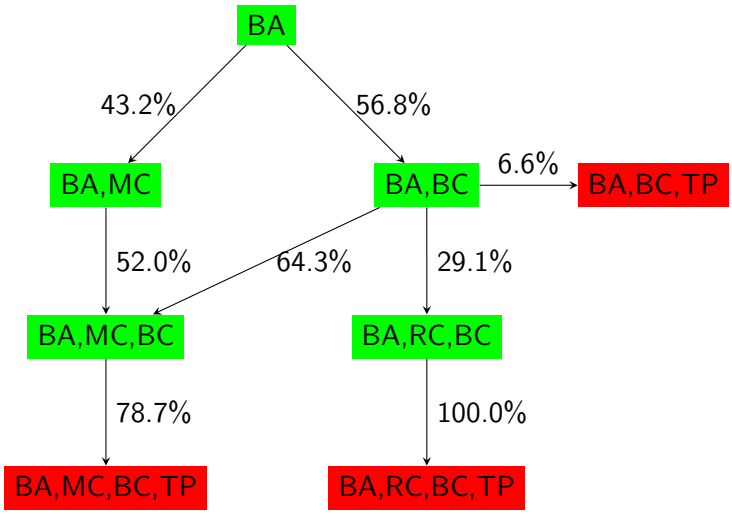


Figure 8.15: **Probability graph of the decision process in the SL Layer following the second demonstration of the third sequence.** Only the paths that lead to an error are represented. After a second demonstration with more past events affecting the robot's prediction, the error probability is still high, with a value of 66.7%.

Negative feedback by the tutor caused a further decrease in the decay rate of h_{pa}. Following a new demonstration, the model did not make erroneous predictions anymore (Figure 8.16 and 8.17). Only after all the columns were in place, TP was selected.

Like in the previous recall attempts, the weights were stored and an a posteriori test was made with each decision condition simulated 1000 times. Figure 8.18 depicts the resulting probability tree. By summing the probabilities one can verify that the error

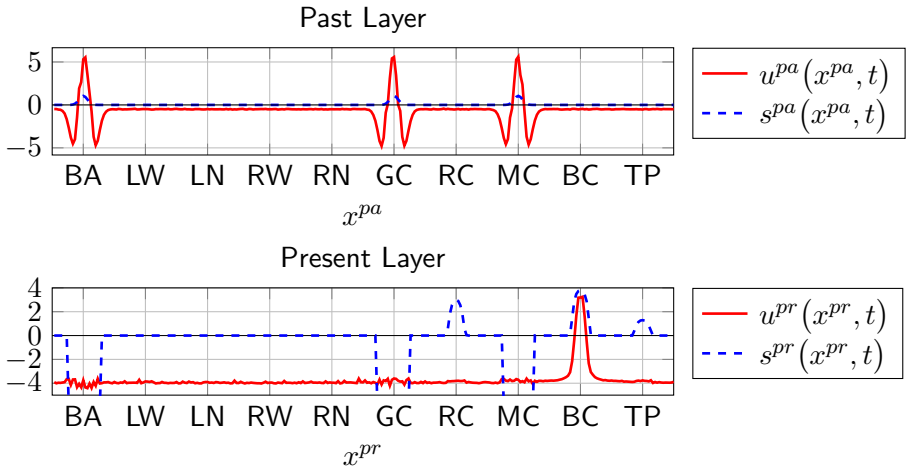


Figure 8.16: **Activation profile of the SL: correct learning.** After the third demonstration of the third sequence, the active populations (BA), GC and MC in u^{pa} support the correct prediction to select BC as the next sub-goal.

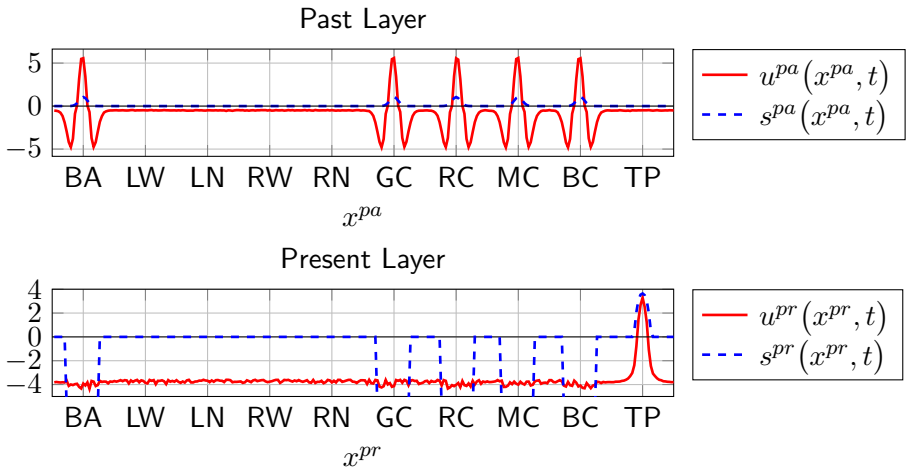


Figure 8.17: **Activation profile of SL: Correct recall of the Top Platform.** TP was only recalled after inserting all columns.

probability dropped from 89.6% and 66.7% in the first and second attempts to 1.3% in the third attempt. It is also visible that from the three demonstrated sequences of sub-goals the model generalized a task representation that favours mostly two sequential orders.

Figure 8.19 depicts the temporal evolution of u^{pa} for the columns RC, GC and MC and for the TP during the rehearsal times. Figure 8.19a corresponds to the second rehearsal period and Figure 8.19b corresponds to the third rehearsal period. One can

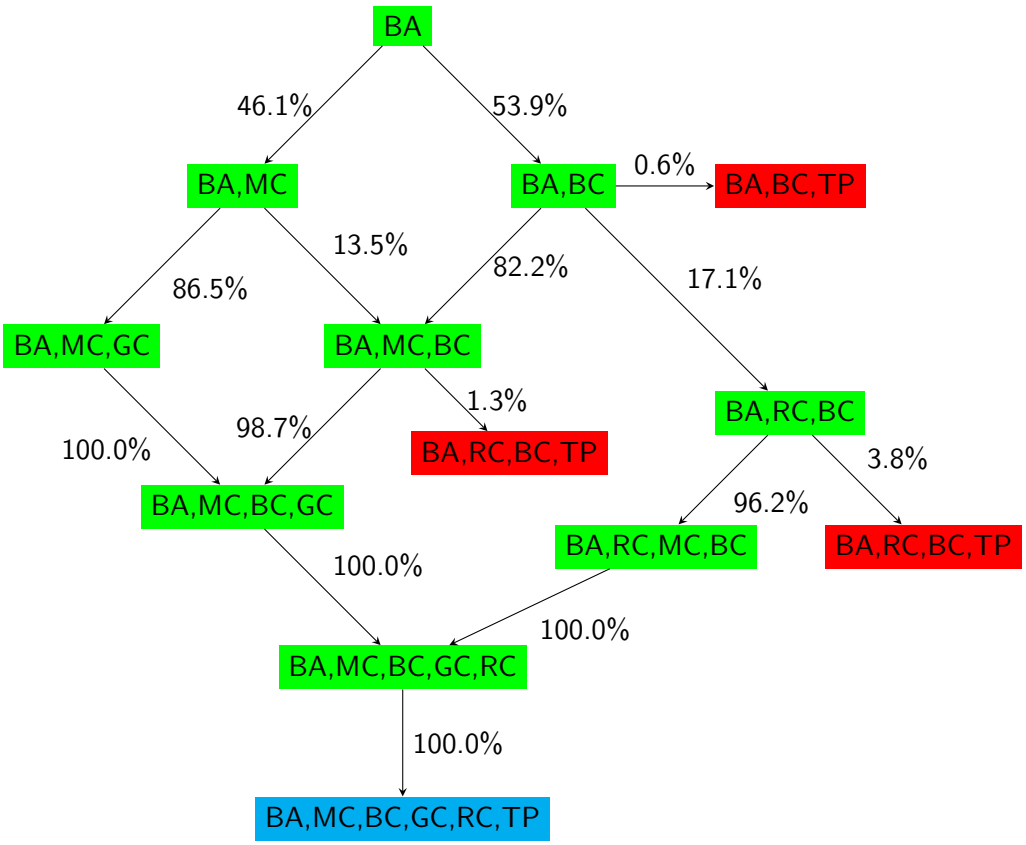
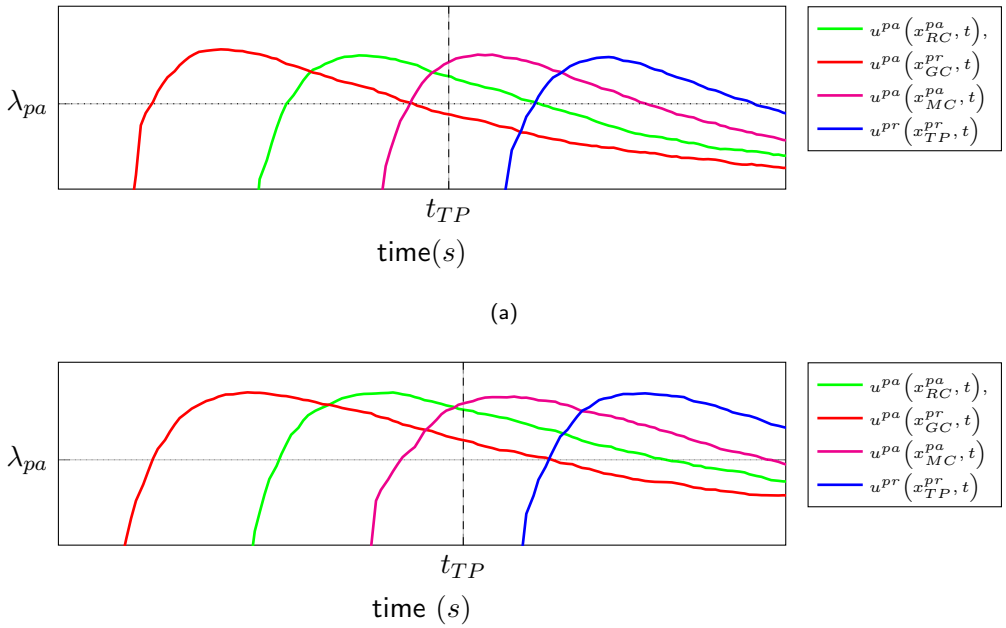


Figure 8.18: **Probability graph of the decision process in the SL Layer following the third demonstration of the third sequence.** The time window for associative learning was now large enough to represent the sequential dependencies between the four columns and the TP. As a consequence, the error probability dropped to 1.3%.

see that, during the second rehearsal, when TP was placed at time t_{TP} the activities of the populations encoding MC and GC was above while RC was below the learning threshold λ^{pa}. During the third rehearsal, however, the memory representation of all the three columns was above threshold at time t_{TP}.

8.1.5 Time differences between demonstration and internal rehearsal

One important point concerning the sequence model is related with the time intervals between the formation of consecutive activity peaks in field u^{pr}. These intervals are influenced by different factors during the stages of "Observational Learning" and "Internal Rehearsal". During "Observational Learning", the intervals depend on factors external



(a)

(b)

Figure 8.19: **Time Evolution of u^{pa} during the second (a) and third (b) rehearsal periods of the third demonstration.** (a) After the first error, the decay rate of h_{pa} decreased and when TP was inserted the activity of neurons representing GC and MC was above threshold. (b) After the second error, the decay rate of h_{pa} decreased further. When TP was inserted, the activity of neurons representing GC, MC and RC was above threshold.

to the model, namely the variability in the execution of the demonstrator and the delays introduced by the vision system in the recognition of the actions. During the "Internal Rehearsal", the variability in the inter-event times is in general much smaller and results mainly from the different strengths of competition between sub-goal alternatives in each step of the sequence evolution in u^{pr}. This is visible in Figure 8.20 which depicts, for the first three demonstrated sequences, the time intervals between pairs of consecutive events. The intervals are defined by the instants when population activity in u^{pa} crosses the zero-threshold for creating the memory peak of a particular sub-goal. The top row shows the intervals during demonstrations, while the bottom row presents the average (bar) and standard deviation (error) of each demonstration.

The much reduced variability in timing during rehearsal has an important impact in the learning process. It allows the system to systematically control the time window

for associative learning in order to systematically include increasingly more past events in the learning process.

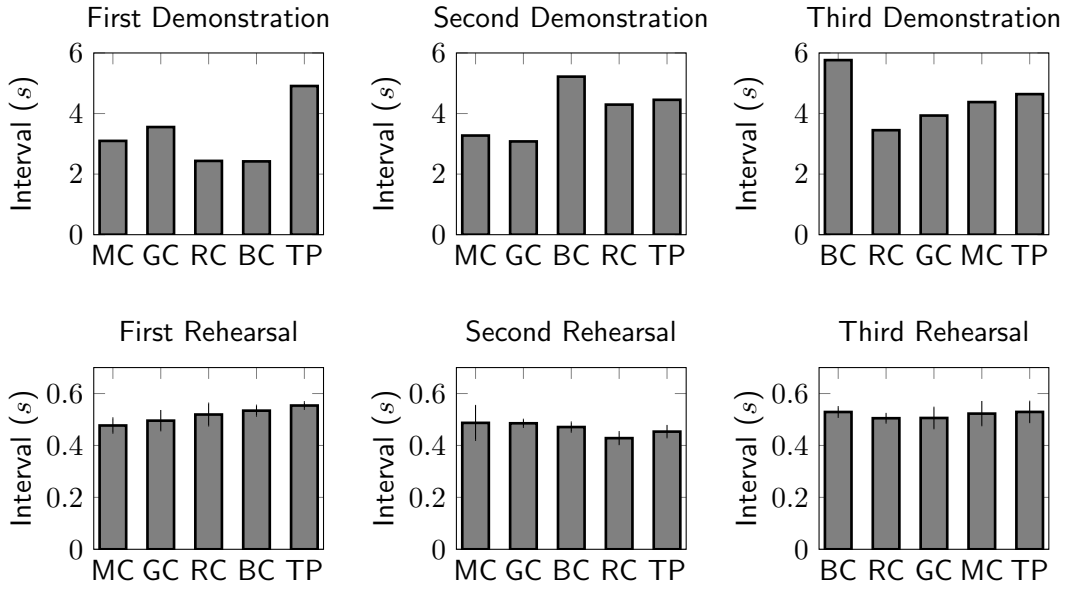


Figure 8.20: **Comparison of the time intervals between activations in the past field during observation and rehearsal.** The top row shows the intervals during observations and the bottom row depicts the average value and standard deviation during rehearsal. Note the much reduced variability of inter-event duration and the much higher execution speed during rehearsal.

8.2 Development of sub-task representations

The model for learning sequential tasks was extended to address the emergence of new populations representing sub-tasks. These are formed as chunks of related sub-goals. The purpose of this section is to report results that demonstrate how these new populations emerge and how they can affect the posterior sequence recall.

The experimental setup and implementation details are the same described in the previous section. The mechanism for the formation of sub-task representations was actuating in parallel with the sequence learning model during the third experiment when the error occurred and was corrected. A fourth experiment is also reported where the construction order of the lower part of the toy vehicle is demonstrated and a second sub-task representation is established.

8.2.1 Forming a representation in HCM Layer

As described in Chapter 7, the emergence of new populations in the HCM Layer occurres in two stages. The first establishes the connections between $u^{pr_{HL}}$ and u^{pr}. The second links u^{pa} to $u^{pa_{HL}}$. The two stages occurred during the third experiment and are described next.

First developmental stage

During the third experiment, the occurrence of the first error that triggered the adaptation of the time window for past events also started the development of a sub-task representation in the HCM Layer. Figure 8.21 depicts the time evolution of the neurons from active populations in fields u^{pr}, u^{tm} and $u^{pr_{HL}}$. The negative feedback by the tutor caused the rise of the resting level h^{tm} in the TM layer. As a result, a supra-threshold activity peak evolves at a field site that matches the location of the wrong prediction in u^{pr}. Mediated by the weak random connections $\underset{tm \to pr_{HL}}{R}$, the memory representation of the error triggers in turn a self-established pattern (at a random location x_{RE}) in present field $u^{pr_{HL}}$ of the HCM Layer. This activity does not spread to the past layer $u^{pr_{HL}}$ since it is below the propagation threshold $\lambda^{pr_{HL}}$.

Since now there is an active population in $u^{pr_{HL}}$, the learning rule formalized by (7.22) will associate, during the subsequent learning episodes, the population in $u^{pr_{HL}}$ with all populations in u^{pr} with an activation above the learning threshold λ^{pr}. The newly established connections $\underset{pr_{HL} \to pr}{a}$ will provide simultaneously input to connected sub-populations in u^{pr} whenever the high-level representation becomes active.

Second developmental stage

The second phase begins when the error is corrected and the tutor gives positive feedback to previously wrong prediction (Figure 8.22). The comparison between peaks in u^{pr} and u^{tm} decreases the resting level h^{tm} back to its initial value h^{tm}_{low}, and the activation in

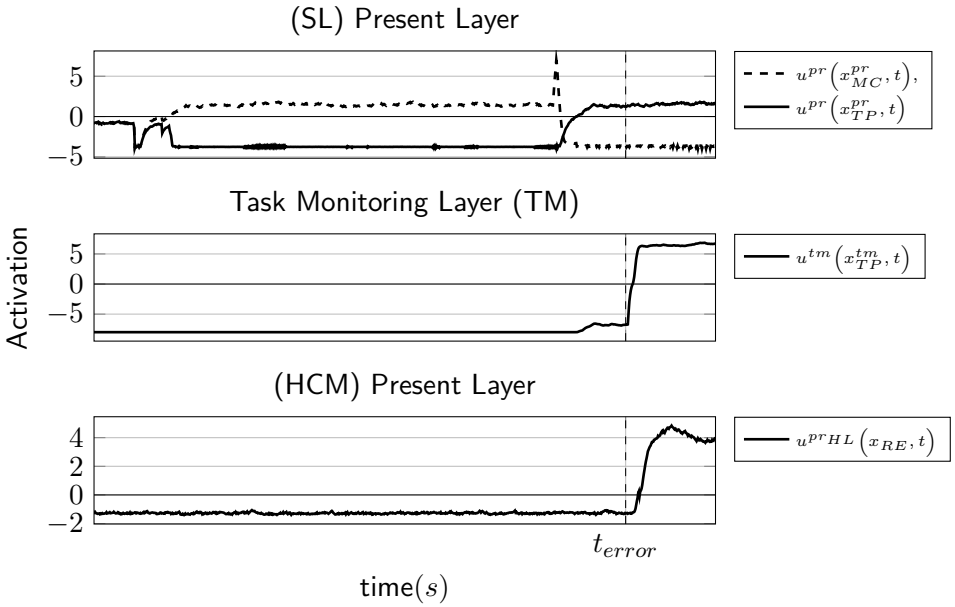


Figure 8.21: **Time evolution of fields u^{pr}, u^{tm} and $u^{pr_{HL}}$ during an error trial.** Following the insertion of MC, ARoS recalled TP as the next sub-goal. The prediction was verbalized by the robot and in response it received negative feedback from the tutor, at time t_{error}. This caused the raising of the resting level in u^{tm} resulting in a supra-threshold activation at location x_{TP}. Due to the connections $\underset{tm\rightarrow pr_{HL}}{R}$ to $u^{pr_{HL}}$, a peak also evolves in this field.

u^{tm} disappears. It also lowers the threshold $\lambda^{pr_{HL}}$ which makes the activation in $u^{pr_{HL}}$ propagate to $u^{pa_{HL}}$ forming a supra-threshold pattern. The learning rule guarantees that connections $\underset{pa\rightarrow pa_{HL}}{a}$ are formed between all currently active populations in u^{pa} and population representation in $u^{pa_{HL}}$. Thus, from this point on, when all the sub-goals that constitute the sub-task become active, the summed input will drive a peak in $u^{pa_{HL}}$ representing the completion of the sub-task.

The new populations in $u^{pr_{HL}}$ and $u^{pa_{HL}}$ will thus represent the planning and the accomplishment of an entire sub-task, respectively.

8.2.2 Fourth experiment: developing a second representation in the HCM Layer

Having demonstrated the construction sequence of the upper part of the toy vehicle, a fourth experiment was conducted in which the lower part of the construction was

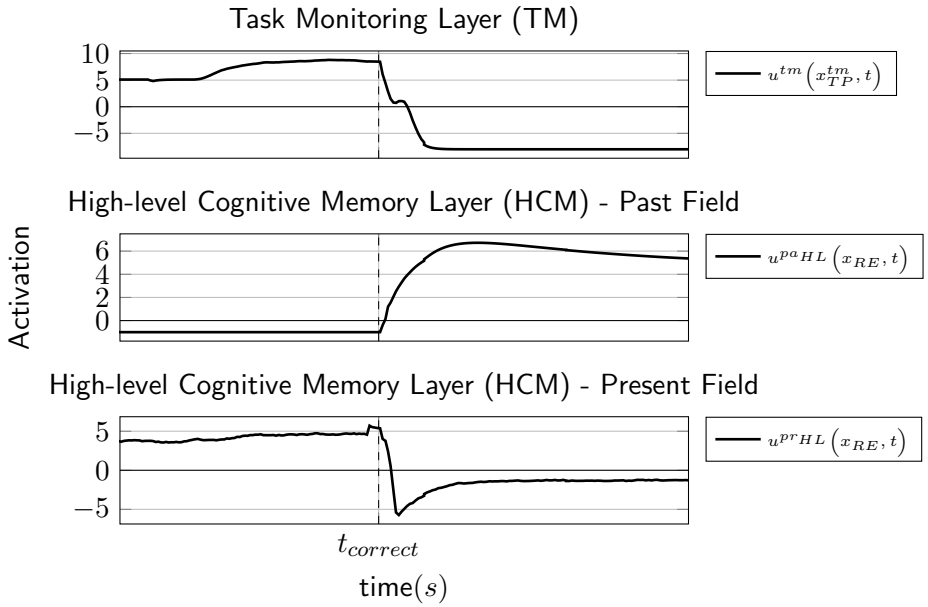


Figure 8.22: **Time evolution of fields u^{tm}, $u^{pa_{HL}}$ and $u^{pr_{HL}}$ during the correction of an error.** The field location of the high level activation in $u^{pr_{HL}}$ is represented by x_{RE}. At time $t_{correct}$, positive feedback from the tutor is given for a prediction that was previously wrong. This has two effects. On on hand, the threshold $\lambda^{pr_{HL}}$ is lowered, making the activation in $u^{pr_{HL}}$ to propagate to $u^{pa_{HL}}$. On the other hand, the resting level h^{tm} is decreased and the activation in u^{tm} disappears, which in turn removes the excitatory input of field $u^{pr_{HL}}$, causing the high-level population representation to decay back to resting level.

demonstrated. This lower part is formed by the base BA, left and right wheels (LW and RW) and the left and right nuts (LN and RN). The goal of this experiment was two-folded: to demonstrate the construction order of the remaining parts and to develop a second representation in the HCM Layer. Since it is intended for the lower part to constitute a separate representation from the top part it was considered for this demonstration that recalling any of the columns before the wheels and nuts would be a prediction error.

A fourth tutor (Figure 8.23) demonstrated the task by presenting two sequences [4]. Each sequence was demonstrated separately through the standard learning process. The following sequences were demonstrated:

[4]The video can be found at `http://marl.dei.uminho.pt/public/videos/TaskLearningVideo4.htm`



Figure 8.23: **Snapshot of the fourth experiment: demonstration.** The fourth tutor demonstrates the construction of the lower part of the toy vehicle.

$$\text{BA} \rightarrow \text{LW} \rightarrow \text{LN} \rightarrow \text{RW} \rightarrow \text{RN}$$

$$\text{BA} \rightarrow \text{RW} \rightarrow \text{RN} \rightarrow \text{LW} \rightarrow \text{LN}$$

Following the learning of this second sequence, a recall attempt was conducted. After presentation of the base by the tutor, the model recalled BC as the next step. In this context this constitutes an error, since the goal is to have wheels and nuts inserted before anything else. The tutor gave negative feedback, forming an error population in the TM Layer, and a new representation in HCM Layer. The tutor then performed a second demonstration in which connections were established between the sub-goal representations in u^{pr} and the active population in HCM.

Following the second learning period, a new recall attempt was made. Thanks to the new input from the HCM Layer which pre-activated all sub-populations constituting the assembly steps of the lower part, ARoS was able to recall the new sub-task before recalling the first column.

The division of the entire assembly task into two sub-tasks increases the flexibility of ARoS since, depending on the context, the assembly work could be started either by activating first the representation of the lower or the representation of the upper

part. Furthermore, the high-level planning also reduces significantly prediction errors. Figure 8.24 depicts the initial part of the probability tree in the situations where no high-level population was active (a) and when the representation of the lower part was active (b). One can see that when no high-level representation is active, the probability of choice for the next action is distributed between four sub-goals (Figure 8.24a). These sub-goals were all performed in the demonstrations immediately after the base was presented. When the high-level representation is active, the probability of choice favours only the sub-goals that are part of the lower part of the toy vehicle (Figure 8.24b).

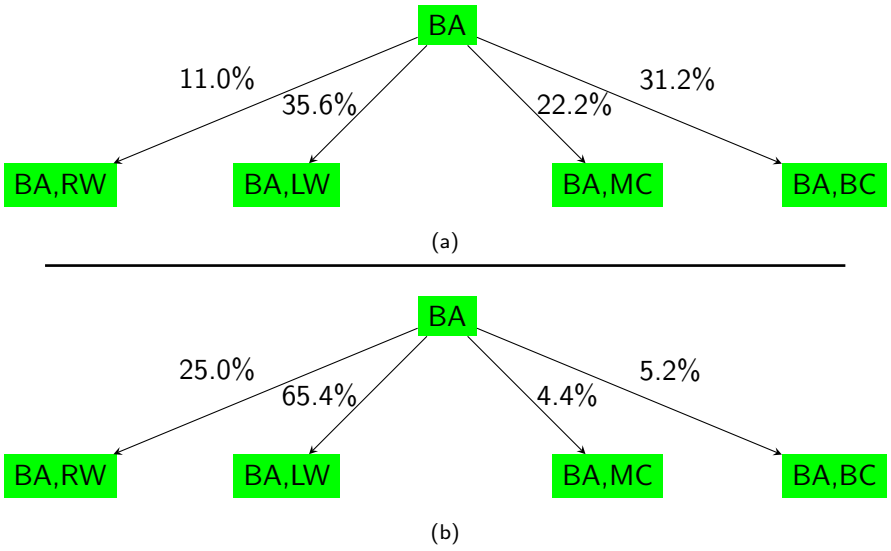


Figure 8.24: **Probability trees of the decision process for the first sub-goal after learning the higher-level sub-task representation**. (a) depicts the situation where there is no activation in $u^{pr_{HL}}$. In this case, the probability of decision for a specific sub-goal is distributed toward all the sub-goals that, during the demonstrations, followed the presentation of the base. (b) shows the situation where the population in $u^{pr_{HL}}$ representing the lower part of the toy vehicle is active. In this case, the decision graph is strongly biased towards the insertion of LW and RW.

8.3 Discussion

The experimental results presented in this chapter prove the ability of ARoS to acquire a representation of the sequential structure of a task, based on task demonstrations and

simple verbal interactions. Specifically, the successful learning shows that the dynamics of the neural fields and associated learning rules can deal with the challenging real-time constraints of a learning-by-demonstration approach in Human-Robot interaction. This is possible thanks to a neural processing mechanism that robustly represents the ordinal information of a demonstration, while abstracting from the natural temporal variations in observed task execution. By combining two complementary learning systems, the model only requires a reduced number of demonstrations to develop a sequence representation that encodes multiple paths of task execution.

The compound mechanism implemented by the model allows the accommodation of different demonstrations of the same task and reflect longer-term dependencies between sub-goals needed for successful task completion. Potential errors that initially may result from this implementation are corrected by successively increasing the length of past sub-goals history that become associated with each newly observed sub-goal. A unique feature of the model is that this time window is defined by the time course of the populations activity itself. This notion that working memory maintenance of recently achieved sub-goals contributes to the establishment of long-term memory representations is in agreement with the results of recent behavioural and neuro-imaging studies (Ranganath et al., 2005). These studies were focused on the dorsolateral prefrontal cortex and the hippocampus and were intended to study the transition between Working Memory and Long-term Memory.

A context dependent variation of the resting state of neural population that has been applied in various parts of the model is also supported by neurophysiological and computational studies (Asaad et al., 2000; Salinas, 2003) showing that a similar process is responsible for making neuronal properties, like selectivity to input, depend on context (Toth and Assad, 2002). It represents an efficient mechanism to change the gain of populations response to perceptual and other inputs and also affects the time course of population activity.

The ability to learn a sequence from a single demonstration is implemented by an activation based learning mechanism in the STM Layer which does not suffers the

problem of associative learning which usually must be slow to be robust. This activation based memory is used in an "Internal Rehearsal" process, to simulate the sequential inputs from the vision system. While the real-time demonstrations took around 30 seconds, the rehearsal process re-activated the demonstrated sequence thirty times in about three minutes during which the presence of the user was obviously not needed. More importantly, user acceptance will be greatly increased since the process saves many demonstrations. From the cognitive modelling point a view, the existence of a "fast-forward" rehearsal process translating task information from short-term to long-term memory is in line with recent findings. Neuro-physiological studies suggest the existence of a "replay" mechanism in the brain, which seems to have a role in memory consolidation (Euston et al., 2007; Ji and Wilson, 2007).

The model extension proved successful in chunking several sub-goal representations into high-level sub-tasks representations. This was made with no apriori knowledge on the nature of task, using prediction errors and feedback as a cue for segmentation. It was shown in the fourth experiment (Subsection 8.2.2) that these compact representations of temporally extended activity can be used to direct the task recall towards specific action paths. Such ability can prove useful towards increasing flexibility in action planning and coordination in tasks requiring joint activity (Hard et al., 2006a). Moreover, it is consistent with the idea that hierarchical encoding facilitates decision processes by providing top-down influence on specific chunks, reducing the number of competing sub-goals (Weaver and Arrington, 2013).

The results of the model implementation provide further support for the hypothesis that prediction errors are significant events serving as boundaries between chunks in hierarchical task representations (Reynolds et al., 2007; Zacks et al., 2011). However, other sources of information could be employed to guide the construction of hierarchical representations as well. For instance, it is known that prominent perceptual changes during task execution tend to be considered as action breakpoints (Zacks and Tversky, 2001; Hard et al., 2006b). Making a parallel with the toy vehicle task, one can think that the placement of the top platform constitutes a stronger physical variation than

the columns (since it is a larger object) and thus could be used to cue a boundary. Very interestingly for the dynamic neural field approach with a time window for learning, also small pauses in the linear sequence of demonstrated events may be used to define sub-task boundaries. These solutions represent bottom-up approaches to hierarchical task encoding in the same sense as the error driven segmentation. Top-down approaches could be employed as well, for instance by adding to the model the possibility of receiving more explicit verbal cues from the human expert, which may guide the learning of a hierarchical task representation (Saunders et al., 2007).

The replication of the two-layered structure from the SL to the HCM layer is the first step towards the development of a more advanced learning mechanism where the sequence learning occurs at different levels of abstraction (Sakai et al., 2003; Hard et al., 2006a). One important limitation that still remains to be solved is related with the formation of a second high-level representation in $u^{pa_{HL}}$ after a first has emerged. Because the learning rule associating $u^{pa}\big(x^{pa},t\big)$ to $u^{pa_{HL}}$ connects all $u^{pa}>0$ activations to the most recently achieved chunks ($u^{pa_{HL}}>h^{pa_{HL}}$), the second chunk would be associated with all the sub-tasks represented in u^{pa} belonging to the first chunk. A possible solution to this problem may be to introduce the notion of learning time window which is controlled by the time course of supra-threshold activity also to the higher-level sub-task representation.

The last two chapters describe an approach to task learning by demonstration, focused on the specific problem of learning the sequential structure of the task. It was assumed that the sub-goal representations already exist. The following two chapters address the problem of learning population representations of goal-directed actions from lower-level sensory input.

Part V

Emerging neuronal representations of motor primitives

Chapter 9

Model for the development of neural representations - Design

In this chapter a mechanism for evolving neuronal representations in a Dynamic Neural Field is presented. The mechanism is based on the Self-Organizing Map developed by Teuvo Kohonen (Kohonen, 1990) for the development of topological organized feature maps on a neuronal lattice. The model described here is applied to the development of neuronal representations of the individual motor primitives that are part of goal-directed actions. More concretely, the model clusters sensory input data describing the angular trajectories of the arm joints of a human grasping and manipulating objects. Consistently with the mirror neuron theory (Rizzolatti and Sinigaglia, 2010), these representations can be nested into chains of motor primitives, allowing the robot to anticipate the outcome of goal-directed actions. This is achieved here by using the sequence learning mechanism proposed in Chapter 7.

The chapter begins with the integration of Self-Organizing Maps for robotic applications in the framework of Dynamic Neural Fields. This is followed by a description of the chaining model of mirror neuron populations. The chapter proceeds with the detailed presentation of the model and the sequence learning mechanism for the development of chains of motor primitives.

9.1 General Overview

9.1.1 Evolving neuronal representations in Dynamic Neural Fields

One of the most distinctive characteristics of Dynamic Neural Fields is their ability to encode a behavioural dimension in a spatial representation. As stated in Chapter 3, this allows, for instance, to represent in parallel a variable number of instances of a perceptual input, avoiding the need for pre-specifying a fixed number of structures (e.g. nodes) to encode those perceptual instances. Moreover, because DNFs representations are inherently topological, the spatial location of the neuronal pools characterize the represented instances in terms of the dimension encoded in the field. Let us consider for instance a field representing the dimension color. An active peak at a specific location may represent for instance, a red object in the visual scene.

This spatial representation provides a simple and useful way of memorizing and representing information about a particular input dimension. However, in more complex applications, the instances to be represented are characterized not just by one, but by multiple dimensions, like for instance size, color or shape of an object. While multi-dimensional fields can be used to cope with this multi-dimensionality of input (Faubel and Schöner, 2008), the computational cost of integrating the field equation increases quadratically with the dimension. Thus, applications where dimensionality is above two are rare.

This question becomes even more pressing when representations become more abstract, as the ones required in higher-level reasoning processes. In the ARoS architecture, the DNFs represent instances of abstract dimensions (e.g. goal-directed actions like grasping) that can hardly be described in terms of one or two physical quantities. Thus, the metrics for the representations are defined heuristically. In other words, the field locations where activation peaks evolve are set by design to respond to specific features of the input (e.g. grip type) detected through predefined algorithmic processes. This approach was used in the sequence learning model presented in the previous chapters and is also employed in the ARoS's architecture for predicting ongoing actions of the

user. To achieve greater flexibility and apply the joint action model to novel tasks, it is highly desirable that the representations may emerge autonomously from the sensory input through some learning process.

Previous work from Minho's research group has already addressed this topic by proposing a Hebbian mechanism for splitting a neuronal population into two context specific representations (Erlhagen et al., 2007). This chapter describes a model for evolving complex neural representations in DNFs through use of self-organization principles that extends this previous work. More concretely, a learning and classification mechanism based on the Self-Organizing Map (SOM) (Kohonen, 1990) is proposed. The SOM principle proposed by Kohonen was developed to model the emergence of the topological organization that characterizes the mapping of sensory patterns in the cerebral cortex. Concretely, stimuli with similar attributes tend to be mapped closer together than the ones that are more dissimilar. Consequently, through a SOM it is possible to obtain a topological representation of a multi-dimensional input space according to a metric that is not pre-specified. Rather it emerges from high-dimensional inputs in an unsupervised manner.

The SOM model of Kohonen has encountered multiple applications in a wide variety of fields (Kaski et al., 1998; Oja et al., 2003). In the field of robotics, SOMs have been used primarily to reduce the dimensionality of the sensory input space. In one of the earliest applications (Walter and Schulten, 1993) a SOM was integrated in a control architecture for visuo-motor control of a robotic arm. The SOM was used to map visual input corresponding to the horizontal and vertical location of a target point in the images provided by two cameras. In Blank et al. (2005) a SOM formalized the first input layer to develop state representations as abstractions of high-dimensional perceptual data. A control rule was then learned that, given a current state and a goal state, tries to match both. Abstract state representations emerging through SOMs are also used in some RL approaches to define the state space over which the control policies are evolved (Smith, 2002; Toussaint, 2003; Provost et al., 2006). In Johnsson and Balkenius (2011) SOMs are employed to map the haptic perception of an object

provided by a robotic hand. To develop the map, haptic information from contact sensors was joined with proprioceptive information about the hand configuration.

SOMs can also be used to perform a direct mapping between input state and output response. This can be achieved by using a state/response pair as input to the network. A review of this technique for controlling robot manipulators is presented in Barreto et al. (2003). A similar approach was used in an autonomous underwater vehicle by Ishii and Nishida (2004) to map an input formed by the distance to obstacles given by distance sensors to an output actuation controlling the navigation direction.

9.1.2 Integrating Self-Organizing Maps and Dynamic Neural Fields

As demonstrated in Chapter 4, integration of the frameworks of Dynamic Neural Fields and Self-Organizing networks can be rather straightforward given that the two basic principles of a SOM, decision and kernel learning, are naturally embedded in the DNFs dynamics. Thus, one can easily model a SOM network where the output layer is formalized by a DNF.

Nonetheless, few works has been presented in this area. Early studies were performed by Takeuchi and Amari (1979); Amari (1980, 1983) which were mostly focused on mathematical analysis. Fellenz and Taylor (2002) presented a model of retinotopic organization (topographic mapping from retinal input to neurons). More recently, Alecu et al. (2011) compared different field equations and their relations to SOMs.

In this chapter, Self-Organizing DNFs are proposed as a method to autonomously extract meaningful representations from multidimensional input. The model is applied to the concrete problem of developing representations of motor primitives of arm movements directed towards a goal. To achieve this, the movements, described as angular trajectories of the arm joints, are segmented into motor primitives and represented as spatial patterns which are then clustered together with additional contextual information by a Self-Organizing DNF. Following principles of the mirror neuron theory, the resulting neuronal representations can be chained according to the sequence in which they are

observed to predict the outcome of goal-directed actions.

9.1.3 Chaining motor primitives

One of the most important developments in neuroscience in recent years was the discovery of the mirror neuron system (Rizzolatti et al., 1996; Gallese et al., 1996). It was first discovered in monkeys and later described also in the human cortex (Grèzes et al., 2003; Buccino et al., 2004) and in neural structures of other species (Prather et al., 2008). Mirror Neurons (MN) encode entire motor behaviours directed towards a goal. For example, grasping a piece of food activates a particular set of mirror neurons. But the exact same movement applied with no object in the scene, and thus with no particular goal, yields only a weak activation (Fogassi et al., 2005). Importantly, they fire both when an individual executes an action and when he observes the same action being executed by another individual.

It was hypothesized by Rizzolatti and colleagues that the mirror mechanism is a key factor in the interpretation of the intentions of others. According to this hypothesis, one is capable of understanding the purpose behind a particular observed goal-directed action because the observation triggers the exact same set of neurons that is active when the observer performs that same action. Subsequent studies by Fogassi et al. (2005) that used a sequence paradigm showed that the exact same motor action (e.g. grasping) is coded by different neuronal populations depending on the particular final goal of the action sequence (e.g. grasping for eating vs grasping for placing). This finding led to the hypothesis that MN are organized in chains of motor primitives directed towards a final goal. A chained organization of primitives can be used by the cognitive system to predict the outcome of ongoing actions of others. Through the motor resonance mechanism, an observed primitive activates the chain in which this primitive is embedded.

Since a particular primitive (e.g. a certain grasping behaviour) may be part of several chains linked to distinct goals, the obvious question is how the system selects the correct one. It has been suggested that the firing of MN not only reflects observed motor behaviour but also additional contextual cues that may disambiguate the situation. In

the study by Fogassi et al. (2005), for instance, the monkey could infer the ultimate goal of two sequences involving the grasping of food since in one situation a container was present in which the food had to be placed whereas the absence of the container indicated that the food was grasped for eating. fMRI studies indeed show that when additional contextual information is present, motor actions induce stronger activations in mirror neuron areas (Iacoboni et al., 2005).

Overall, the MN system is considered one of the core components of action understanding, joining action perception and contextual information to allow inference of action goals. In the last years, many computational models of the mirror neuron system have been proposed (Oztop et al., 2006). However, few have tried to make the connection between motion perception and the theory of mirror neurons chains as a way of interpreting human intentions. Erlhagen et al. (2007) used Dynamic Neural Fields to model neuronal populations coding motor primitives and competitive Hebbian principles for developing the chaining between populations. Thill et al. (2012) developed a SOM based mechanism to model the emergence of neuronal populations.

The model proposed in this chapter is related to both (Erlhagen et al., 2007) and (Thill et al., 2012) models since it also uses self-organizing principles to develop representations and builds on the chaining theory of mirror neurons to make predictions about action outcomes. Different to the early approaches, the present model integrates a methodology for segmenting continuous motion into primitives which are then clustered into neuronal representations. These representations, which are also context dependent, are kept synchronized with the actual observation, and the sequence learning is fully autonomous.

The model can be divided into the three core components: (1) segmentation; (2) development of representations of primitives; (3) sequence encoding. Thus far, the model has only been implemented in off-line scenarios in which the joint trajectories are fully known *apriori*. While some of the implementation choices, specially the ones related to segmentation, are based on this *apriori* knowledge it is important to stress that the algorithm could be adapted to deal also with on-line learning.

Next the three parts are described in detail. To facilitate the reading the same field label simplifications made in the previous thesis part are also applied here.

9.2 Segmentation

The first step in the development of motion primitives is the segmentation of a continuous motor action into simpler blocks. This requires the search for the start and end boundaries of time frames during which a relevant motion segment occurred. There are several ways in which this can be achieved, both in on-line and off-line processing. The majority of the segmentation methods is based on the analysis of motion speed and direction. Typically, the transition between different types of motions is characterized by expressive variations in these two measures. The technique employed here is the one used by (Breazeal et al., 2005) which requires the calculation of the Mean Square Velocity (MSV) of the joint trajectories describing the motion. Let $\theta_j(t)$ be the position of the joint j at time t. The MSV of the joints is given by

$$MSV(t) = \sum_{j=1}^{J} \left(\frac{\theta_j(t)}{dt} \right)^2 , \tag{9.1}$$

where J is the total number of joints being considered. In (9.1) the square of the velocities of all joints are summed and can be used to classify the movement. Finding a primitive means discovering the time window during which this quantity is greater than a certain threshold.

The segmentation method is illustrated in Figure 9.1. The first step consists in calculating the $MSV(t)$ for the trajectory and finding the first value of $MSV(t)$ above a minimum threshold value, λ_{low}^{msv}. An instant t that satisfies this condition is a candidate point for defining the start of a primitive. The following $MSV(t)$ values are then scanned to see if a second threshold value, $\lambda_{high}^{msv} > \lambda_{min}^{msv}$, is reached. If this happens then the candidate point is accepted as the start of a primitive. Having detected the beginning of a primitive the segmentation proceeds, now seeking for the time instant when the

value of $MSV(t)$ falls below λ_{low}^{msv}. This second point will be deemed as the end of the primitive. The time instants marking the beginning and end of a primitive p are labelled t_p^{st} and t_p^{end}, respectively. After finding an end point, the segmentation proceeds with the search for the next starting point.

The reason for the dual threshold approach is the following: while a motion may at some point reach a significant value of MSV, it normally begins and ends with lower values. Using just the higher threshold, could lead the segmentation to ignore the initial and final parts of a movement while the opposite solution, using just the lower threshold, could make the segmentation process consider as primitives segments that do not reach sufficiently high levels of motion.

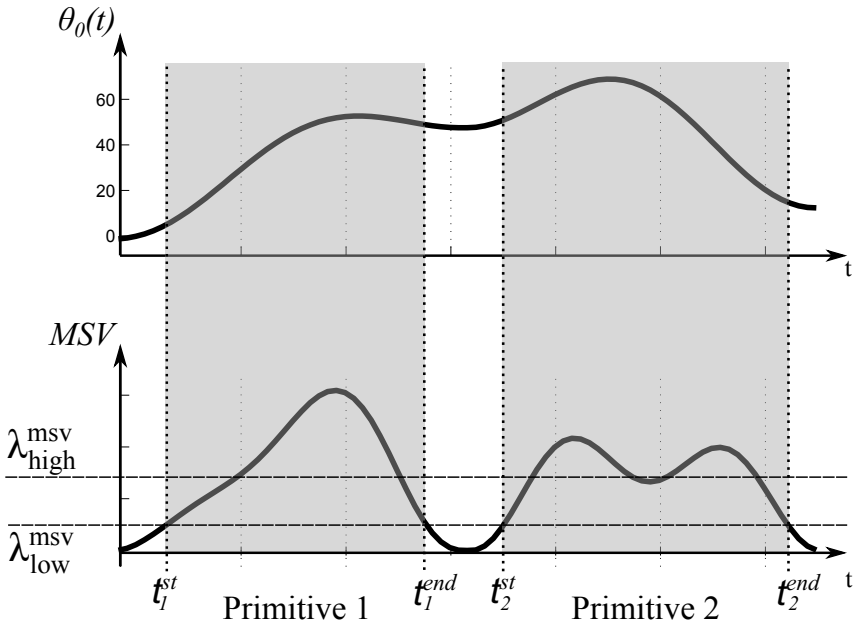


Figure 9.1: **Examples of joint trajectory and MSV.** The figure shows one joint trajectory and the MSV curve of the complete motion, with the detected primitives. Two time segments corresponding to two primitives were calculated based on the value of $MSV(t)$.

Like in Breazeal et al. (2005), the values for λ_{low}^{msv} and λ_{high}^{msv} are determined based on a combination of the average value and the standard deviation of the MSV, using, however, different weighting factors. The value of λ_{low}^{msv} is given by

$$\lambda_{low}^{msv} = \langle MSV \rangle - \frac{1}{2}\sigma_{MSV} \ , \tag{9.2}$$

where $\langle MSV \rangle$ is the average value of the function $MSV(t)$ over the total duration of a motion and σ_{MSV} represents the standard deviation. λ_{high}^{msv} is chosen simply equal to the mean value:

$$\lambda_{high}^{msv} = \langle MSV \rangle \ . \tag{9.3}$$

This value is different from the one in Breazeal et al. (2005) where the distribution of λ_{low}^{msv} and λ_{high}^{msv} was symmetric around $\langle MSV \rangle$. However, the values presented here proved to be more effective in the detection of the primitives.

9.3 Developing representations of primitives

Before the joint trajectories can be fed into the Self-Organizing Dynamic Neural Field, some pre-processing steps must be followed to normalize the input of joint angles $\theta_j(t)$ in the interval $\left[t_p^{st}, t_p^{en}\right]$.

The first normalization step consists in scaling the values of $\theta_j(t)$ between zero and one. This guarantees that the variations of $\theta_j(t)$ across joints can be compared in terms of overall magnitude. For instance, depending on the typical range of values attained by each of the joints, twenty degrees may represent a strong variation in one joint and only a minor variation in another. The choice of a $[0, 1]$ range is motivated by the second normalization step, which applies a complement coding of the input. The process of complement coding was introduced by Carpenter et al. (1991) to eliminate the differences between the absolute values of the input vectors between different samples. The two processes are detailed next, followed by the description of the self-organization mechanism.

9.3.1 Input normalization

To achieve the scaling between zero and one of each joint, two values, $\theta_{j_{min}}$ and $\theta_{j_{max}}$ must be defined, such that any value of $\theta_j(t)$ fits in the interval $[\theta_{j_{min}}, \theta_{j_{max}}]$. For the simulations in which the Self-Organizing DNFs will be employed all the motion trajectories are sampled *apriori*. Since each joint has a defined set of N trajectories, it is possible to know beforehand the maximum and minimum values reached by each joint. The trajectories and instants of time that represent the minimum and maximum values of $\theta_j(t)$ are given by

$$\{n_{j_{min}}, t_{j_{min}}\} = \underset{n,t}{\operatorname{argmin}} \left(\theta_j^n(t)\right) \ , \tag{9.4}$$

and

$$\{n_{j_{max}}, t_{j_{max}}\} = \underset{n,t}{\operatorname{argmax}} \left(\theta_j^n(t)\right) \ , \tag{9.5}$$

where $n \in 1, \ldots, N$ identifies the motion trajectory within the complete input set of N trajectories. The values of $\theta_{j_{min}}$ and $\theta_{j_{max}}$ are then given by

$$\theta_{j_{min}} = \theta_j^{n_{j_{min}}} \left(t_{j_{min}}\right) \ , \tag{9.6}$$

and

$$\theta_{j_{max}} = \theta_j^{n_{j_{max}}} \left(t_{j_{max}}\right) \ . \tag{9.7}$$

The scaled value of $\theta_j(t)$, which will be represented by $\theta'_j(t)$, is then defined by

$$\theta'_j(t) = \frac{\theta_j(t) - \theta_{j_{min}}}{\theta_{j_{max}} - \theta_{j_{min}}} \ . \tag{9.8}$$

9.3.2 Variable transformation

In many applications of motion encoding by SOMs, the input to the map is formed by each individual time sample. In the present application, the input is not a single data sample but a joint trajectory expressed as a function of time defined in the interval $[t_p^{st}, t_p^{en}]$. Since this function constitutes the input to a DNF mediated by a set of synaptic connections, one can, for the sake of coherence, perform a variable transformation and translate the time variable into a spatial dimension formalized by a spatial field distribution.

Since each joint is represented by an independent function one must define a dimension (as if it was a different set on neurons) for each joint. A set of transformations can then be defined, substituting the dimension t by x^{jt} for each joint, such that

$$\theta_j(x^{jt}) \equiv \theta'_j(t) \ . \tag{9.9}$$

The equation above represents the transformation which identifies the value $\theta_j\big(x^{jt}\big)$ at spatial location x^{jt} with the joint value θ_j at time t.

9.3.3 Complement coding

The evolving activation in the Self-Organizing DNF is the result of localized input mediated by a set of synaptic connections. Strong differences in the total amount of input $\theta_j\big(x^{jt}\big)$ may result in strong differences in the activation profile, which is undesirable. While a simple normalization using the Euclidean norm might be a solution to this problem, it has the disadvantage not to distinguish between trajectories with similar profiles but different amplitudes. To avoid this a complement coding process is used here (Carpenter et al., 1991).

The complement coding principle is inspired by the existence of retinal ON-cells and OFF-cells whose activity is complementary (Schiller, 2010). Normally, the complement coding principle consists of, starting with an initial vector v of size d with values

comprising the interval 0 and 1, and producing a new vector v^{cmp} with size $2d$. Half of the values of the new vector are equal to v and the other half is equal to $1 - v$:

$$v^{cmp} = \left(1 \middle| 1 - v\right) . \tag{9.10}$$

A graphical representation of the complement coding process is depicted in Figure 9.2. The function $\theta_j(x^{jt})$ forms the argument of an ON channel and $1 - \theta_j(x^{jt})$ represents the OFF channel.

Complement coding implies that both the presence and absence of features (or in this case, parts of the trajectory) are coded. In the context of this work, applying a complement coding principle to the input trajectories ensures that the total amount of input is constant. It also implies, for the self-organizing mechanism, that the presence of an input component can compete with the absence of that same component. This balances the competition and thus facilitates the formation of separate field representations.

Formally, performing the complement coding of the input trajectories implies rearranging the transformation function, beginning by defining its limits such that

$$x_0^{jt} \equiv t_p^{st} , \tag{9.11}$$

and

$$x_{last}^{jt} \equiv 2\left(t_p^{end} - t_p^{st}\right) . \tag{9.12}$$

The middle value of the field of size $2d$ is defined by

$$x_{mid}^{jt} \equiv t_p^{end} - t_p^{st} . \tag{9.13}$$

Finally, the input $Q_j(x^{jt})$ of each joint for the Self-Organizing DNF is defined by

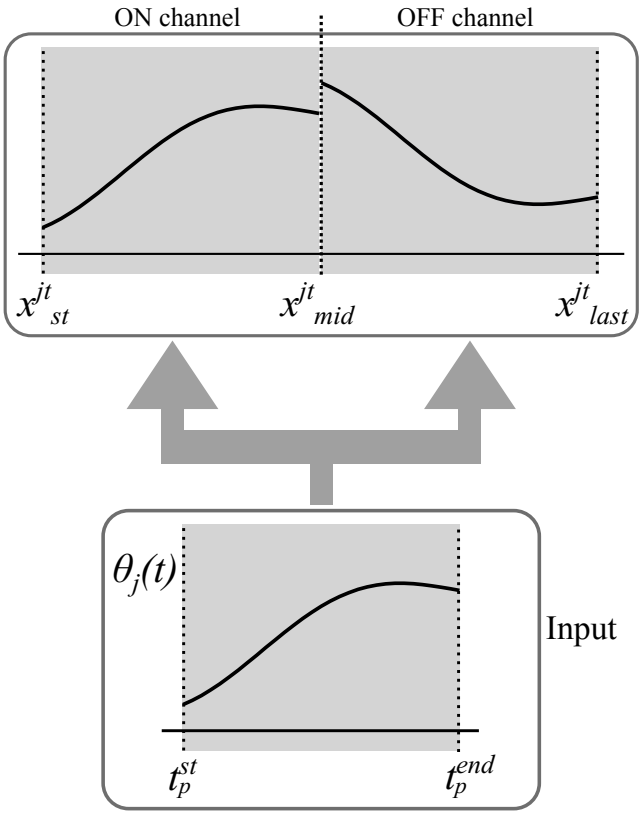


Figure 9.2: **Process of complement coding.** The input trajectory segment corresponding to a primitive p is transformed into a spacial distribution with a length twice the transformed value of $t_p^{end} - t_p^{st}$. The first half of the spatial distribution has the same shape as the input trajectory, while the second half is given by its complement-to-1 value.

$$Q_j(x^{jt}) = \left[\theta_j(x^{jt}) \middle| 1 - \theta_j(x^{jt})\right] = \begin{cases} \theta_j\left(x^{jt}\right), & x \in \left[x_0^{jt}, x_{mid}^{jt}\right[\\ 1 - \theta_j\left(x^{jt} - x_{mid}^{jt}\right), & x \in \left[x_{mid}^{jt}, x_{last}^{jt}\right] \end{cases}, \tag{9.14}$$

where the branches define the ON and OFF channels.

9.3.4 The context signal

The Self-Organizing DNF will receive as input not only the joint trajectories, but also contextual information that expectedly is responsible for the formation of different representations for the same primitives. For the present simulation study, this contextual

signal is given by the color of a graspable object. The continuous *Hue* component of the *HSV* color space can be directly mapped on a one-dimensional field. Thus an observed color will be represented by a localized Gaussian input function $\kappa^{hue}(x^{hue})$ (Figure 9.3).

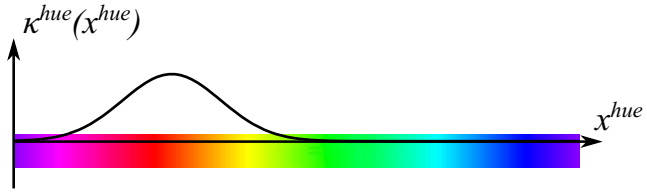


Figure 9.3: **Example of a contextual input.** Contextual information is provided as a color feature, encoded through the Hue component of the HSV color space. The value of the component is defined by a Gaussian curve defined over the hue dimension x^{hue} and centred over the corresponding hue value.

Since the amplitude of the Gaussian curve does not transmit any information, it can be normalized so that its maximum value is equal to one. This is necessary since, similarly to the joint trajectories, complement coding is also applied to the contextual signal $\kappa^{hue}(x^{hue})$. The resulting function is defined by

$$Q^{hue}(x^{hue}) = \left[\kappa^{hue}(x^{hue}) \middle| 1 - \kappa^{hue}(x^{hue})\right] . \tag{9.15}$$

After pre-processing all input components, the next step is the concatenation of all the input components (trajectories and context) under a single dimension. Let x^{obs} represent this dimension (obs stands for *observation*). Then, $Q^{obs}(x^{obs})$ is described as

$$Q^{obs}(x^{obs}) = \left[Q_0(x^{0t}) \middle| \cdots \middle| Q_J(x^{Jt}) \middle| Q^{hue}(x^{hue})\right] . \tag{9.16}$$

From a practical point of view, Q^{obs} works as a buffer, storing in its spatial pattern the joint trajectories of the last observed primitive and context. The pattern it encodes is time independent since its values do not evolve continuously in time. Rather, it is only updated at each t_p^{end}, when a new primitive has just been detected.

9.3.5 Self-Organizing Layer

The Self-Organizing Layer (SOL) is the layer where the representations of primitives will emerge. It is formed by two components: a DNF, $u^{som}(x^{som},t)$, with a Gaussian kernel and a set of synaptic connections $\underset{obs\to som}{a}(x^{obs},x^{som},t)$ that connects the input dimension x^{obs} to the output dimension x^{som} (the model is depicted in figure 9.4). The input $s^{som}(x^{som},t)$ of field u^{som} is given by

$$s^{som}(x^{som},t) = C^{som}\int z^{obs}(x^{obs},t)\ \underset{obs\to som}{a}(x^{obs},x^{som},t)dx^{obs}\ , \tag{9.17}$$

which corresponds to the propagation of an input $z^{obs}(x^{obs},t)$ through the synaptic connections $\underset{obs\to som}{a}$. It will be shown next that this input function z^{obs} depends directly on Q^{obs}. The parameter C^{som} controls the input strength.

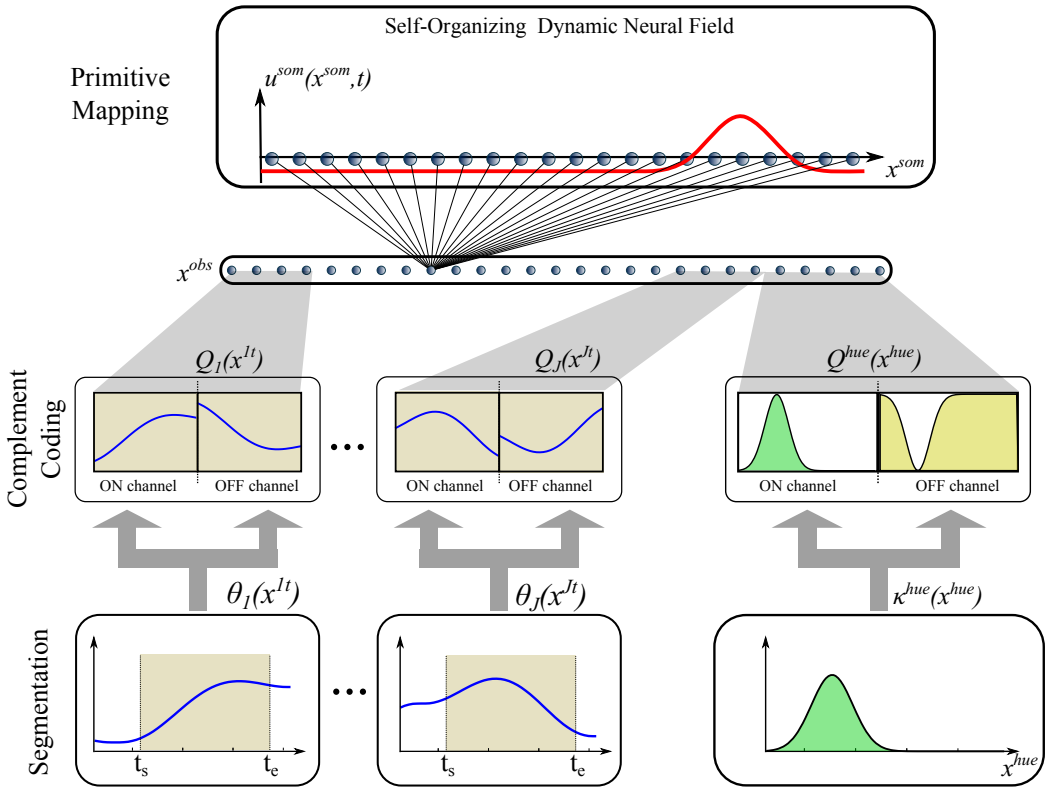


Figure 9.4: **Model for primitives encoding.**

The functioning of the model is based on the assumption that the trajectory or

trajectories shown to the model are presented several times and although they will vary to some degree, they tend to be similar for the same motion. This means that the primitives defined by the segmentation algorithm reoccur often. It is also assumed that if more than one motion is shown, their demonstration follows a stochastic order.

When a new primitive is detected and Q^{obs} is updated, the field activation is reset to resting level. The field u^{som} is then iterated continuously until the next primitive is detected. Meanwhile the weights of the synaptic connections are adapted by the Kohonen principle. After several demonstrations, the activations in u^{som} tend to appear on different locations depending on the specific primitive that has produced them. This means that the active populations have specialized to represent inputs belonging to that primitive.

The construction of the input z^{obs} is one of the most important components of the model. In a typical implementation of a SOM, z^{obs} would be equal to Q^{obs}. The SOM would then cluster similar patterns into different neuronal representations, formalized by different populations. However, in Chapter 4 it was shown that the connection weights to the neurons of a representation are shaped as the principal component of the set of input patterns that are being represented. This property can be used to differentiate between patterns in situations where the difference between primitives is not enough to drive the development of distinct populations (e.g. identical grasping behaviour for different goals). To that end, the input function z^{obs} must be formalized as the difference between a function $\hat{Q}\big(x^{obs}\big)$ and the first principal component encoded in the active population of u^{som}. Let this component be represented by $c^{obs}\big(x^{obs},t\big)$. Function z^{obs} is then given by

$$z^{obs}\big(x^{obs},t\big) = \hat{Q}^{obs}\big(x^{obs}\big) - \hat{c}\big(x^{obs},t\big)\ , \tag{9.18}$$

where $\hat{Q}^{obs}$ corresponds to the normalization of the equation Q^{obs}, given by

$$\hat{Q}^{obs}(x^{obs}) = \frac{Q^{obs}(x^{obs})}{\sqrt{\int Q(x^{obs})^2 dx^{obs}}} \ , \tag{9.19}$$

and $\hat{c}^{obs}$ corresponds to the normalization of the component c^{obs} which is encoded in the active population in the Self-Organizing DNF. The component c^{obs} can be obtained by the back propagation of the activation in u^{som} through the synaptic connections $\underset{obs \to som}{a}$. This is formalized by

$$c^{obs}(x^{obs}, t) = \int \Big[\ u^{som}(x^{som}, t) \Big]_0^+ \ \underset{obs \to som}{a}(x^{obs}, x^{som}, t) dx^{som} \ . \tag{9.20}$$

Conversely to Q^{obs}, the function c^{obs} is time dependent since one of its terms is the field activation u^{som} that evolves continuously in time.

The inclusion of this mechanism is based on the idea that by subtracting the first component of a cluster to its members, one is in fact diminishing the similarity between them, fostering the competition effect. As a result, the activation peak will shift its location. An example of this dynamics is visible in Figure 9.5, that depicts a situation where a new Q^{obs} is presented to a Self-Organizing DNF that already has specialized populations. The new Q^{obs} will tend to "search" the population encoding the primitive most similar to itself, providing stronger input to it. However, as soon as an activation rises, the back-propagation will change the value of z^{obs} and the previously excited region of the field will now be inhibited, causing the activation to drift and stabilize at a different (albeit nearby) location.

As the representations become more specialized in representing specific primitives, the differences between Q^{obs} and the principal components encoded in their populations becomes very small. At that stage, when an activation is formed, the value of z^{obs} tends to zero and the activation in u^{som} will depend only on the self-sustaining dynamics, maintaining the same location.

The adaptation rule of the synaptic connections $\underset{obs \to som}{a}$ is similar to (4.34) with the input term replaced by $\hat{Q}^{obs}$:

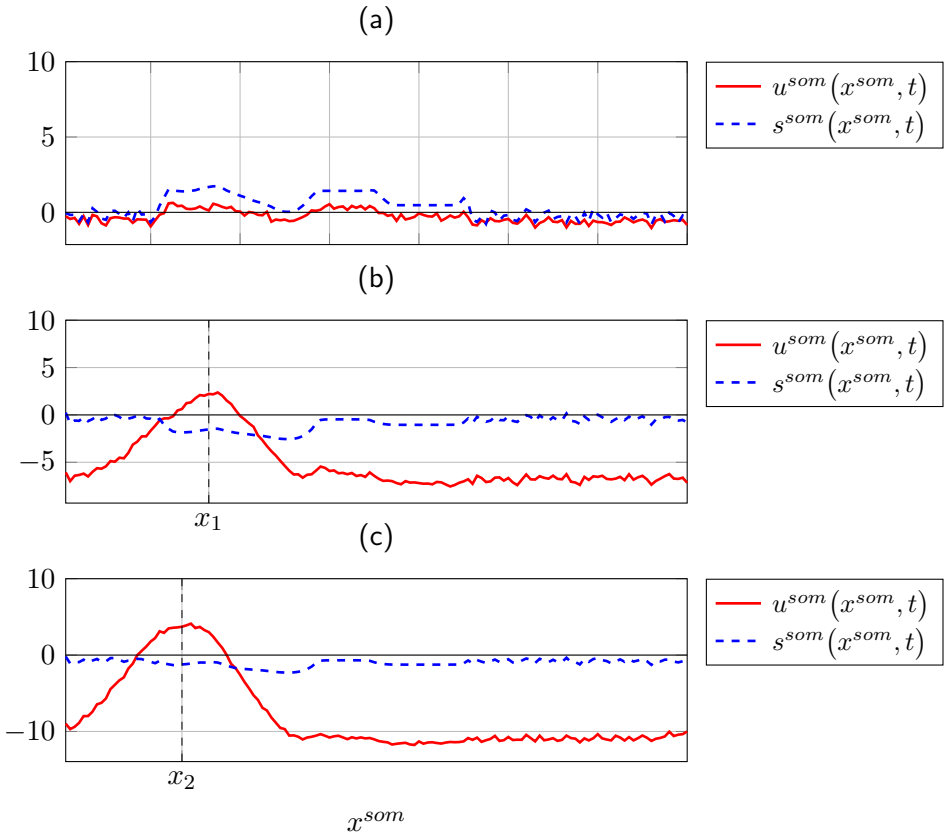


Figure 9.5: **Effect of the back-propagation.** The figure illustrates the situation where a new pattern Q^{obs} is presented to a Self-Organizing DNF which already has specialized populations. The new pattern will produce stronger input on the DNF population whose weights are already more tuned to encode it (a). This will produce a localized activation, that through the back-propagation inhibits the input. If the difference between the new input and the one already coded is significant, the inhibition will produce a depression on s^{som} (b). This causes the activation to shift location slightly, "searching" the locations of less inhibited neurons (c) that due to the synaptic adaptation rule will become more tuned to the new input. After several learning trials, a distinct population represents the new input.

$$\tau^{a_{som}} \frac{\partial}{\partial t} \underset{obs \to som}{a} \left(x^{obs}, x^{som}, t\right) = \left[u^{som}(x^{som}, t)\right]^{+} \hat{Q}\left(x^{obs}\right) . \tag{9.21}$$

After each iteration step, the weights are normalized through (4.36) that can be rewritten for this particular case:

$$\underset{obs \to som}{a} \left(x^{obs}, x^{som}, t\right) = \frac{\underset{obs \to som}{a} \left(x^{obs}, x^{som}, t\right)}{\left\| \underset{obs \to som}{a} \left(x^{obs}, x^{som}, t\right) \right\|_{x^{obs}}} . \tag{9.22}$$

9.4 Encoding the sequence of primitives

Having formed a set of context dependent primitive representations, the next step consists in establishing the chain of associations between primitives that will allow the inference of the final outcome of an observed action. This is achieved by using a simplified version of the model presented in Chapter 7. In this case only the core module of the model, formed by the past and present layer will be required (see Section 7.2.2). While the inter-layer associations between u^{pa} and u^{pr} are the same, the input terms are different, since in this case, the fields receive contributions from u^{som}.

Figure 9.6 depicts the past and present layers u^{pa} and u^{pr} respectively, and u^{som}. Every peak activation emerging in u^{som} will provide contributions to the input of the past and the present fields (s^{pa} and s^{pr}, respectively). Thus, the sequence learning module will be fed with a different sequence of input patterns for each motion demonstrated. The populations in u^{pa} and u^{pr} representing different primitives have the same locations of the ones in u^{som}. Thus, as the populations in u^{som} become more specialized and activations begin to rise on different locations, associations between populations of u^{pa} and u^{pr} are formed. These associations are reinforced whenever the trajectories reoccur (because the sequence of input patterns also reoccurs). After several demonstrations, these associations will be strong enough to produce prediction peaks in u^{pr} resulting from the propagation of u^{pa}. Thus, an observed primitive is encoded in u^{som} which propagates to u^{pa} causing a prediction peak in u^{pr}. The prediction peak is located at the population that will be excited by the next primitive. It appears ahead of the actual observation of the primitive. Because of this property, the peaks in u^{pr} emerging from the propagation of activity in u^{pa} can be considered as an anticipation of subsequent primitives.

The sequencing model is parametrized to function in its learning mode, meaning that the baseline value of h^{pa} is set to its lower value $h^{pa}_{bas} = H^{pa}_{low}$. (see Subsection 7.2.2). Thus, the Sequence Learning Layer is constantly learning from the observed sequences and making predictions at the same time.

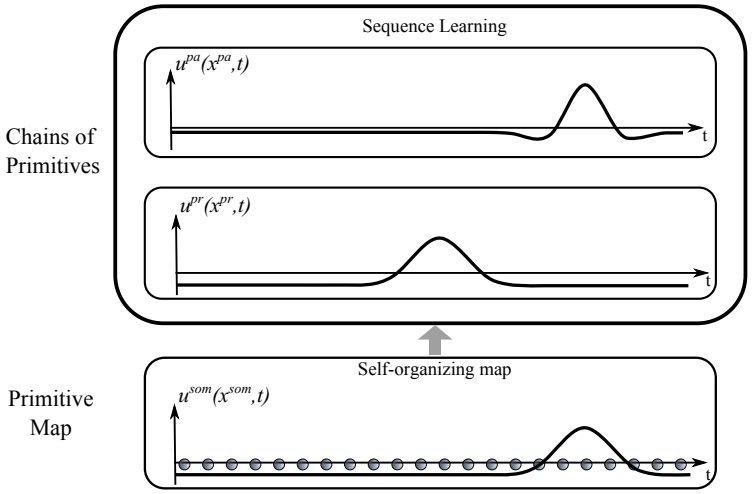


Figure 9.6: **Model for chaining representations of primitives.**

The external input to the past and present layers is given solely by the activity u^{som}. Thus, (7.2), that describe the input to s^{pa} can be replaced with

$$s^{pa}\left(x^{pa},t\right) = C^{pa}\left[u^{som}(x^{som},t) - \lambda^{som}\right]_0^+ + \\ + \int f_{\lambda^{pr}}\left[u^{pr}\left(x^{pr},t\right)\right] \underset{pr\to pa}{E}\left(x^{pr},x^{pa}\right) dx^{pr} \ , \tag{9.23}$$

where the first term $\left[\ u^{som}\left(x^{som},t\right) - \lambda^{som}\right]_0^+$ describes the input contribution given by the SOM field. This contribution is gated by the threshold λ^{som} ensuring that only activations in u^{som} above this value are propagated. The gating function limits the width of the active region in u^{som} that can effectively contribute to the input of field u^{pa}. This ensures that the activations in u^{pa} are spatially separated, which is a requirement for the maintenance of several stable activations.

The input of the present layer s^{pr} is formalized by

$$
\begin{aligned}
s^{pr}\big(x^{pr},t\big) = C^{pr}\Big[\; u^{som}\big(x^{som},t\big) - \lambda^{som}\Big]_0^+ \\
+ \int f_0\Big[\; u^{pa}\big(x^{pa},t\big)\Big] \underset{pa\to pr}{I}\Big(x^{pa},x^{pr}\Big)dx^{pa} \\
+ \int f_0\Big[\; u^{pa}\big(x^{pa},t\big)\Big] \underset{pa\to pr}{a}\big(x^{pa},x^{pr},t\big)dx^{pa}\;,
\end{aligned}
\tag{9.24}
$$

where again, the first term, $\Big[\; u^{som}\big(x^{som},t\big) - \lambda^{som}\Big]_0^+$, describes the gated contribution from u^{som}.

The learning rule is similar to (7.8), employed in Chapter 7, with an additional decaying term. This term is used to cope with the fact that the locations of the representations change during learning. Thus, not only it is necessary that new connections are established, but also that the old connections decay, preventing activity in u^{pa} from propagating to the former positions of the representations in u^{pr}. The learning rule is given by

$$
\begin{aligned}
\tau_a \frac{\partial \underset{pa\to pr}{a}\big(x^{pa},x^{pr},t\big)}{\partial t} = f_{\lambda^{pr}}\Big[\; u^{pr}\big(x^{pr},t\big)\Big] f_{\lambda^{pa}}\Big[\; u^{pa}\big(x^{pa},t\big)\Big] \times \\
\times \Big[e^{pr}\big(x^{pr},t\big) - \eta \underset{pa\to pr}{a}\big(x^{pa},x^{pr},t\big)\Big] - \alpha \underset{pa\to pr}{a}\big(x^{pa},x^{pr},t\big)\;,
\end{aligned}
\tag{9.25}
$$

where $\alpha \underset{pa\to pr}{a}$ is the decaying term regulated by the constant α. The error term $e^{pr}\big(x^{pr},t\big)$ is formalized by (7.9).

This page has been intentionally left blank.

Chapter 10

Model for the development of neural representations - Implementation and results

This chapter reports simulation results of the implementation of the model described in Chapter 9. It begins by presenting the setup, inspired on principles of the mirror neuron theory. It then describes results of the several stages of the mechanism for formation and sequential encoding of motor primitives, namely the motion segmentation, the formation of representations and the encoding of sequences of primitives. The chapter finishes with a discussion of the results.

10.1 Simulation inspired by Mirror Neurons

The main goal of the model described in the previous chapter is to explain the formation of neuronal representations in dynamic neural fields representing high dimensional inputs. As a particular implementation example, the model is applied here to the development of population representations of motor primitives in the MN system. Recent neurophysiological studies show that these representations are highly context depend-

ent, meaning that the same primitive is represented by different neuronal populations depending on the sequential context in which it is executed. Inspired by the idea of a chaining of primitive representations in the MN system (Fogassi et al., 2005), the Self-organizing Dynamic Neural Field is coupled with a simplified version of the sequence mechanism described in Chapter 7, allowing the anticipation of the outcome of observed motor sequences.

The simulations reported here intend to show that the model is in fact successful in addressing the above mentioned topics. Concretely, this chapter has three main purposes:

1. To show that the model based on dynamic neural fields combined with the SOM algorithm is capable of autonomously evolving neuronal representations of high-dimensional inputs that do not have an euclidean metric.

2. To demonstrate how the model is capable of evolving context dependent representations of motor primitives, in accordance with principles of the mirror neuron theory.

3. To show that the coupling of the Self-organizing Dynamic Neural Field with the sequencing mechanism allows the cognitive system to anticipate the outcome of sequences of motor primitives.

10.2 Setup

The experimental paradigm simulates a real world scenario where a robot interacts with a human, recognizes his/her actions and tries to anticipate the underlying outcome. Following the previous HRI experiment with ARoS, the simulations address the situation where the robot observes a demonstrator performing goal-directed actions with a single hand, manipulating a green wheel and a pink nut located on a table. It is assumed here that a motor resonance principle is implemented, that is, the mapping from observed motor acts onto the corresponding representations in the robot's motor system already

exists. For the applications, the congruency should be relatively strict, meaning that, for instance, specific grasp types are matched. This assumption is in line with neurophysiological studies that describe MNs with different levels of congruency (broadly and strictly congruent) (Rizzolatti and Sinigaglia, 2010).

For the model this means that it is presented with a set of N arm trajectories, where each arm trajectory Θ_n is translated into seven individual rotational joint trajectories θ_j of a 7 DoF-arm:

$$\Theta^n = \{\theta_1^n(t), \cdots, \theta_7^n(t)\} \ . \tag{10.1}$$

The trajectories are composed by a "reaching" phase, where the hand approaches and grasps a coloured object using a particular type of grip and a "goal" phase, where the object is manipulated for a particular purpose, like inserting it on a peg or handing it over. The arm trajectories were generated through the movement planning algorithm proposed by Costa e Silva et al. (2014) that guarantees the human-like arm movements of the robot ARoS (see Section 2.3). Contextual information is conveyed by the colour of the manipulated object described as the Hue parameter in the HSV colour space.

The model was implemented using Matlab. For the simulations, the DNFs were discretized to have a total of 140 units (or neurons). The field dynamics is integrated with circular boundary condition. The Gaussian kernel of u^{som} was defined with a $\sigma = 10.5$, an amplitude $A = 0.9$ and a negative bias of $w_{inhib} = -0.5$. The resting level is chosen as $h = 0.7$. Solving the equation $W(x) = h$, where $W(x)$ is the integral of the interaction kernel it is possible to determine the width of the activation bumps which is approximately 21.3 units (Amari, 1977). This value will be used as a comparison measure for the degree of separations between the representations. All the other parameter values are described in Appendix B.

The simulations were divided into two phases. In the first phase, it is shown that the model is effectively capable of differentiating between motor primitives, forming separate representations. This is achieved by providing the model with demonstrations

of two distinct goal-directed motor actions which are separately encoded as different sets of motor primitive representations. It is also demonstrated how the primitives can be chained to anticipate the action outcome.

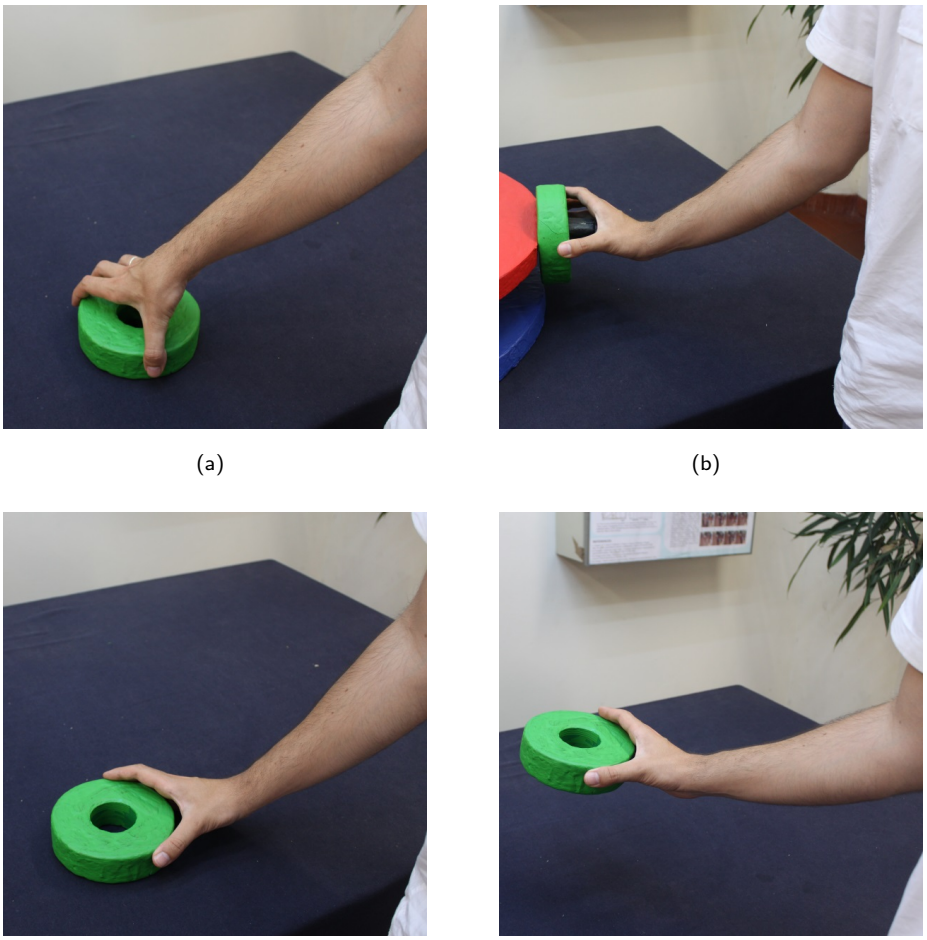
In the second learning phase, a third goal-directed motor action is added to the input set. It has primitives in common with one of the previous motor actions but the contextual information differs. The model behaviour illustrates how contextual information drives the formation of new representations for the common primitives. The model thus explains the finding of MN populations that encode a certain grasping behaviour as a function of the context in which the action sequence is embedded.

The setup of the different phases and respective results are presented next.

10.3 First phase - Description

In the first experiment, the model was presented with two different goal directed actions. Each action is formed by an arm trajectory and a context indicated by the color of the object. The motion of the first action is composed of a "reach-to-grasp" phase performed towards a green wheel placed at a specific location of the table, employing an above grip (Figure 10.1a), and of a goal phase when the wheel is inserted on a peg placed on the left of the arm position (Figure 10.1b). The second action sequence is also formed by a "reach-to-grasp" phase towards a green wheel, but with a side grip (Figure 10.1c), and a handover gesture towards the partner (Figure 10.1d) as the second phase.

For each goal directed action, six complete arm trajectories were generated, varying in the initial posture of the joints and in the position of object to be grasped. The initial position of the joints varies within a range of 8° while the position of the grasped object varies horizontally relatively to the robot within a range of 80 mm. As an example, Figure 10.2 depicts the 6 trajectories of the second joint for the first motion part. This variability is intended for testing the ability of the model to cluster similar movements in a single neural representation (providing that the context is similar). The complete input set of arm motions is thus composed of twelve trajectories.



(a) (b)

(c) (d)

Figure 10.1: **Example of motions employed in the first phase.** (a) A grasp motion applied a green wheel with an above grip. (b) Insertion of the wheel on a peg. (c) A grasp motion applied on a green wheel with a side grip. (d) Handover of the wheel.

The simulations were composed by a total of 150 trials. In each trial, an arm trajectory from the input set was randomly selected and segmented into primitives. For each primitive p a starting and ending time (t_p^{st} and t_p^{en}, respectively) was determined by the segmentation process.

The primitives were supplied to the SOL layer, together with the contextual information as described in Section 9.3.4. Here it is assumed that in a real world scenario, each primitive would be computed during the interval between its end time t_p^{st} and the

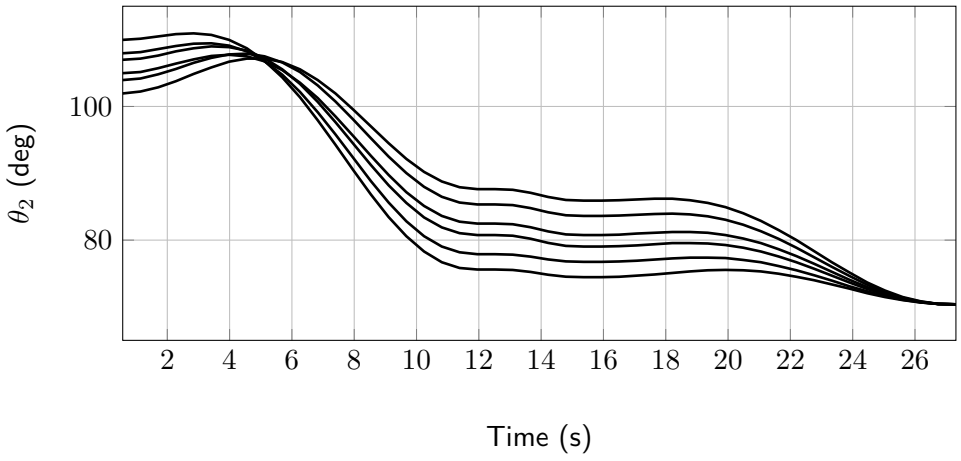


Figure 10.2: **Trajectories of** θ_2. The figure depicts the 6 trajectory variations for joint 2 for the handing-over task.

ending time of the next primitive t^{st}_{p+1}. For simulating this situation, each primitive is iterated during a time T_p, which is directly proportional to the time interval between the end of the primitive and the end of the next. Thus T_p is given by:

$$T_p = c^T \left(t^{en}_{p+1} - t^{en}_{p} \right) . \tag{10.2}$$

where c^T is the constant of proportionality. The last primitive in the sequence is iterated during a fixed time.

After the iteration time is over, the field is algorithmically set to its resting value h^{som} and the following primitive is presented to the network. During this time, the fields u^{pa} and u^{pr} of the sequence model are iterated continuously.

Upon the presentation of all primitives of a motion, all fields from the model are reset to their resting values and a new trial begins with a new arm trajectory being randomly chosen from the input set.

10.4 First phase - Results

This section presents results of the several stages of the algorithm, namely the segmentation, the primitive encoding and the sequence learning.

10.4.1 Segmentation

The segmentation of the arm trajectories using the MSV consistently resulted in two primitives per motion. Figure 10.3 depicts the $MSV(t)$ of one of the trajectories and the resulting segments. It is clearly visible that the motion has two time intervals where the value of MSV increases, the first one corresponding to the reaching phase and the last one to the goal phase. Figure 10.4 presents the seven joint trajectories θ_j of the segmented motion together with the time windows defining each segment.

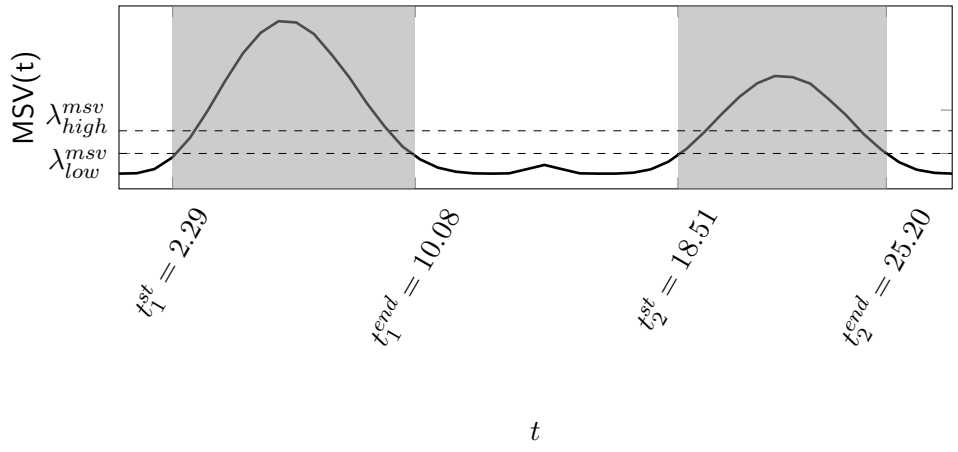


Figure 10.3: **MSV of an arm trajectory.** The figure shows the two time windows where the value of the MSV is higher than the threshold value λ_{low}^{msv}, which correspond to the reaching and final phase of the arm motion. Values within the time windows also reach the λ_{high}^{msv}, validating the primitives.

10.4.2 Primitive encoding

For each trial, the primitives resulting from the segmentation were sequentially presented as input to the Self-Organizing DNF. For each primitive the field was iterated during the time T_p. After this time the activation pattern of the Self-Organizing DNF is logged

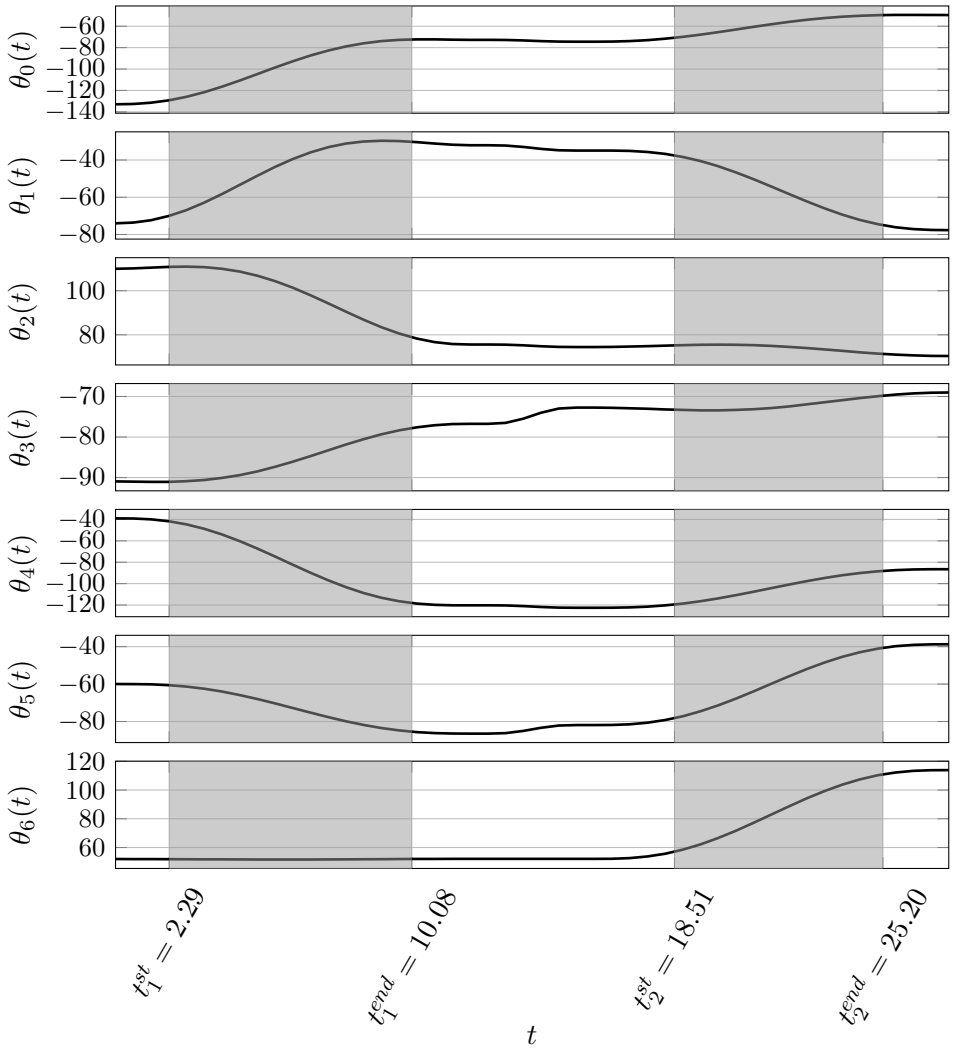


Figure 10.4: **Segmented trajectories.** One can see that the segments defined by the MSV analysis correspond to areas of larger variation of the joints angles.

and labelled with the arm trajectory and the primitive that it represents. This allows one to verify if primitives of the same type and belonging to the same motion activate the same population. Considering that the arm trajectories from the input set are grouped into two types according to the specific goal, and that each motion is divided into two primitives, a total of four representations was expected to emerge. Results show that this is indeed the case.

Figure 10.5 depicts the recorded activation profiles for the first and second primitives of trial 151 (Grasp Wheel and Insert) and trial 156 (Grasp Wheel and Handover). The primitives are labelled according to table 10.1. In the figure, on can easily see that the four populations have their maximum at distinct spatial locations. However, the localized patterns, with a width of about 20 units overlaps to some extent.

Actions	Label	Description
Grasp Wheel and Handover	GWS	Grasp Wheel with Side Grip
	HW	Handover wheel
Grasp Wheel and Insert	GWA	Grasp Wheel with Above Grip
	IW	Insert Wheel

Table 10.1: **Labelling of primitives for the first simulation phase.**

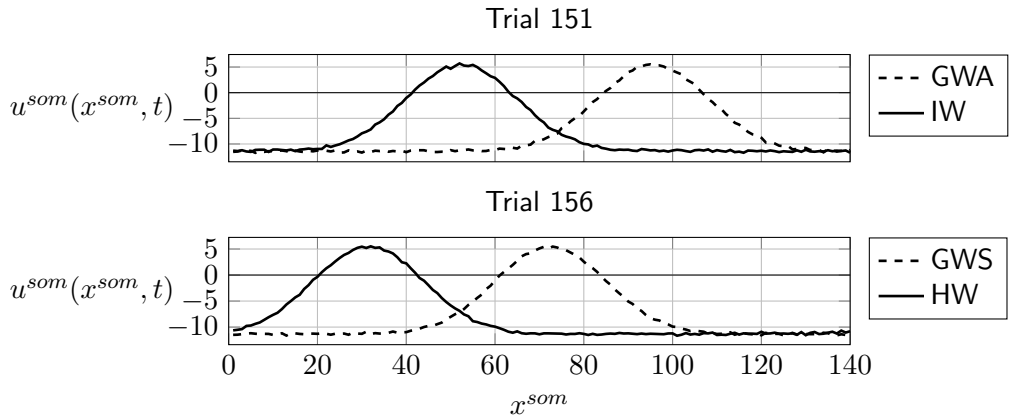


Figure 10.5: **Emerging field activations.** One can see that four distant populations are formed, representing the four primitives. The populations have a width of around 20 units.

For each activation a peak location is calculated, defined as the center of the active population. Since the logging process associates each motion drawn from the input set with the current peak location, it is possible to group the previous peak locations of similar primitives, and calculate, for each primitive, a centroid and standard deviation of these peak locations. Figure 10.6 depicts the evolution of these average locations and their standard deviations as a function of the number of trials. One can see that there is an organization dynamics through which the representations, starting from a similar location, spread away in the field until all four primitives are represented separately. Around trial 80 the standard deviation of the distribution of activations of each primitive is already small compared to the distance between the population centroids.

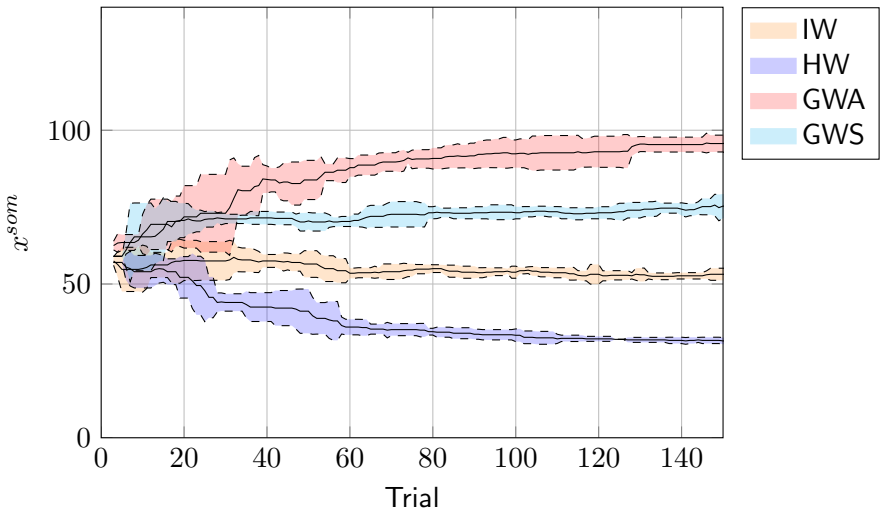


Figure 10.6: **Centroid and Standard Deviation of the representations.** The shaded areas mark the space between $\pm 2\sigma$ (standard deviation) around the centroid (black line) of the peak locations of the activations of the same primitive. One can see that the centroids spread away as the number of trials advance, and that the standard deviation around the centroids decrease, meaning that the peak locations of activations of the same primitive tend to vary to a lesser degree.

The plot from Figure 10.7 compares the evolution of the minimum distance between the centroids of the four populations. Again, it is possible to verify that this distance increases continuously and tends to stabilize around a value slightly above 21 units, which is close to the width of the activations in the field u^{som}.

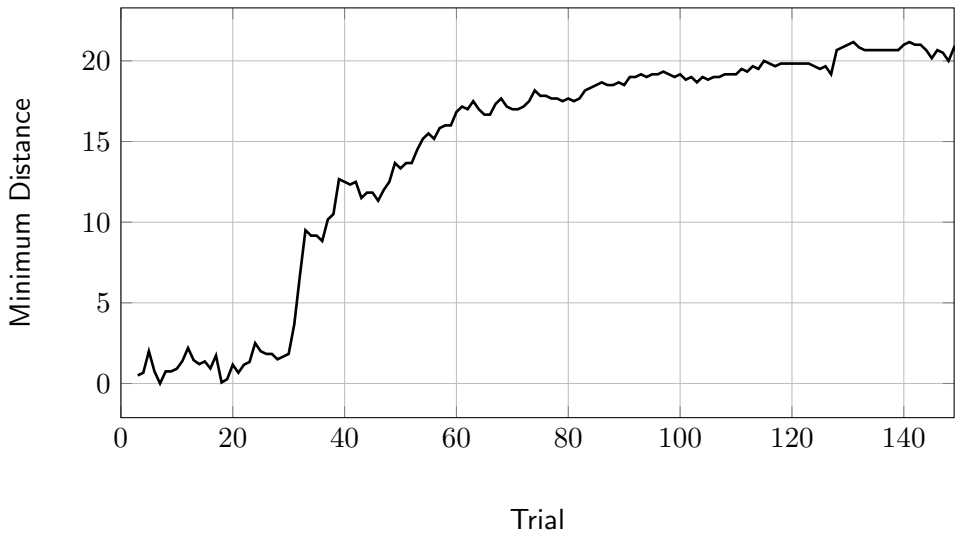


Figure 10.7: **Minimum distance between centroids.** During the first 150 trials, the minimum distance between the average location of the activations belonging to the same primitive increases until it stabilizes around a value of about 21 units.

10.4.3 Sequence learning

During each learning trial, the fields u^{pa} and u^{pr} of the sequence model are continuously iterated, receiving input from u^{som}. Thus, as the representations in u^{som} start to differentiate, so do the ones in u^{pa} and u^{pr} getting input from u^{som}. After some trials, the sequential mechanism begins to produce predictions in u^{pr} about the subsequent primitive. Figure 10.8 shows the temporal evolutions in u^{pa}, u^{pr} and u^{som} for the trial number 60. It can be easily seen that the supra-threshold activations in u^{pr} evolve earlier than the corresponding ones in u^{som}. The field dynamics thus anticipates the second primitive of the trial.

To capture the accuracy of the predictions, a logging process was conducted where, in each trial, the peak location of the activation in u^{pr} was recorded for the first primitive, and the peak location of the activation in u^{som} was recorded for the second primitive. An error measure was then computed simply by calculating the distance between these values. Figure 10.9 depicts a black bar for each trial where a prediction ahead of time

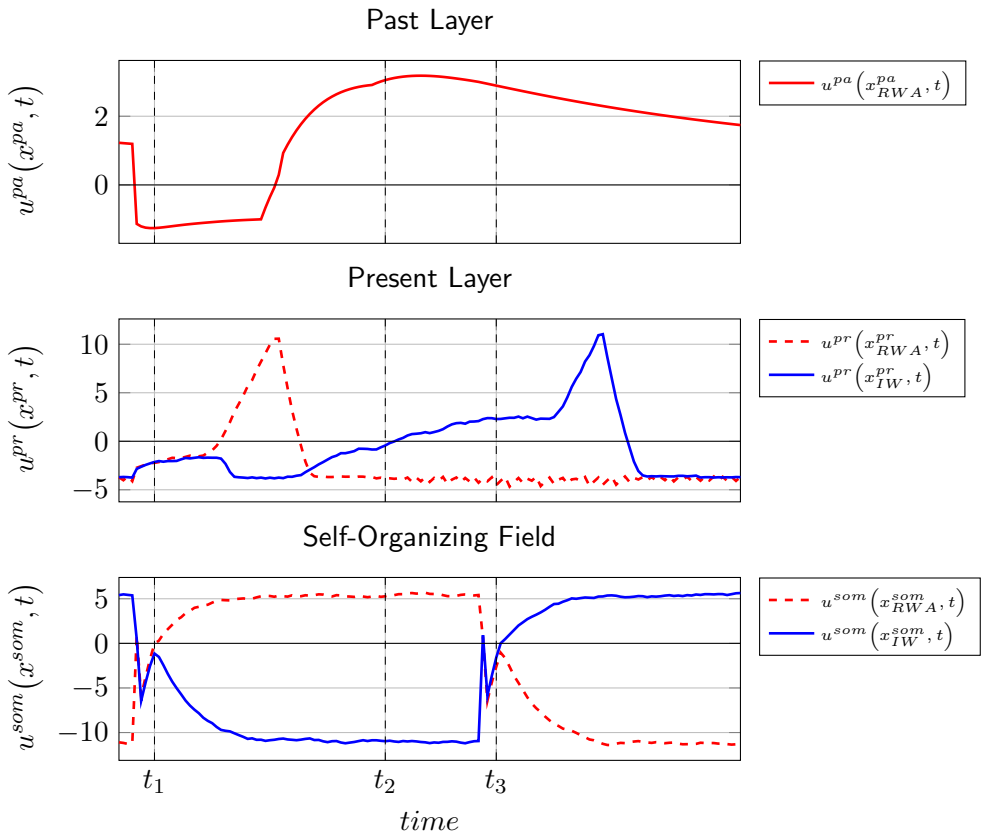


Figure 10.8: **Sequence field activations during a trial.** The figure depicts the time profile of the activations in $u^{pa}(x^{pa}, t)$, $u^{pr}(x^{pr}, t)$ and $u^{som}(x^{som}, t)$, at the peak locations of primitives RWA and IW. At time t_1 the representation of RWA in $u^{som}(x^{som}, t)$ was active, projecting to $u^{pr}(x^{pr}, t)$ and then to $u^{pa}(x^{pa}, t)$. This caused a prediction peak in $u^{pr}(x^{pr}, t)$ that rise above zero at time t_2, anticipating the appearance of the second primitive (IW) in $u^{som}(x^{som}, t)$ at time t_3.

occurred. In the figure it is visible that the model made a prediction in the large majority of trials (129) and that the error decreases continuously until a value of around 1 unit is reached.

10.5 Second phase - Description

In the second simulation phase, a third goal directed action was added to the two already existing ones. The new action corresponds to the grasping of a nut and its placing in a container. The contextual information for the new action is different however. A

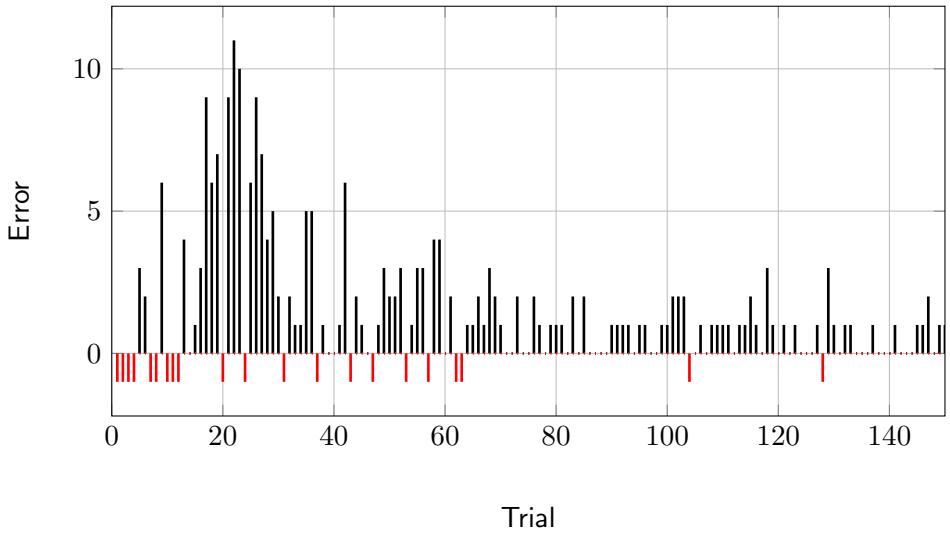
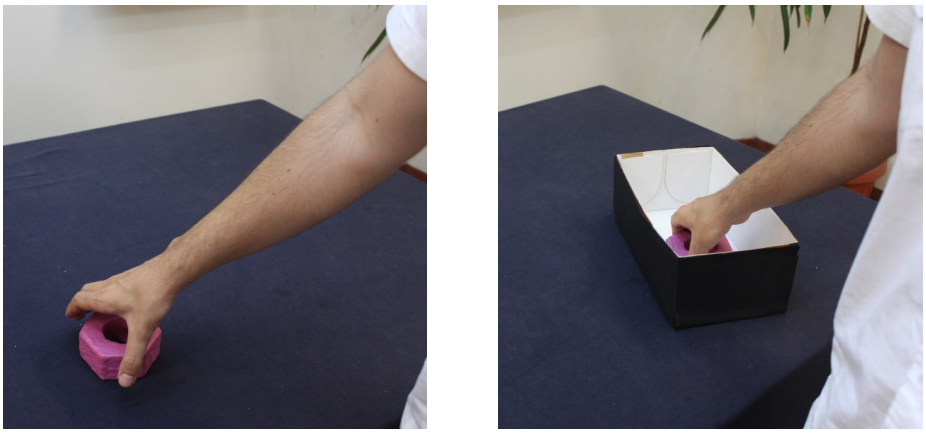


Figure 10.9: **Prediction error.** Positive black bars represent trials where a prediction was made, while negative red bars marks trials where no prediction occurred. The height of the black bars represents the prediction error, computed as the distance between the peak locations of the anticipation in u^{pr} and the activation of the second primitive in u^{som}. The model predicted the next primitive ahead of time in 129 trials. At the end of the simulation the error had a value of about 1 spatial unit.

magenta nut has to be manipulated instead of a green wheel. Like with the other actions, six arm trajectories were generated. The trajectories had an approach phase during which the object was grasped from above (Figure 10.10a) and a goal phase when the object was placed in a container located to the right of the initial object position (Figure 10.10b). While the first phase of this new action was characterized by a trajectory similar to one where the wheel was grasped from above, it was expected that the different contextual information would cause the emergence of a new population representation. This was indeed the case.

The second simulation phase was composed of 350 new trials. In each trial, a sample from the input set (now composed of 18 motions) was randomly chosen as input to the model.



(a) (b)

Figure 10.10: **Example of motions employed for the second learning phase.** (a) Grasping a magenta nut with an above grip. (b) Placing the nut in the container.

10.6 Second learning phase - Results

This section reports the results of the second phase of the learning experiments. Concretely, it addresses the development of new representations and impact on the chaining mechanism.

10.6.1 Encoding new primitives

Similar to the previous goal-directed actions, the segmentation of the new action sequence resulted in two new primitives represented by two distinct populations in the field. The same logging process applied to the first four primitives was used to capture the peak locations of the new populations and calculate their centroids and standard deviations. Figure 10.11 depicts the centroids and standard deviations of the old and new populations. Labels are chosen according to table 10.2.

Figure 10.11 shows that the activations have shifted their location, with new neurons being recruited to encode a new primitive. In some cases, an established population splits into two, each one recruiting new neurons. This is a common behaviour when the

Actions	Label	Description
Grasp Wheel and Handover	GWS	Grasp Wheel with Side grip
	HW	Handover wheel
Grasp Wheel and Insert	GWA	Grasp Wheel with Above grip
	IW	Insert Wheel
Grasp Nut and place	GNA	Grasp Nut with Above grip
	PN	Place Nut

Table 10.2: **Labelling of primitives for the second simulation phase.**

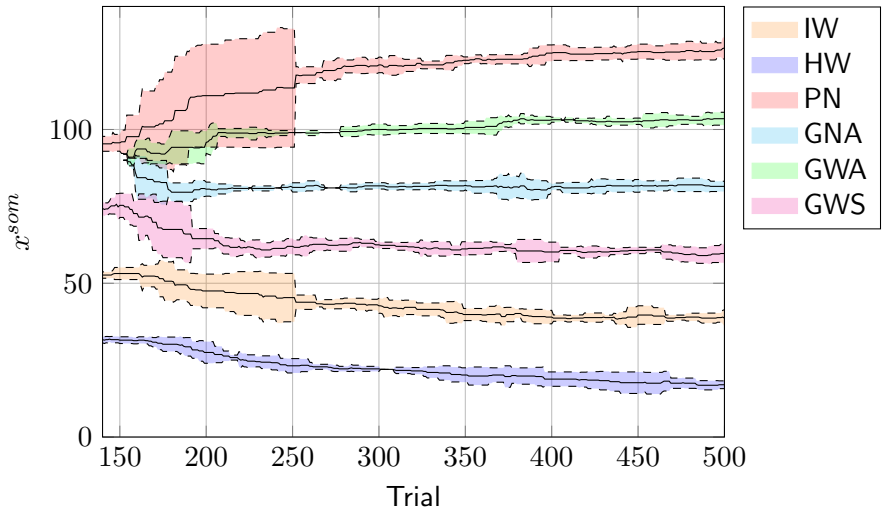


Figure 10.11: **Centroid and standard deviation of the representations - second phase.** The shaded areas correspond to the difference between $\pm 2\sigma$ (standard deviation) around the centroid (black line) of the peak locations of the activations of the same primitive. One can see that the centroids shifted location when the populations rearranged to accomodate the new representations.

inputs are similar like for instance, the primitives GWA (Grasp Wheel from Above) and GWA (Grasp Nut from Above). While GWA was already established when the second simulation phase began, its population splits into two, one representing GWA and the other representing the new primitive GNA (Figure 10.12). This reflects the known property of SOMs, which map similar input onto neighbourly locations. The figure also shows that, as expected, the same motor primitive "above grip" was encoded in two different populations reflecting the distinct contextual information.

Figure 10.13 depicts the minimum distance between the centroids of peak loca-

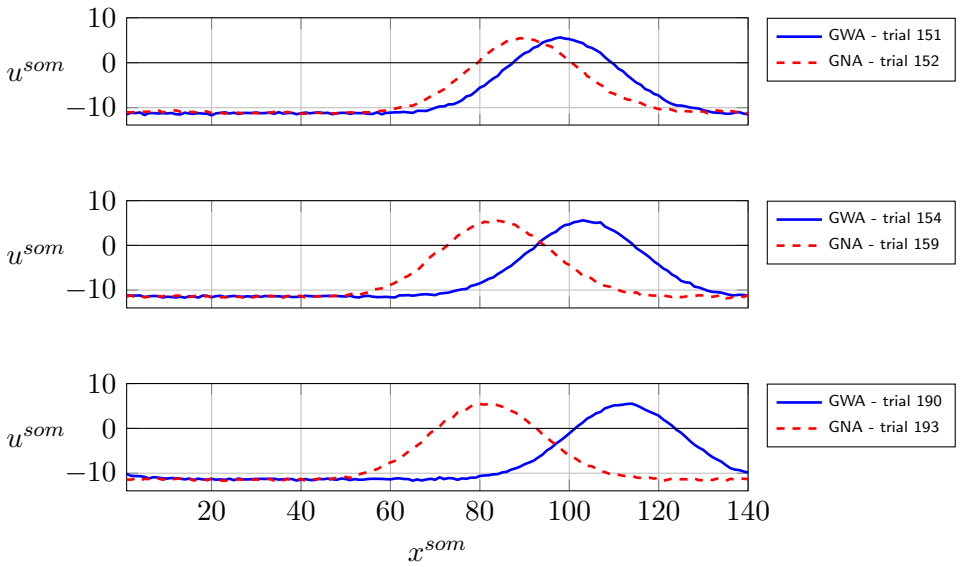


Figure 10.12: **Splitting of populations in u^{som}.** With increasing number of learning trials, the population that was initially representing the primitive GWA (Grasp Wheel from Above) splits into two, one continuing to represent GWA, and the other representing the new primitive GNA (Grasp Nut from Above).

tions belonging to the same primitives for the second simulation phase. The distance drops abruptly when the second simulation phase begins, because, initially, the new inputs produce activation patterns spatially close to the already existing ones. As the populations reorganize, the distance increases reaching again a value of above 21 units.

10.6.2 Adjusting the sequences

As the new representations appear and the old ones shift location, the adaptable connections $\underset{pa\to pr}{a}$ that connect u^{pa} to u^{pr} also reorganize, establishing new associations and decreasing the previous ones. Figure 10.14 depicts the same error measure used in the first simulation corresponding to the absolute difference between the locations of the anticipation in u^{pr} and the activations in u^{som}. While in the beginning there are less predictions (visible by the increased frequency of red bars) and the error values are greater, as the number of trials increases also the number of predictions ahead of time increases (a total of 312 from 350 trials) and the error stabilizes at about one unit.

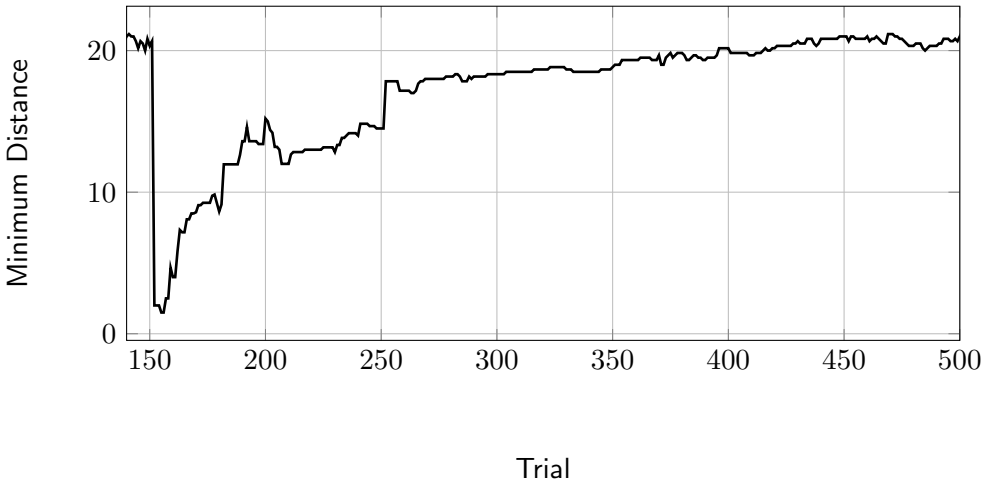


Figure 10.13: **Minimum distance between centroids - second phase.** During the initial trials of the second phase, the minimum distance between the average location of the activations belonging to the same primitive decreases. This happens because the new inputs tend to be initially encoded near the places of the already existing representation. As the simulation proceeds, the activations spread and the minimum distance increases until reaching the previous value of near 21 units.

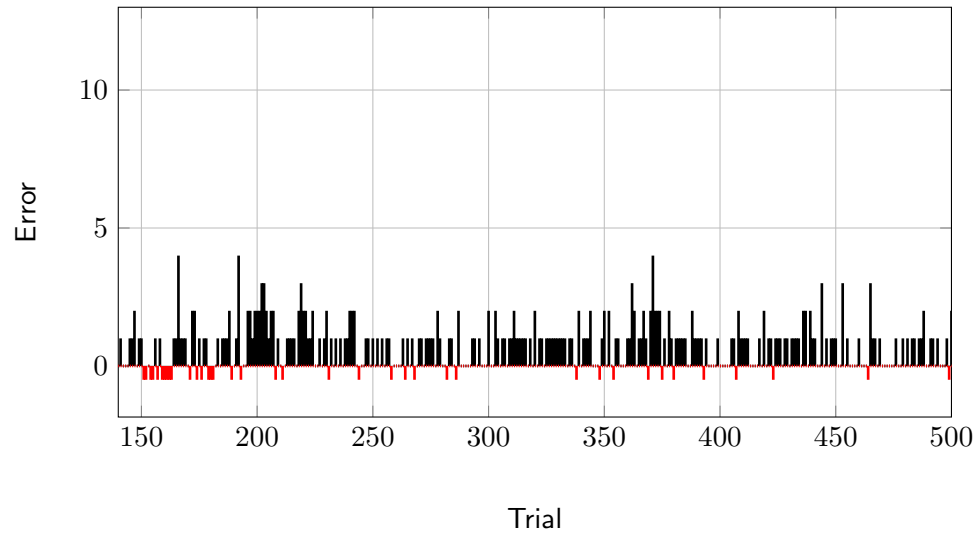


Figure 10.14: **Prediction error - second phase.** Positive black bars represent trials where a prediction was made, while negative red bars mark trials where no prediction occurred. The height of the black bars represents the prediction error, computed as the distance between the peak location of the prediction in u^{pr} and the activation of the second primitive in u^{som}. The model predicts the next primitive ahead of time in 312 trials.

10.7 Effect of the Back-propagation

To verify the advantage of propagating the activity in u^{som} back to the input, the first phase of the simulation was repeated without the use of back-propagation. Mathemat-

ically this means that equation (9.18) is replaced by:

$$Z(x^{obs}, t) = \hat{Q}^{obs}(x^{obs}) \tag{10.3}$$

The simulation was run for 500 trials. Figure 10.15 depicts the centroids and standard deviation of the peak locations. One can see that without the back-propagation, two of the primitives (GWA and GWS) fail to form separate representations. These two primitives are the ones from the initial set that present greater similarity. To show this, the trajectories belonging to the same primitive were grouped and the mean values of the corresponding input vectors were calculated. The dot product was then calculated between all the mean trajectories. Results are presented in Table 10.3. The primitives GWA and GWS are the ones with higher value of the dot product, meaning that, according with the similarity measure of the SOM algorithm they are indeed the most similar ones.

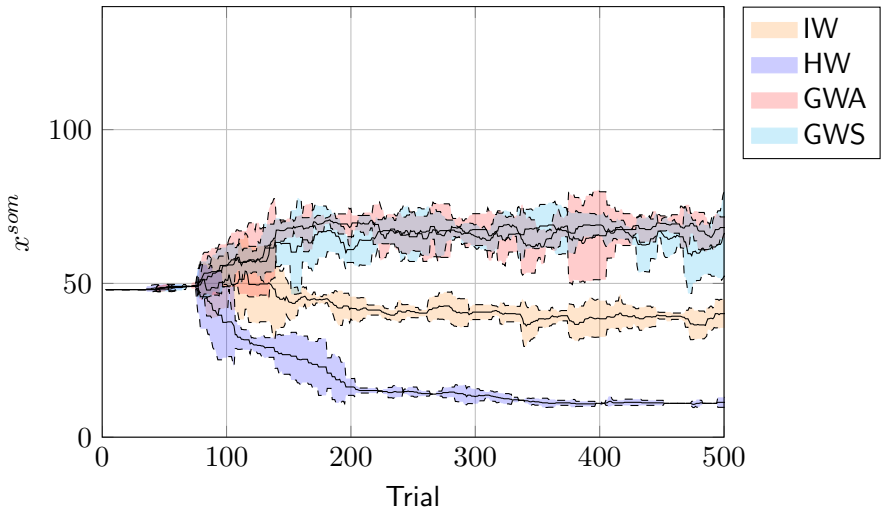


Figure 10.15: **Centroid and standard deviation of the representations.** One can see that without the back propagation, two of the primitives, GWA and GWS, fail to separate completely.

	GWA	IW	GWS	HW
GWA	1.0000	0.8216	0.8823	0.7929
IW	0.8216	1.0000	0.8264	0.8052
GWS	0.8823	0.8264	1.0000	0.7849
HW	0.7929	0.8052	0.7849	1.0000

Table 10.3: Dot products between the mean trajectories corresponding to each primitive. GWA and GWP have the highest correspondence and are thus more difficult to separate by the combined SOM and field dynamics.

10.8 Discussion

As the most important result, the model simulations show that neuronal representations of multi-dimensional inputs that cannot be immediately represented by a single metric dimension may evolve in dynamic neural fields. By extracting the statistical similarities between the inputs, the model is capable of generalizing one-dimensional topographic maps. This is possible thanks to the use of the SOM mechanism whose bio-inspired properties are at the core of the model. Furthermore, the results also show that the SOM principles can be combined with other learning mechanisms to endow robots with additional cognitive abilities. In this work, the self-organization dynamics was combined with a sequencing mechanism allowing the system to learn chains of motor primitives directed towards a final goal. The chained organization can be used in HRI to predict actions of the human partner.

The segmentation through MSV proved to be effective, since the resulting segments are consistent with a "natural segmentation" that a human observer would perform. It confirms results of Breazeal et al. (2005) who applied a similar methodology. For the simulation, human-like trajectories of the robot arm were used. It remains to be seen how the segmentation method would cope with human motions captured with a computer vision system. Real human motions may prove to be less consistent in terms of trajectories, resulting in a more variable number of segments and in greater differences between segments corresponding to the same action. An approach for dealing

with this issue could be for instance, changing the motor description, currently based on the trajectory of all joints, to one less sensitive to the natural variation of human motion. For instance, analysing just the hand trajectory instead of the whole arm may be a way of reducing the variability. Either way, choosing the most adequate solution will only be possible using real human motion data, obtained by a motion tracker or a computer visions system like the Kinect from Microsoft.

The Self-Organizing mechanism combining Dynamic Neural Fields and the Kohonen SOM proved to be successful in the development of neural representations of the primitives. It was able to establish a stable feature map on a DNF, with clearly distinct populations representing each of the primitives. Moreover, by combining motion data with contextual information to form the input to the map, it is possible to evolve context dependent representations, consistently with the mirror neuron theory. This was shown in the second simulation phase, when the arm motion corresponding to the "grasp from above" gesture is represented by two different populations, corresponding to two different contexts (GWA and GNA).

While the use of self-organization for primitive encoding is comparable with the one presented by Thill and Ziemke (2010), the model presented in this thesis extends their work in several important ways: 1) it combines the SOM and the DNF paradigm; 2) it uses more realistic inputs, applying a transformation of joint motion data from temporal to a spatial dimension; 3) it introduces the back-propagation mechanism which guarantees that similar motion inputs may lead to distinct representations if additional cues justify.

Future work on the self-organizing mechanism may lead to deeper analytical and numerical analysis of the back-propagation mechanism, improving understanding of its dynamics. Other aspects of the model may also be improved, for instance by using different dynamical equations like the one described in Alecu et al. (2011). Alecu and colleagues claim that, with the field equations they propose, a newly demonstrated input is capable of destabilizing an existing activation in the dynamic field forming a new one in a different field location. This would avoid the "Algorithmic resets" between the

presentation of inputs, applied in this study.

The extension of the model to include the sequencing mechanism of Chapter 7 allowed the model to make predictions, anticipating the outcome of observed motions. This is a fundamental cognitive ability for any robot which is supposed to closely work with humans on shared tasks. The use of a single chaining mechanism to achieve this is only viable because the representations are dependent on context such as, for instance, the object type identified by the colour. This result is consistent with the chaining theory of MN system proposed by Fogassi et al. (2005), placing the model alongside with other models like the ones in Erlhagen et al. (2007); Chersi et al. (2011). However, the ability to autonomously generate the representations of the primitives from real data without prior specification of neuronal populations constitutes an important new step in developmental robotics (Asada et al., 2009).

One final note concerns the number of trials used in the simulations. 150 and 350 may seem a high number of trials when considering the use of the mechanism in real world scenarios. However, the main purpose of such a high number of trials was to demonstrate how the representations in the field tend to stabilize at specific field locations. From the results, one can see that after 80 trials in the first scenario and 100 in the second scenario the representations were already clearly distinct and the prediction error was low. Thus, it can be concluded that a successful learning was achieved well before the end of all trials. Still, it may be possible to further accelerate the formation of representations by using bi-dimensional DNFs. In one-dimensional fields the splitting of population into two often requires that all existing representations change location so that the two new populations may find enough neurons to recruit. The use of bi-dimensional fields might increase the flexibility of the generalization, by allowing the new representations to spread in different directions, recruiting still unspecific neurons and thus avoiding the displacement of all existing representations.

This page has been intentionally left blank.

Part VI

Conclusion and future work

Chapter 11

Conclusion and future work

The ability to learn is fundamental for the survival of humans and other animals since it constitutes a powerful adaptation tool allowing behaviour to be influenced by knowledge acquired in past experiences. Through learning we can take advantage from the surrounding environment by pursuing actions that proved valuable in some way in the past while avoiding those that have somehow undermined us. Perhaps even most importantly, the ability to learn allows knowledge to be exchanged among people, meaning that we can effectively learn with others experiences (without having to go through them) and build new knowledge upon them. This is possibly one of the main justifications for the powerful transformative ability we have over the environment.

Over the last years, robotics research has provided significant attention to the development of mechanisms that can furnish robots with the ability to learn. The hope is that through learning, robots can become more adaptable and therefore more useful and also that this will make working with them easier and enjoyable. The work presented in this thesis represents several advances towards this demanding goal. Seeking inspiration from known principles of the human brain function, it focused on the development of computational models that, framed in appropriate control architectures, could allow robots to learn tasks from demonstration. Two main topics were approached. One

was the learning of sequential tasks. The other was the formation of new neuronal representations in dynamic neural fields.

The first topic materialized in the development of a computational model that learns the potentially multiple orders in which a sequence of sub-goals can be completed. The model proved to be able to encode the sequential constraints defining the task through observation of a small number of demonstrations. Two additional mechanisms were also developed and successfully tested. The first is a rehearsal mechanism that avoids the need for repeated demonstrations. The second addresses the emergence of representations on a higher abstraction level, representing chunks of sub-goals. This was a first approach to the development of hierarchical task representations, a currently hotly debated research topic in neuro-computation and cognitive psychology.

The second research subject was addressed through the development of a self-organized mechanism based on Kohonen SOMs, capable of clustering sensory input data into abstract representations. The mechanism was employed for emerging representations of the individual motor primitives, described as sets of joint trajectories, which constitute components of goal-directed actions. Inspired by the chaining theory of mirror neurons, the newly formed representations were also employed as inputs to a sequence learning mechanism. As part of the control architecture, these newly established sequences of motor primitives can be used by the robot to predict the outcome of observed motor behaviour.

11.1 Future work

While the work has achieved its main objectives, there is large room for future developments in both research topics.

In terms of the sequence learning mechanism, future work should focus mostly on the development of hierarchical representations. While the formation of chunks was a first step towards that direction, it still remains to be seen how this can be extended to develop a multi-level hierarchical structure where the past/present representation of

sub-goals is present on multiple levels. This type of structure is expected to further increase the flexibility, that is context dependence, of task execution. Emergent hierarchical structure will also support learning since acquired sub-task representations may be integrated in different task contexts (Cooper and Shallice, 2000). Further research must also be conducted on which features, apart from prediction errors, may be used to segment sequences of sub-goals.

From the neuro-cognitive modelling perspective, it would be important to approach the occurrence of errors in the sequence learning model, beyond the prediction errors caused by the learning of multiple sequences. Since the seminal work of Lashley (1951) different error types have been extensively used to distinguish between different model classes of serial behaviour (see Henson, 1998; Cooper and Shallice, 2000; Botvinick and Plaut, 2004, for examples). A complete study of the type of errors that the presented model may explain could provide further hints on how it compares to real empirical data about typical errors occurring during human sequence learning.

In relation to the development of new task-relevant representations, work should proceed essentially about two lines. The most immediate change should be the transition from a uni-dimensional to a bi-dimensional map. This is expected to increase the flexibility of the mechanism, facilitating the emergence of representations and speeding up the learning. Learning speed is of particular importance for the second line of the work which will be the application of the learning mechanism with cinematic data acquired on-line. Given that this will necessarily require data obtained from human subjects, it will be important to reduce the number of required trials to avoid tiresome and increasingly error prone demonstrations.

There is also the possibility of using the Self-Organizing DNF to develop representations of other types of sensory input. A good candidate seems to be observed emotional facial expressions described as numerical patterns encoding facial features, namely the individual expression of eyes, eyebrows, mouth and nose (Ekman and Friesen, 1978). Emotion recognition could allow the robot to adapt its interaction dynamics according with the emotional responses of its human partners. From the neuro-cognitive point of

view, there are evidences pointing to a Mirror Neuron related neural mechanism underlying the recognition of emotions (van der Gaag et al., 2007). This mechanism seems to work through the same resonance principle supporting the recognition of goal-directed actions. Neuro-imaging studies supporting this view show that the observation of a facial expression corresponding to an emotion activates the same neuronal representations that would be active if the observer was feeling the emotion himself (Leslie et al., 2004). Thus, the development of emotional feature maps seems to be an appealing research subject, both from the robotics and from the computational modelling point of view.

Part VII

Appendices

Appendix A

Implementation parameters of Chapter 8

A.1 Sequence Learning Layer (SL)

A.1.1 Past field u^{pa}

A^{pa}	First amplitude term of the kernel	9.3
σ_A^{pa}	First σ term of the kernel	2.0
B^{pa}	Second amplitude term of the kernel	6.1
σ_B^{pa}	Second σ term of the kernel	3.0
τ^{pa}	Time constant of u^{pa}	2
H_{low}^{pa}	Lower baseline value of u^{pa}	-4.5
H_{high}^{pa}	Higher baseline value of u^{pa}	-0.5
H_{dec}^{pa}	Higher baseline value of u^{pa}	-4.5
C^{pa}	Gain parameter of visual input to u^{pa}	1.05
ζ_{stoch}^{pa}	Noise level of u^{pa}	0.3
λ^{pa}	Learning threshold of u^{pa}	5.3
T_1	Time constant of the h^{pa} decay with no error	20.0

T_2	Time constant of the h^{pa} decay - one error	35.0
T_3	Time constant of the h^{pa} decay - two errors	56.0

A.1.2 Present field u^{pr}

A^{pr}	Amplitude of the kernel	9.4
σ^{pr}	σ term of the kernel	7.0
w^{pr}_{inhib}	Global inhibition of the kernel	8.8
τ^{pr}	Time constant of u^{pr}	2.0
H^{pr}	Value of the resting level	-1.6
λ^{pr}	Learning threshold of u^{pr}	5.0
C^{pr}	Gain parameter of visual input to u^{pr}	17
ζ^{pr}_{stoch}	Noise level of u^{pr}	3.8
C^{lea}	Relative strength of excitation from u^{stm} during learning	0.9
C^{rec}	Relative strength of excitation from u^{stm} during recall	0.05

A.1.3 Learning rule

τ_a	Time constant of the learning rule	2.0
η	Gain of the decay term of the learning rule	0.1

A.2 Short-Term Memory Layer (STM)

A^{stm}	Amplitude of the kernel	2.5
σ^{stm}	σ term of the kernel	3.0
τ^{stm}	Time constant of u^{stm}	3.0
C^{stm}	Gain parameter of visual input to u^{stm}	7.0

ζ_{stoch}^{stm}	Noise level of u^{stm}	1.0
$\tau_{h^{stm}}$	Time constant of u^{stm}	0.2
H_{ini}^{stm}	Initial value of the resting level	-4.0

A.3 Task Monitoring Layer (TM)

A^{tm}	Amplitude of the kernel	1.5
σ^{tm}	σ term of the kernel	3.5
w_{inhib}^{pr}	Global inhibition of the kernel	0.5
τ^{tm}	Time constant of u^{tm}	1.0
C^{tm}	Gain parameter of input from u^{pr}	1.0
ζ_{stoch}^{tm}	Noise level of u^{tm}	1.0
H_{low}^{tm}	Lower value of resting level in u^{tm}	-8.0
H_{high}^{tm}	Higher value of resting level in u^{tm}	-1.0

A.4 High-level Cognitive Memory Layer (HCM)

A.4.1 Past field $u^{pa_{HL}}$

$A^{pa_{HL}}$	First amplitude term of the kernel	6.15
$\sigma_A^{pa_{HL}}$	First σ term of the kernel	2.5
$B^{pa_{HL}}$	Second amplitude term of the kernel	4.02
$\sigma_B^{pa_{HL}}$	Second σ term of the kernel	3.6
$\zeta_{stoch}^{pa_{HL}}$	Noise level of $u^{pa_{HL}}$	0.5
$\tau^{pa_{HL}}$	Time constant of $u^{pa_{HL}}$	2.0
$H^{pa_{HL}}$	Resting level of $u^{pa_{HL}}$	-0.9
$\lambda^{pa_{HL}}$	Learning threshold of $u^{pa_{HL}}$	5.2
$T_{h^{pa_{HL}}}$	Time constant of the $h^{pa_{HL}}$ decay	50.0

A.4.2 Present field $u^{pr_{HL}}$

$A^{pr_{HL}}$	Amplitude of the kernel	2.0
$\sigma^{pr_{HL}}$	σ term of the kernel	2.9
$\zeta_{stoch}^{pr_{HL}}$	Noise level of $u^{pr_{HL}}$	3.0
w_{inhib}^{pr}	Global inhibition of the kernel	1.4
$\tau^{pr_{HL}}$	Time constant of $u^{pr_{HL}}$	2.0
$H^{pr_{HL}}$	Value of the resting level	-1.6
$\Lambda_{low}^{pr_{HL}}$	Lower value of the learning threshold $\lambda^{pr_{HL}}$	3.5
$\Lambda_{high}^{pr_{HL}}$	Higher value of the learning threshold $\lambda^{pr_{HL}}$	10.0

A.4.3 Learning rules

$\tau_a^{pa_{HL}}$	Time constant of the learning rule applied to $\underset{pa \to pa_{HL}}{a}$	10.0
$\tau_a^{pr_{HL}}$	Time constant of the learning rule applied to $\underset{pr_{HL} \to pr}{a}$	10.0

Appendix B

Implementation parameters of Chapter 10

B.1 Self-organizing field u^{som}

A^{som}	Amplitude term of the u^{som} kernel	0.9
σ_A^{som}	σ term of the u^{som} kernel	10.5
ζ_{stoch}^{som}	Noise level of u^{som}	1.8
C^{som}	Gain applied to input z^{som}	10.2
w_{inhib}^{som}	Global inhibition of the u^{som} kernel	0.5
h^{som}	Resting level of u^{som}	0.7
τ^{som}	Time constant of u^{som}	$7.0 \times dt$

B.1.1 Self-organizing learning rule

$\tau^{a_{som}}$	Time constant of the SOM learning rule	$600.0 \times dt$

B.2 Sequence Learning Layer (SL)

B.2.1 Past field u^{pa}

A^{pa}	First amplitude term of the kernel	4.5
σ_A^{pa}	First σ term of the kernel	2.05
B^{pa}	Second amplitude term of the kernel	2.9
σ_B^{pa}	Second σ term of the kernel	2.87
ζ_{stoch}^{pa}	Noise level of u^{pa}	0.5
τ^{pa}	Time constant of field variation	$10.0 \times dt$
H_{low}^{pa}	Lower baseline value of the resting level function h^{pa}	-1.0
H_{dec}^{pa}	Convergence value of the resting level function h^{pa}	-6.5
C^{pa}	Gain parameter of u^{som} input	0.04
λ^{pa}	Learning threshold	1.6
T_1	Time constant of the h^{pa} decay	$10.0 \times dt$

B.2.2 Present field u^{pr}

A^{pr}	Amplitude of the kernel	5.0
σ^{pr}	σ term of the kernel	8.0
w_{inhib}^{pr}	Global inhibition of the kernel	4.0
ζ_{stoch}^{pr}	Noise level of u^{pr}	3.0
τ^{pr}	Time constant of field variation	$10.0 \times dt$
H^{pr}	Value of the resting level	-3.0
λ^{pr}	Learning threshold	8.0
C^{pr}	Gain parameter of u^{som} input	13

B.2.3 Learning rule between past and present fields

τ_a	Time constant of the learning rule	$5.0 \times dt$
η	Gain of the decay term of the learning rule	0.1
λ^{som}	Threshold for propagation of activity in u^{som} to u^{pr}	4.0

This page has been intentionally left blank.

Bibliography

E. E. Aksoy, B. Dellen, M. Tamosiunaite, and F. Worgotter, "Execution of a dual-object (pushing) action with semantic event chains," *2011 11th IEEE-RAS International Conference on Humanoid Robots*, pp. 576–583, 2011.

L. Alecu, H. Frezza-Buet, and F. Alexandre, "Can self-organisation emerge through dynamic neural fields computation?" *Connection Science*, vol. 23, no. 1, pp. 1–31, 2011.

J. Aleotti, S. Caselli, and M. Reggiani, "Leveraging on a virtual environment for robot programming by demonstration," *Robotics and Autonomous Systems*, vol. 47, no. 2-3, pp. 153–161, 2004.

A. Alissandrakis, C. L. Nehaniv, and K. Dautenhahn, "Imitation with ALICE: learning to imitate corresponding actions across dissimilar embodiments," *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, vol. 32, no. 4, pp. 482–496, 2002.

S.-i. Amari, "Dynamic of pattern formation in lateral inhibition type neural fields," *Biological Cybernetics*, vol. 27, pp. 77–87, 1977.

——, "Topographic organization of nerve fields," *Bulletin of Mathematical Biology*, vol. 42, no. 3, pp. 339–364, 1980.

——, "Field theory of self-organizing neural nets," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. SMC-13, no. 5, pp. 741–748, 1983.

R. Amit and M. J. Matarić, "Learning movement sequences from demonstration," *Proceedings 2nd International Conference on Development and Learning. ICDL 2002*, pp. 203–208, 2002.

B. D. Argall, M. Veloso, S. Chernova, and B. Browning, "A survey of robot learning from demonstration," *Robotics and Autonomous Systems*, vol. 57, pp. 469–483, 2009.

W. F. Asaad, G. Rainer, and E. Miller, "Task-specific neural activity in the primate prefrontal cortex," *Journal of Neurophysiology*, vol. 84, no. 1, pp. 451–459, 2000.

M. Asada, K. Hosoda, Y. Kuniyoshi, H. Ishiguro, T. Inui, Y. Yoshikawa, M. Ogino, and C. Yoshida, "Cognitive Developmental Robotics: A Survey," *IEEE Transactions on Autonomous Mental Development*, vol. 1, no. 1, pp. 12–34, 2009.

G. Barreto, A. Araújo, and H. Ritter, "Self-Organizing Feature Maps for Modeling and Control of Robotic Manipulators," no. ii, pp. 407–450, 2003.

H. Bekkering, A. Wohlschläger, and M. Gattis, "Imitation of gestures in children is goal-directed." *The Quarterly journal of experimental psychology. A, Human experimental psychology*, vol. 53, no. 1, pp. 153–64, 2000.

M. Bennewitz, F. Faber, D. Joho, M. Schreiber, and S. Behnke, "Towards a humanoid museum guide robot that interacts with multiple persons," in *5th IEEE-RAS International Conference on Humanoid Robots*. IEEE, 2005, pp. 418–423.

D. C. Bentivegna, C. G. Atkeson, and G. Cheng, "Learning tasks from observation and practice," *Robotics and Autonomous Systems*, vol. 47, no. 2-3, pp. 163–169, 2004.

D. Bentivegna and C. Atkeson, "A framework for learning from observation using primitives," *RoboCup 2002: Robot Soccer World Cup VI*, pp. 2–7, 2003.

D. P. Bertsekas, *Dynamic programming and optimal control*. Athena Scientific, 1995.

E. Bicho, *Dynamic approach to behavior-based robotics: design, specification, analysis, simulation and implementation*. Shaker Verlag, Aachen, 2000.

E. Bicho and G. Schöner, "The dynamic approach to autonomous robotics demonstrated on a low-level vehicle platform," *Robotics and autonomous systems*, vol. 21, pp. 23–35, 1997.

E. Bicho, P. Mallet, and G. Schöner, "Using attractor dynamics to control autonomous vehicle motion," in *IECON '98. Proceedings of the 24th Annual Conference of the IEEE Industrial Electronics Society (Cat. No.98CH36200)*, vol. 2. IEEE, 1998, pp. 1176–1181.

——, "Target Representation on an Autonomous Vehicle with Low-Level Sensors," *I. J. Robotic Res.*, vol. 19, no. 5, pp. 424–447, 2000.

E. Bicho, L. Louro, N. Hipólito, and W. Erlhagen, "A dynamic neural field architecture for flexible and fluent human-robot interaction," in *Proceedings of the 2008 International Conference on Cognitive Systems*, Karlsruhe, Germany, 2008, pp. 179–185.

——, "A dynamic field approach to goal inference and error monitoring for human-robot interaction," *Proceedings of the 2009 International Symposium on New Frontiers in Human-Robot Interaction . AISB (Adaptive & Emergent Behaviour & Complex Systems) Convention*, pp. 31–37, 2009.

E. Bicho, L. Louro, and W. Erlhagen, "Integrating verbal and nonverbal communication in a dynamic neural field architecture for humanrobot interaction," *Frontiers in Neurorobotics*, vol. 4, pp. 135–164, 2010.

E. Bicho, W. Erlhagen, L. Louro, and E. C. e Silva, "Neuro-cognitive mechanisms of decision making in joint action: a human-robot interaction study." *Human Movement Science*, vol. 30, no. 5, pp. 846–868, 2011.

E. Bicho, W. Erlhagen, L. Louro, E. C. e. Silva, R. M. Silva, and N. Hipólito, "A dynamic field approach to goal inference, error detection and anticipatory action selection in human-robot collaboration," in *New Frontiers in Human-Robot Interaction*, K. Dautenhahn and J. Sanders, Eds. John Benjamins Publishing Company, 2011, pp. 135–164.

E. Bicho, W. Erlhagen, E. Sousa, L. Louro, N. Hipólito, E. Costa e Silva, R. M. Silva, F. Ferreira, T. Machado, M. Hulstijn, Y. Maas, E. D. Bruijn, R. H. Cuijpers, R. Newman-Norlund, H. V. Schie, R. Meulenbroek, and H. Bekkering, "The Power of Prediction : Robots that Read Intentions," in *Proc. International Conference on Intelligent Robots and Systems*, 2012, pp. 5458–5459.

A. G. Billard, S. Calinon, and F. Guenter, "Discriminative and adaptive imitation in uni-manual and bi-manual tasks," *Robotics and Autonomous Systems*, vol. 54, no. 5, pp. 370–384, 2006.

A. G. Billard, S. Calinon, R. Dillmann, and S. Schaal, "Robot programming by demonstration," in *Handbook of Robotics*, B. Siciliano and O. Khatib, Eds. Secaucus, NJ, USA: Springer, 2008, pp. 1371–1394.

D. Blank, D. Kumar, L. Meeden, and J. B. Marshall, "Bringing up robot: Fundamental mechanisms for creating a self-motvated, self-organizing architecture," *Cybernetics and Systems*, vol. 36, no. 2, pp. 125–150, 2005.

M. M. Botvinick, "Hierarchical models of behavior and prefrontal function." *Trends in Cognitive Sciences*, vol. 12, no. 5, pp. 201–208, 2008.

——, "Hierarchical reinforcement learning and decision making." *Current Opinion in Neurobiology*, pp. 1–7, 2012.

M. M. Botvinick and D. C. Plaut, "Representing task context: proposals based on a connectionist model of action." *Psychological Research*, vol. 66, no. 4, pp. 298–311, 2002.

——, "Doing without schema hierarchies: a recurrent connectionist approach to normal and impaired routine sequential action." *Psychological Review*, vol. 111, no. 2, pp. 395–429, 2004.

——, "Short-term memory for serial order: A recurrent neural network model," *Psychological Review*, vol. 113, no. 2, pp. 201–232, 2006.

M. M. Botvinick, Y. Niv, and A. C. Barto, "Hierarchically organized behavior and its neural foundations: a reinforcement learning perspective." *Cognition*, vol. 113, no. 3, pp. 262–80, 2009.

C. Breazeal, D. Buchsbaum, J. Gray, D. Gatenby, and B. Blumberg, "Learning From and About Others: Towards Using Imitation to Bootstrap the Social Understanding of Others by Robots," *Artificial Life*, vol. 11, pp. 31–62, 2005.

C. Breazeal, M. Berlin, R. A. Brooks, J. Gray, and A. L. Thomaz, "Using perspective taking to learn from ambiguous demonstrations," *Robotics and Autonomous Systems*, vol. 54, no. 5, pp. 385–393, 2006.

C. Breazeal, J. Gray, and M. Berlin, "An Embodied Cognition Approach to Mindreading Skills for Socially Intelligent Robots," *The International Journal of Robotics Research*, vol. 28, no. 5, pp. 656–680, 2009.

R. A. Brooks, ""New Approaches to Robotics"," *Science*, vol. 253, pp. 1227–1232, 1991.

G. D. A. Brown, T. Preece, and C. Hulme, "Oscillator-based memory for serial order," *Psychological Review*, vol. 107, p. 127, 2000.

K. Browne and M. N. Nicolescu, "Learning to Generalize from Demonstrations," *Cybernetics and Information Technologies*, vol. 12, no. 3, pp. 27–38, 2012.

G. Buccino, S. Vogt, A. Ritzl, G. R. Fink, K. Zilles, H.-j. Freund, and G. Rizzolatti, "Neural Circuits Underlying Imitation Learning of Hand Actions," *Neuron*, vol. 42, no. 2, pp. 323–334, 2004.

N. Burgess, "Toward a network model of the articulatory loop," *Journal of Memory and Language*, vol. 31, no. 4, pp. 429–460, 1992.

N. Burgess and G. J. Hitch, "Memory for Serial Order: A Network Model of the Phonological Loop and its Timing," *Psychological Review*, vol. 106, pp. 551–581, 1999.

——, "A revised model of short-term memory and long-term learning of verbal sequences," *Journal of Memory and Language*, vol. 55, no. 4, pp. 627–652, 2006.

J. Butterfield, S. Osentoski, G. Jay, and O. C. Jenkins, "Learning from demonstration using a multi-valued function regressor for time-series data," *2010 10th IEEE-RAS International Conference on Humanoid Robots*, pp. 328–333, 2010.

S. Calinon and A. G. Billard, "Incremental learning of gestures by imitation in a humanoid robot," in *Proceedings of the ACM/IEEE International Conference on Human-Robot Interaction*. ACM, 2007, p. 262.

S. Calinon, F. Guenter, and A. G. Billard, "Goal-Directed Imitation in a Humanoid Robot," in *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*, no. April. IEEE, 2005, pp. 299–304.

——, "On Learning, Representing, and Generalizing a Task in a Humanoid Robot," *IEEE Transactions on Systems, Man and Cybernetics, Part B (Cybernetics)*, vol. 37, no. 2, pp. 286–298, 2007.

G. A. Carpenter, S. Grossberg, and D. B. Rosen, "Fuzzy ART: Fast stable learning and categorization of analog patterns by an adaptive resonance system," *Neural Networks*, vol. 4, no. 6, pp. 759–771, 1991.

G. Chang and D. Kulic, "Robot task learning from demonstration using Petri nets," *RO-MAN, 2013 IEEE*, 2013.

R. Chatila, "Deliberation and reactivity in autonomous mobile robots," *Robotics and Autonomous Systems*, vol. 16, no. 2-4, pp. 197–211, 1995.

C. Chen, "Hidden Markov model approach to skill learning and its application to telerobotics," *IEEE Transactions on Robotics and Automation*, vol. 10, no. 5, pp. 621–631, 1994.

T. L. Chen, A. Jain, and C. C. Kemp, "Towards an assistive robot that autonomously performs bed baths for patient hygiene," in *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, no. 1. IEEE, 2010, pp. 319–324.

F. Chersi, L. Fogassi, L. Bonini, G. Rizzolatti, and P. F. Ferrari, "Modeling intentional neuronal chains in parietal and premotor cortex," *Soc. Neurosci. Abstr*, 2007.

F. Chersi, P. F. Ferrari, and L. Fogassi, "Neuronal chains for actions in the parietal lobe: a computational model." *PloS one*, vol. 6, no. 11, p. e27652, 2011.

A. Coates, P. Abbeel, and A. Y. Ng, "Learning for control from multiple demonstrations," in *Proceedings of the 25th International Conference on Machine Learning - ICML '08*. New York, New York, USA: ACM Press, 2008, pp. 144–151.

R. Conrad, "Order error in immediate recall of sequences," *Journal of Verbal Learning and Verbal Behavior*, vol. 4, no. 3, pp. 161–169, 1965.

R. P. Cooper and T. Shallice, "Contention scheduling and the control of routine activities," *Cognitive Neuropsychology*, vol. 17, no. 4, pp. 297–338, 2000.

——, "Hierarchical schemas and goals in the control of sequential behavior." *Psychological Review*, vol. 113, no. 4, pp. 887–916; discussion 917–931, 2006.

E. Costa e Silva, M. F. P. Costa, J. P. Araújo, D. Machado, L. Louro, W. Erlhagen, and E. Bicho, "Towards human-like bimanual actions in anthropomorphic robots: a nonlinear optimization approach," *to appear in Applied Mathematics & Information Sciences (AMIS 2014)*, 2014.

K. Dautenhahn, C. L. Nehaniv, M. Walters, B. Robins, H. Kose-Bagci, N. A. Mirza, and M. Blow, "KASPAR - a minimally expressive humanoid robot for human-robot interaction research," *Applied Bionics and Biomechanics*, vol. 6, no. 3, pp. 369–397, 2009.

G. Deboeck and T. Kohonen, *Visual explorations in finance: with self-organizing maps*, G. Deboeck and T. Kohonen, Eds. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 1998.

J. Decety and J. a. Sommerville, "Shared representations between self and other: a social cognitive neuroscience view," *Trends in Cognitive Sciences*, vol. 7, no. 12, pp. 527–533, 2003.

R. Dillmann, "Teaching and learning of robot tasks via observation of human performance," *Robotics and Autonomous Systems*, vol. 47, no. 2-3, pp. 109–116, 2004.

M. Ehrenmann, R. Zöllner, O. Rogalla, and R. Dillmann, "Programming service tasks in household environments by human demonstration," in *Proceedings. 11th IEEE International Workshop on Robot and Human Interactive Communication*. IEEE, 2002, pp. 460–467.

P. Ekman and W. Friesen, "Facial action coding system: A technique for the measurement of facial movement. Palo Alto," in *Differences among unpleasant feelings. Motivation and Emotion*, P. C. E. . C. A. Smith, Ed. Palo Alto: CA: Consulting Psychologists Press (1988), 1978, pp. 271–302.

S. Ekvall and D. Kragic, "Learning Task Models from Multiple Human Demonstrations," in *ROMAN 2006 - The 15th IEEE International Symposium on Robot and Human Interactive Communication*. IEEE, 2006, pp. 358–363.

R. J. Elliott, L. Aggoun, and J. B. Moore, *Hidden Markov Models*, ser. Stochastic Modelling and Applied Probability. New York, NY: Springer New York, 1995, vol. 29.

J. L. Elman, "Finding structure in time," *Cognitive Science*, vol. 14, pp. 179–211, 1990.

C. Engels and G. Schöner, "Dynamic fields endow behavior-based robots with representations," *Robotics and Autonomous Systems*, vol. 14, pp. 55–77, 1995.

W. Erlhagen and E. Bicho, "The dynamic neural field approach to cognitive robotics." *Journal of Neural Engineering*, vol. 3, no. 3, pp. R36–R54, 2006.

W. Erlhagen, A. Mukovskiy, and E. Bicho, "A dynamic model for action understanding and goal-directed imitation," *Brain Research*, vol. 1083, no. 1, pp. 174–188, 2006.

W. Erlhagen, A. Mukovskiy, E. Bicho, G. Panin, C. Kiss, A. Knoll, H. T. Van Schie, and H. Bekkering, "Goal-directed imitation for robots: A bio-inspired approach to action understanding and skill learning," *Robotics and Autonomous Systems*, vol. 54, pp. 353–360, 2006.

W. Erlhagen, A. Mukovskiy, F. Chersi, and E. Bicho, "On the development of intention understanding for joint action tasks," in *IEEE 6th International Conference on Development and Learning. ICDL*, 2007, pp. 140–145.

W. K. Estes, "An associative basis for coding and organization in memory," *Coding processes in human memory*, 1972.

——, "Processes of memory loss, recovery, and distortion." *Psychological Review*, vol. 104, no. 1, pp. 148–69, 1997.

D. R. Euston, M. Tatsuno, and B. L. McNaughton, "Fast-forward playback of recent memory sequences in prefrontal cortex during sleep." *Science (New York, N.Y.)*, vol. 318, no. 5853, pp. 1147–50, 2007.

F. Faber, M. Bennewitz, C. Eppner, A. Gorog, C. Gonsior, D. Joho, M. Schreiber, and S. Behnke, "The humanoid museum tour guide Robotinho," *RO-MAN 2009 - The 18th IEEE International Symposium on Robot and Human Interactive Communication*, pp. 891–896, 2009.

C. Faubel and G. Schöner, "Fast learning to recognize objects: Dynamic Fields in label-feature spaces," *... Conference on Development and Learning*, 2006.

——, "2008 Special Issue: Learning to recognize objects on the fly: A neurally based dynamic field approach," *Neural Networks*, vol. 21, no. 4, pp. 562–576, 2008.

W. A. Fellenz and J. G. Taylor, "Establishing retinotopy by lateral-inhibition type homogeneous neural fields," *Neurocomputing*, vol. 48, no. 1-4, pp. 313–322, 2002.

F. Ferreira, W. Erlhagen, and E. Bicho, "A Dynamic Field Model of Ordinal and Timing Properties of Sequential Events," in *ICANN 2011, Part II, Lecture Notes in Computer Science*, T. Honkela, W. Duch, M. Girolami, and S. Kaski, Eds., vol. 6792, 2011, pp. 323–332.

F. Ferreira, W. Erlhagen, E. Sousa, and E. Bicho, "Learning a musical sequence by observation:A robotics implementation of adynamic neural field model," in *International*

Conference on Developmentand Learning and on Epigenetic Robotics - ICDL 2014, 2014.

R. E. Fikes and N. J. Nilsson, "Strips: A New Approach to the Application of Theorem Proving to Problem Solving," *Artificial Intelligence*, vol. 2, no. 3-4, pp. 189–208, 1971.

L. Fogassi, P. F. Ferrari, B. Gesierich, S. Rozzi, F. Chersi, and G. Rizzolatti, "Parietal lobe: from action organization to intention understanding," *Science*, vol. 308, pp. 662–666, 2005.

T. Fong, I. Nourbakhsh, and K. Dautenhahn, "A survey of socially interactive robots," *Robotics and Autonomous Systems*, vol. 42, pp. 143–166, 2003.

H. Friedrich, S. Mnch, R. Dillmann, S. Bocionek, and M. Sassin, "Robot programming by Demonstration (RPD): Supporting the induction by human interaction," *Machine Learning*, vol. 23, no. 2-3, pp. 163–189, 1996.

J. M. Fuster, "Upper processing stages of the perception-action cycle." *Trends in Cognitive Sciences*, vol. 8, no. 4, pp. 143–5, 2004.

——, "The cognit: a network model of cortical representation," *International Journal of Psychophysiology*, vol. 60, no. 2, pp. 125–132, 2006.

V. Gallese, L. Fadiga, L. Fogassi, and G. Rizzolatti, "Action recognition in the premotor cortex." *Brain : A Journal of Neurology*, vol. 119 (Pt 2, pp. 593–609, 1996.

A. Garland and N. Lesh, "Learning hierarchical task models by demonstration," *Technical Report TR2003-01, Mitsubishi Electric Research Laboratories*, 2003.

A. Genovesio, P. J. Brasted, and S. P. Wise, "Representation of future and previous spatial goals by separate neural populations in prefrontal cortex." *The Journal of Neuroscience : The Official Journal of the Society for Neuroscience*, vol. 26, no. 27, pp. 7305–7316, 2006.

J. Grèzes, J. Armony, J. Rowe, and R. Passingham, "Activations related to mirror and canonical neurones in the human brain: an fMRI study," *NeuroImage*, vol. 18, no. 4, pp. 928–937, 2003.

S. Grossberg, "Behavioral contrast in short term memory: Serial binary memory models or parallel continuous memory models?" *Journal of Mathematical Psychology*, vol. 17, no. 3, pp. 199–219, 1978.

S. Gulati, K. Richmond, C. Flesher, B. P. Hogan, A. Murarka, G. Kuhlmann, M. Sridharan, W. C. Stone, and P. T. Doran, "Toward autonomous scientific exploration of ice-covered lakesField experiments with the ENDURANCE AUV in an Antarctic Dry Valley," in *2010 IEEE International Conference on Robotics and Automation*. IEEE, 2010, pp. 308–315.

B. M. Hard, S. C. Lozano, and B. Tversky, "Hierarchical encoding of behavior: translating perception into action." *Journal of experimental psychology. General*, vol. 135, no. 4, pp. 588–608, 2006.

B. M. Hard, B. Tversky, and D. S. Lang, "Making sense of abstract events: building event schemas." *Memory & Cognition*, vol. 34, no. 6, pp. 1221–35, 2006.

S. Harnad, "The symbol grounding problem," *Physica D: Nonlinear Phenomena*, vol. 42, no. 1-3, pp. 335–346, 1990.

T. Hartley and G. Houghton, "A linguistically constrained model of short-term memory for nonwords," *Journal of Memory and Language*, vol. 35, pp. 1–31, 1996.

S. Haykin, *Neural networks and learning machines*, 2009.

D. O. Hebb, *The organization of behavior*. Cambridge, MA, USA: Lawrence Erlbaum Associates Inc, 1949.

R. N. A. Henson, "Short-term memory for serial order: the Start-End Model." *Cognitive Psychology*, vol. 36, no. 2, pp. 73–137, 1998.

R. N. a. Henson, D. G. Norris, M. Page, and A. D. Baddeley, "Unchained Memory: Error Patterns Rule out Chaining Models of Immediate Serial Recall," *The Quarterly Journal of Experimental Psychology A*, vol. 49, no. 1, pp. 80–115, 1996.

G. Houghton, "The problem of serial order: a neural network model of sequence learning and recall," *Academic Press Cognitive Science Series*, pp. 287–319, 1990.

G. Hovland, P. Sikka, and B. McCarragher, "Skill acquisition from human demonstration using a hidden Markov model," in *Proceedings of IEEE International Conference on Robotics and Automation*, vol. 3. IEEE, 1996, pp. 2706–2711.

M. W. Howard and M. J. Kahana, "A Distributed Representation of Temporal Context," *Journal of Mathematical Psychology*, vol. 46, no. 3, pp. 269–299, 2002.

M. W. Howard, B. Jing, V. a. Rao, J. P. Provyn, and A. V. Datey, "Bridging the gap: transitive associations between items presented in similar temporal contexts." *Journal of experimental psychology. Learning, memory, and cognition*, vol. 35, no. 2, pp. 391–407, 2009.

D. Hubel and T. Wiesel, "Functional architecture of macaque monkey visual cortex," *Proceedings of the Royal Society of London. Series B, Biological Sciences*, vol. 198, no. 1130, pp. 1–59, 1977.

C. Hulme, "High- and low-frequency words are recalled equally well in alternating lists: Evidence for associative effects in serial recall," *Journal of Memory and Language*, vol. 49, no. 4, pp. 500–518, 2003.

M. J. Hurlstone, G. J. Hitch, and A. D. Baddeley, "Memory for Serial Order Across Domains: An Overview of the Literature and Directions for Future Research." *Psychological Bulletin*, 2013.

M. Iacoboni, I. Molnar-Szakacs, V. Gallese, G. Buccino, J. C. Mazziotta, and G. Rizzolatti, "Grasping the intentions of others with one's own mirror neuron system." *PLoS biology*, vol. 3, no. 3, p. e79, 2005.

K. Ishii and S. Nishida, "A Self-Organizing Map Based Navigation System for an Underwater Robot," no. April, pp. 4466–4471, 2004.

D. Ji and M. a. Wilson, "Coordinated memory replay in the visual cortex and hippocampus during sleep." *Nature Neuroscience*, vol. 10, no. 1, pp. 100–7, 2007.

M. Johnsson and C. Balkenius, "Sense of Touch in Robots With Self-Organizing Maps," *IEEE Transactions on Robotics*, vol. 27, no. 3, pp. 498–507, 2011.

M. I. Jordan, "Serial Order: A Parallel Distributed Processing Approach," California University, San Diego, La Jolla Institute for Cognitive Science, San Diego: University of California, Institute for Cognitive Science, Tech. Rep., 1986.

M. J. Kahana and J. Jacobs, "Interresponse times in serial recall: effects of intraserial repetition." *Journal of experimental psychology. Learning, memory, and cognition*, vol. 26, no. 5, pp. 1188–97, 2000.

T. Kanda, R. Sato, N. Saiwaki, and H. Ishiguro, "A Two-Month Field Trial in an Elementary School for Long-Term HumanRobot Interaction," *IEEE Transactions on Robotics*, vol. 23, no. 5, pp. 962–971, 2007.

S. Kaski, J. Kangas, and T. Kohonen, "Bibliography of self-organizing map (SOM) papers: 19811997," *Neural Computing Surveys*, vol. 1, pp. 102–350, 1998.

J. Kober and J. Peters, "Learning motor primitives for robotics," in *2009 IEEE International Conference on Robotics and Automation*. IEEE, 2009, pp. 2112–2118.

T. Kohonen, "Self-organized formation of topologically correct feature maps," *Biological Cybernetics*, vol. 43, no. 1, pp. 59–69, 1982.

——, "The self-organizing map," *Proceedings of the IEEE*, vol. 78, no. 9, pp. 1464–1480, 1990.

T. Kohonen, S. Kaski, K. Lagus, J. Salojarvi, J. Honkela, V. Paatero, and A. Saarela, "Self organization of a massive document collection." *IEEE transactions on neural networks / a publication of the IEEE Neural Networks Council*, vol. 11, no. 3, pp. 574–85, 2000.

G. Konidaris, S. Kuindersma, R. Grupen, and A. G. Barto, "Robot learning from demonstration by constructing skill trees," *The International Journal of Robotics Research*, vol. 31, no. 3, pp. 360–375, 2011.

L. Koski, A. Wohlschläger, H. Bekkering, P. Woods, M.-c. Dubeau, J. C. Mazziotta, and M. Iacoboni, "Modulation of Motor and Premotor Activity during Imitation of Target-directed Actions," pp. 847–855, 2002.

A. Krogh and J. A. Hertz, "A simple weight decay can improve generalization," *Advances in Neural Information Processing Systems*, vol. 4, pp. 950–957, 1992.

D. Kulic and Y. Nakamura, "Incremental learning of human behaviors using hierarchical hidden Markov models," in *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2010, pp. 4649–4655.

D. Kulic, C. Ott, D. Lee, J. Ishikawa, and Y. Nakamura, "Incremental learning of full body motion primitives and their sequencing through human motion observation," *The International Journal of Robotics Research*, vol. 31, no. 3, pp. 330–345, 2012.

Y. Kuniyoshi, M. Inaba, and H. Inoue, "Learning by watching: extracting reusable task knowledge from visual observation of human performance," *IEEE Transactions on Robotics and Automation*, vol. 10, no. 6, pp. 799–822, 1994.

C. Kunz, C. Murphy, H. Singh, C. Pontbriand, R. A. Sohn, S. Singh, T. Sato, C. Roman, K.-i. Nakamura, M. Jakuba, R. Eustice, R. Camilli, and J. Bailey, "Toward extraplanetary under-ice exploration: Robotic steps in the Arctic," *Journal of Field Robotics*, vol. 26, no. 4, pp. 411–429, 2009.

C. A. Kurby and J. M. Zacks, "Segmentation in the perception and memory of events." *Trends in Cognitive Sciences*, vol. 12, no. 2, pp. 72–79, 2008.

C. R. Laing, W. C. Troy, B. Gutkin, and G. B. Ermentrout, "Multiple bumps in a neuronal model of working memory," *SIAM Journal on Applied Mathematics*, vol. 62, no. 1, pp. 62–97, 2002.

K. Lashley, "The problem of serial order in behavior," *Cerebral Mechanisms in Behaviour*, pp. 112–142, 1951.

C. L. Lee and W. K. Estes, "Order and position in primary memory for letter strings," *Journal of Verbal Learning and Verbal Behavior*, vol. 16, no. 4, pp. 395–418, 1977.

——, "Item and order information in short-term memory: Evidence for multilevel perturbation processes." *Journal of Experimental Psychology: Human Learning & Memory*, vol. 7, no. 3, pp. 149–169, 1981.

K. R. Leslie, S. H. Johnson-Frey, and S. T. Grafton, "Functional imaging of face and hand imitation: towards a motor theory of empathy." *NeuroImage*, vol. 21, no. 2, pp. 601–7, 2004.

V. Levenshtein, "Binary codes capable of correcting deletions, insertions and reversals," *Soviet Physics Doklady*, vol. 10, pp. 707–710, 1966.

S. Lewandowsky and B. B. Murdock, "Memory for serial order," *Psychological Review*, vol. 96, pp. 25–57, 1989.

S. Lewandowsky, G. D. A. Brown, T. Wright, and L. M. Nimmo, "Timeless memory: Evidence against temporal distinctiveness models of short-term memory for serial order," *Journal of Memory and Language*, vol. 54, no. 1, pp. 20–38, 2006.

S. Lewandowsky, L. M. Nimmo, and G. D. A. Brown, "When temporal isolation benefits memory for serial order," *Journal of Memory and Language*, vol. 58, no. 2, pp. 415–428, 2008.

T. Machado, T. Malheiro, S. Monteiro, E. Bicho, and W. Erlhagen, "Transportation of long objects in unknown cluttered environments by a team of robots: A dynamical systems approach," in *2013 IEEE International Symposium on Industrial Electronics*. IEEE, 2013, pp. 1–6.

L. Maddalena and A. Petrosino, "A self-organizing approach to background subtraction for visual surveillance applications." *IEEE transactions on image processing : a publication of the IEEE Signal Processing Society*, vol. 17, no. 7, pp. 1168–77, 2008.

M. Matarić, "Situated Robotics," in *Encyclopedia of cognitive science*, L. Nadel, Ed. Nature Publishers Group, Macmillian Reference, 2002.

J. L. McClelland, R. C. O'reilly, and B. L. McNaughton, "Why There Are Complementary Learning Systems in the Hippocampus and Neocortex: Insights from the Successes

and Failures of Connectionist Models of Learning and Memory," *Psychological Review*, vol. 102, no. 3, pp. 419–457, 1995.

D. McColl and G. Nejat, "Meal-Time with a Socially Assistive Robot and Older Adults at a Long-term Care Facility," *Journal of Human-Robot Interaction*, vol. 2, no. 1, pp. 152–171, 2013.

W. S. McCulloch and W. Pitts, "A logical calculus of the ideas immanent in nervous activity," *Bulletin of Mathematical Biology*, vol. 5, pp. 115–133, 1943.

G. Metta, P. Fitzpatrick, and L. Natale, "YARP: Yet Another Robot Platform," *International Journal on Advanced Robotics Systems*, vol. 3, pp. 43–48, 2006.

G. Miller, E. Galanter, and K. Pribram, *Plans and the structure of behavior.* New York, NY, USA: Henry Holt & Co, 1960, vol. 62, no. 6.

T. Minato, M. Shimada, H. Ishiguro, and S. Itakura, "Development of an android robot for studying human-robot interaction," *Innovations in Applied Artificial Intelligence, Lecture Notes in Computer Science*, vol. 3029, pp. 424–434, 2004.

J. Miura, K. Iwase, and Y. Shirai, "Interactive Teaching of a Mobile Robot," in *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*. Barcelona: IEEE, 2005, pp. 3378–3383.

S. Monteiro and E. Bicho, "A dynamical systems approach to behavior-based formation control," in *Proc. IEEE International Conference on Robotics and Automation ICRA*, vol. 3, 2002, pp. 2606–2611.

——, "Robot formations: Robots allocation and leader-follower pairs," in *Proc. IEEE International Conference on Robotics and Automation ICRA*, 2008, pp. 3769–3775.

S. Monteiro, M. Vaz, and E. Bicho, "Attractor dynamics generates robot formation: from theory to implementation," in *Proc. IEEE International Conference on Robotics and Automation ICRA '04*, vol. 3, 2004, pp. 2582–2586.

T. Murata, "Petri nets: Properties, analysis and applications," *Proceedings of the IEEE*, vol. 77, no. 4, pp. 541–580, 1989.

B. B. Murdock, "TODAM 2: A model for the storage and retrieval of item, associative, and serial-order information," *Psychological Review*, vol. 100, pp. 183–203, 1993.

——, "Developing TODAM: three models for serial-order information." *Memory & Cognition*, vol. 23, no. 5, pp. 631–45, 1995.

R. Murphy, S. Takadoro, D. Nardi, A. Jacoff, P. Fiorini, H. Choset, and A. Erkmen, "Search and rescue robotics," in *Springer Handbook of Robotics*, B. Siciliano and O. Khatib, Eds. Springer Berlin Heidelberg, 2008, pp. 1151–1173.

J. Nakanishi, J. Morimoto, G. Endo, G. Cheng, S. Schaal, and M. Kawato, "Learning from demonstration and adaptation of biped locomotion," *Robotics and Autonomous Systems*, vol. 47, no. 2-3, pp. 79–91, 2004.

C. Nehaniv and K. Dautenhahn, "Of hummingbirds and helicopters: An algebraic framework for interdisciplinary studies of imitation and its applications," in *Interdisciplinary Approaches to Robot Learning*, Y. Demiris and B. A., Eds. Singapore: World Scientific Press, 1999, pp. 136–161.

R. D. Newman-Norlund, M. L. Noordzij, R. G. J. Meulenbroek, and H. Bekkering, "Exploring the brain basis of joint action: co-ordination of actions, goals and intentions." *Social Neuroscience*, vol. 2, no. 1, pp. 48–65, 2007.

M. N. Nicolescu and M. J. Matarić, "Natural methods for robot task learning: instructive demonstrations, generalization and practice," in *AAMAS '03: Proceedings of the second international joint conference on Autonomous agents and multiagent systems*. New York, NY, USA: ACM, 2003, pp. 241–248.

L. M. Nimmo and S. Lewandowsky, "From brief gaps to very long pauses: temporal isolation does not benefit serial recall." *Psychonomic Bulletin & Review*, vol. 12, no. 6, pp. 999–1004, 2005.

D. A. Norman, "Categorization of action slips." *Psychological Review*, vol. 88, no. 1, pp. 1–15, 1981.

I. Nourbakhsh, K. Sycara, M. Koes, M. Yong, M. Lewis, and S. Burion, "Human-Robot Teaming for Search and Rescue," *IEEE Pervasive Computing*, vol. 4, no. 1, pp. 72–78, 2005.

K. Ogawara, J. Takamatsu, H. Kimura, and K. Ikeuchi, "Generation of a task model by integrating multiple observations of human demonstrations," in *Proceedings 2002 IEEE International Conference on Robotics and Automation (Cat. No.02CH37292)*, vol. 2, no. May. IEEE, 2002, pp. 1545–1550.

E. Oja, "A simplified neuron model as a principal component analyzer," *Journal of Mathematical Biology*, vol. 15, pp. 267–273, 1982.

M. Oja, S. Kaski, and T. Kohonen, "Bibliography of Self-Organizing Map (SOM) Papers: 1998-2001 Addendum," *Neural Computing Surveys*, vol. 3, pp. 1–156, 2003.

M. Ollis, W. H. Huang, and M. Happold, "A Bayesian approach to imitation learning for robot navigation," in *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2007, pp. 709–714.

R. C. O'Reilly and M. J. Frank, "Making working memory work: a computational model of learning in the prefrontal cortex and basal ganglia." *Neural Computation*, vol. 18, no. 2, pp. 283–328, 2006.

N. Otero, J. Saunders, and K. Dautenhahn, "Teaching robot companions: the role of scaffolding and event structuring," *Connection Science*, 2008.

E. Oztop, M. Kawato, and M. Arbib, "Mirror neurons and imitation: a computationally guided review." *Neural Networks : the Official Journal of the International Neural Network Society*, vol. 19, no. 3, pp. 254–71, 2006.

M. Page and D. Norris, "The primacy model: a new model of immediate serial recall." *Psychological Review*, vol. 105, no. 4, pp. 761–81, 1998.

M. Pardowitz, R. Zöllner, and R. Dillmann, "Learning sequential constraints of tasks from user demonstrations," in *Humanoid Robots, 2005 5th IEEE-RAS International Conference on*. IEEE, 2005, pp. 424–429.

M. Pardowitz, S. Knoop, R. Dillmann, and R. Zöllner, "Incremental learning of tasks from user demonstrations, past experiences, and vocal comments," *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, vol. 37, no. 2, pp. 322–332, 2007.

P. Pastor, H. Hoffmann, T. Asfour, and S. Schaal, "Learning and generalization of motor skills by learning from demonstration," in *2009 IEEE International Conference on Robotics and Automation*. IEEE, 2009, pp. 763–768.

J. Pineau, M. Montemerlo, M. Pollack, N. Roy, and S. Thrun, "Towards robotic assistants in nursing homes: Challenges and results," *Robotics and Autonomous Systems*, vol. 42, no. 3-4, pp. 271–281, 2003.

J. F. Prather, S. Peters, S. Nowicki, and R. Mooney, "Precise auditory-vocal mirroring in neurons for learned vocal communication," *Nature*, vol. 451, no. 7176, pp. 305–10, 2008.

A. R. Preston and H. Eichenbaum, "Interplay of hippocampus and prefrontal cortex in memory." *Current Biology*, vol. 23, no. 17, pp. R764–73, 2013.

J. Provost, B. J. Kuipers, and R. Miikkulainen, "Developing Navigation Behavior Through Self-organizing Distinctive-State Abstraction," *Connection Science*, vol. 18, no. 2, pp. 159–172, 2006.

L. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257–286, 1989.

C. Ranganath, M. X. Cohen, and C. J. Brozinsky, "Working memory maintenance contributes to long-term memory formation: neural and behavioral evidence," *Journal of Cognitive Neuroscience*, vol. 17, no. 7, pp. 994–1010, 2005.

J. R. Reynolds, J. M. Zacks, and T. S. Braver, "A computational model of event segmentation from perceptual prediction." *Cognitive Science*, vol. 31, no. 4, pp. 613–643, 2007.

G. Rizzolatti and C. Sinigaglia, "The functional role of the parieto-frontal mirror circuit: interpretations and misinterpretations." *Nature reviews. Neuroscience*, vol. 11, no. 4, pp. 264–74, 2010.

G. Rizzolatti, L. Fadiga, V. Gallese, and L. Fogassi, "Premotor cortex and the recognition of motor actions." *Cognitive Brain Research*, vol. 3, no. 2, pp. 131–41, 1996.

D. A. Rosenbaum, R. J. Meulenbroek, J. Vaughan, and C. Jansen, "Posture-based motion planning: Applications to grasping." *Psychological Review*, vol. 108, no. 4, pp. 709–734, 2001.

S. Rosenthal, J. Biswas, and M. Veloso, "An effective personal mobile robot agent through symbiotic human-robot interaction," in *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems*, vol. 1, 2010, pp. 915–922.

K. Sakai, K. Kitaguchi, and O. Hikosaka, "Chunking during human visuomotor sequence learning." *Experimental Brain Research*, vol. 152, no. 2, pp. 229–242, 2003.

M. Salichs, R. Barber, A. Khamis, M. Malfaz, J. Gorostiza, R. Pacheco, R. Rivas, A. Corrales, E. Delgado, and D. Garcia, "Maggie: A Robotic Platform for Human-Robot Social Interaction," in *2006 IEEE Conference on Robotics, Automation and Mechatronics*. IEEE, 2006, pp. 1–7.

E. Salinas, "Background synaptic activity as a switch between dynamical states in a network," *Neural Computation*, vol. 15, no. 7, pp. 1439–1475, 2003.

Y. Sandamirskaya and G. Schöner, "An embodied account of serial order: How instabilities drive sequence generation," *Neural Networks*, 2010.

J. Saunders, C. L. Nehaniv, and K. Dautenhahn, "Teaching robots by moulding behavior and scaffolding the environment," in *Proceeding of the 1st ACM SIGCHI/SIGART*

conference on Human-robot interaction - HRI '06. New York, New York, USA: ACM Press, 2006, p. 118.

J. Saunders, C. L. Nehaniv, K. Dautenhahn, and A. Alissandrakis, "Self-imitation and Environmental Scaffolding for Robot Teaching," *International Journal of Advanced Robotic Systems*, vol. 4, no. 1, pp. 109–124, 2007.

P. H. Schiller, "Parallel information processing channels created in the retina." *Proceedings of the National Academy of Sciences of the United States of America*, vol. 107, no. 40, pp. 17 087–94, 2010.

G. Schöner, "Dynamical systems approaches to cognition," *Cambridge Handbook of Computational Cognitive Modeling*, pp. 101–125, 2007.

G. Schöner, M. Dose, and C. Engels, "Dynamics of behavior: Theory and applications for autonomous robot architectures," *Robotics and Autonomous Systems*, vol. 16, pp. 213–245, 1995.

A. R. Seitz and H. R. Dinse, "A common framework for perceptual learning," *Current Opinion in Neurobiology*, vol. 17, no. 2, pp. 148–153, 2007.

R. M. Shiffrin and J. Cook, "Short-term forgetting of item and order information," *Journal of Verbal Learning and Verbal Behavior*, 1978.

R. Silva, "Design e construção de um robot antropomórfico," Master Thesis, University of Minho, 2008.

R. M. Silva, E. Bicho, and W. Erlhagen, "Aros: An anthropomorphic robot for human-robot interaction and coordination studies," in *Proceedings of the CONTROLO2008 Conference 8th Portuguese Conference on Automatic Control*. UTAD - Vila Real, Portugal: UTAD, 2008, pp. 819–826.

D. Silver, J. a. Bagnell, and A. Stentz, "Learning from Demonstration for Autonomous Navigation in Complex Unstructured Terrain," *The International Journal of Robotics Research*, vol. 29, no. 12, pp. 1565–1592, 2010.

A. J. Smith, "Applications of the self-organising map to reinforcement learning," *Neural Networks*, vol. 15, no. 8-9, pp. 1107–1124, 2002.

R. Soares, E. Bicho, T. Machado, and W. Erlhagen, "Object transportation by multiple mobile robots controlled by attractor dynamics: theory and implementation," in *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems IROS 2007*, 2007, pp. 937–944.

J. Solis, K. Petersen, T. Ninomiya, M. Takeuchi, and A. Takanishi, "Development of anthropomorphic musical performance robots: From understanding the nature of music performance to its application to entertainment robotics," in *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2009, pp. 2309–2314.

A. Solway, B. B. Murdock, and M. J. Kahana, "Positional and temporal clustering in serial order memory." *Memory & Cognition*, vol. 40, no. 2, pp. 177–90, 2012.

E. Sousa, W. Erlhagen, and E. Bicho, "On Observational Learning of Hierarchies in Sequential Tasks: A Dynamic Neural Field Model," in *Computational Models of Cognitive Processes*. San Sebastian, Spain: World Scientific, 2014, pp. 196–210.

M. Sousa, S. Monteiro, T. Machado, W. Erlhagen, and E. Bicho, "Multi-robot cognitive formations," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2012, pp. 545–550.

L. Steels, "Intelligence with representation," *Philosophical transactions. Series A, Mathematical, Physical, and Engineering sciences*, vol. 361, no. 1811, pp. 2381–95, 2003.

R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. Cambridge, MA, USA: The MIT Press, 1998.

A. Takashima, I. L. C. Nieuwenhuis, O. Jensen, L. M. Talamini, M. Rijpkema, and G. Fernández, "Shift from hippocampal to neocortical centered retrieval network with consolidation." *The Journal of Neuroscience : the Official Journal of the Society for Neuroscience*, vol. 29, no. 32, pp. 10 087–93, 2009.

A. Takeuchi and S.-i. Amari, "Formation of topographic maps and columnar microstructures in nerve fields," *Biological Cybernetics*, vol. 35, no. 2, pp. 63–72, 1979.

P. Tamayo, D. Slonim, J. Mesirov, Q. Zhu, S. Kitareewan, E. Dmitrovsky, E. S. Lander, and T. R. Golub, "Interpreting patterns of gene expression with self-organizing maps: Methods and application to hematopoietic differentiation," *Proceedings of the National Academy of Sciences*, vol. 96, no. 6, pp. 2907–2912, 1999.

J. Tanji, K. Shima, and H. Mushiake, "Concept-based behavioral planning and the lateral prefrontal cortex," *Trends in Cognitive Sciences*, vol. 11, no. 12, pp. 528–534, 2007.

A. Tapus, C. Tapus, and M. Mataric, "The role of physical embodiment of a therapist robot for individuals with cognitive impairments," in *RO-MAN 2009 - The 18th IEEE International Symposium on Robot and Human Interactive Communication*. IEEE, 2009, pp. 103–107.

S. Thill and T. Ziemke, "Learning new motion primitives in the mirror neuron system: a self-organising computational model," *From Animals to Animats 11*, 2010.

S. Thill, J. Behr, and T. Ziemke, "Flexible sequence learning in a SOM model of the mirror system," *Proc. Of CogSci 2012*, pp. 2423–2428, 2012.

L. J. Toth and J. a. Assad, "Dynamic coding of behaviourally relevant stimuli in parietal cortex." *Nature*, vol. 415, no. 6868, pp. 165–8, 2002.

M. Toussaint, "Learning a World Model and Planning with a Self-Organizing, Dynamic Neural System." *NIPS*, 2003.

L. L. Travis, "Goal-based organization of event memory in toddlers," in *Developmental spans in event comprehension and representation: Bridging fictional and actual events*. Lawrence Erlbaum Associates, Inc, 1997, pp. 111–138.

S. Tsujimoto, A. Genovesio, and S. P. Wise, "Transient neuronal correlations underlying goal selection and maintenance in prefrontal cortex." *Cerebral Cortex (New York, N.Y. : 1991)*, vol. 18, no. 12, pp. 2748–61, 2008.

C. van der Gaag, R. B. Minderaa, and C. Keysers, "Facial expressions: what the mirror neuron system can and cannot tell us." *Social Neuroscience*, vol. 2, no. 3-4, pp. 179–222, 2007.

M. T. R. van Kesteren, G. Fernández, D. G. Norris, and E. J. Hermans, "Persistent schema-dependent hippocampal-neocortical connectivity during memory encoding and postencoding rest in humans." *Proceedings of the National Academy of Sciences of the United States of America*, vol. 107, no. 16, pp. 7550–5, 2010.

M. van Lent and J. E. Laird, "Learning procedural knowledge through observation," in *Proceedings of the International Conference on Knowledge Capture - K-CAP 2001*. New York, New York, USA: ACM Press, 2001, pp. 179–186.

H. Veeraraghavan and M. Veloso, "Teaching sequential tasks with repetition through demonstration," in *Proceedings of the 7th International Joint Conference on Autonomous Agents and Multiagent Systems-Volume 3*. International Foundation for Autonomous Agents and Multiagent Systems, 2008, pp. 1357–1360.

J. A. Walter and K. I. Schulten, "Implementation of self-organizing neural networks for visuo-motor control of an industrial robot," *IEEE Transactions on Neural Networks*, vol. 4, no. 1, pp. 86–96, 1993.

S. M. Weaver and C. M. Arrington, "The effect of hierarchical task representations on task selection in voluntary task switching." *Journal of Experimental Psychology. Learning, Memory, and Cognition*, vol. 39, no. 4, pp. 1128–41, 2013.

G. Weinberg and S. Driscoll, "Toward Robotic Musicianship," *Computer Music Journal*, vol. 30, no. 4, pp. 28–45, 2006.

D. West, *Introduction to graph theory*. Upper Saddle River: Prentice hall., 2001.

A. Whiten, *Natural theories of mind: Evolution, development and simulation of everyday mindreading*. Oxford: Basil Blackwell, 1991.

A. Whiten, E. Flynn, K. Brown, and T. Lee, "Imitation of hierarchical action structure by young children." *Developmental Science*, vol. 9, no. 6, pp. 574–582, 2006.

B. Widrow and M. E. Hoff, "Adaptive switching circuits," *IRE WESCON Convention Record, New York: IRE Part 4*, pp. 96–104, 1960.

H. R. Wilson and J. D. Cowan, "A mathematical theory of the functional dynamics of cortical and thalamic nervous tissue," *Kybernetik*, vol. 13, no. 2, pp. 55–80, 1973.

M. Wilson and G. Knoblich, "The case for motor involvement in perceiving conspecifics." *Psychological Bulletin*, vol. 131, no. 3, pp. 460–73, 2005.

J. M. Zacks and B. Tversky, "Event structure in perception and conception." *Psychological Bulletin*, vol. 127, no. 1, pp. 3–21, 2001.

J. M. Zacks, C. A. Kurby, M. L. Eisenberg, and N. Haroutunian, "Prediction error associated with the perceptual segmentation of naturalistic events," *Journal of Cognitive Neuroscience*, vol. 23, no. 12, pp. 4057–4066, 2011.

R. Zöllner, T. Asfour, and R. Dillmann, "Programming by demonstration: dual-arm manipulation tasks for humanoid robots," in *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE Cat. No.04CH37566)*, vol. 1, no. Iros. IEEE, 2004, pp. 479–484.

R. Zöllner, M. Pardowitz, S. Knoop, and R. Dillmann, "Towards Cognitive Robots: Building Hierarchical Task Representations of Manipulations from Human Demonstration," in *Proceedings of the IEEE International Conference on Robotics and Automation ICRA*, no. April, 2005, pp. 1535–1540.