UMinho | 2020

Ana Isabel Barros da Cunha

Efficient Learning of Sequential Tasks for Collaborative Robots: a Neurodynamic Approach

Universidade do Minho
Escola de Engenharia

Ana Isabel Barros da Cunha

Efficient Learning of Sequential Tasks for Collaborative Robots: a Neurodynamic Approach

janeiro de 2020

Universidade do Minho
Escola de Engenharia

Ana Isabel Barros da Cunha

Efficient Learning of Sequential Tasks for Collaborative Robots: a Neurodynamic Approach

Dissertação de Mestrado
Mestrado Integrado em Engenharia Eletrónica
Industrial e Computadores

Trabalho efetuado sob a orientação de
Professora Doutora Estela Bicho
Doutor Emanuel Sousa

janeiro de 2020

Acknowledgements

First and foremost, I would like to thank my scientific advisers, Prof. Estela Bicho and Dr. Emanuel Sousa, for their continuous help, motivation and scientific advice. Since I met Prof. Estela, she has been an enormous inspiration for my work, being a constant source of academic and personal support. For that, I will be forever grateful.

Secondly, I would like to thank all my colleagues at the Mobile and Anthropomorphic Robotics Laboratory at the University of Minho for their friendship, advice and collaboration. A special word goes to Flora, Paulo and Prof. Luís for all the knowledge and expertise they shared with me, and most importantly, for their friendship.

I am also very grateful to all my family and friends that have always supported me during this journey. I wish to thank them for their friendship and for all the good moments they have provided me through the years.

Next, I want to thank my past English teacher and dearest friend, Prof. Adelaide Fernandes, who always encouraged and supported any challenge I presented to her. All the good advice and knowledge she shared with me were fundamental to improve my work.

Lastly, I could not even begin to express how grateful I am to Vasco for the amount of patience and care he has provided me. He always made me believe in myself and made sure I had all I needed to overcome any challenge ahead. He has been my primary source of strength and personal support, and I feel very fortunate to have him in my life.

This work was carried out within the scope of the project "PRODUTECH SIF - Soluções para a Indústria do Futuro", reference POCI-01-0247-FEDER-024541, cofunded by "Fundo Europeu de Desenvolvimento Regional (FEDER)", through "Programa Operacional Competitividade e Internacionalização (POCI)".

STATEMENT OF INTEGRITY

Abstract

In the recent years, there has been an increasing demand for collaborative robots able to interact and cooperate with ordinary people in several human environments, sharing physical space and working closely with people in joint tasks, both within industrial and domestic environments. In some scenarios, these robots will come across tasks that cannot be fully designed beforehand, resulting in a need for flexibility and adaptation to the changing environments.

This dissertation aims to endow robots with the ability to acquire knowledge of sequential tasks using the Programming by Demonstration (PbD) paradigm. Concretely, it extends the learning models - based on Dynamic Neural Fields (DNFs) - previously developed in the Mobile and Anthropomorphic Robotics Laboratory (MARLab), at the University of Minho, to the collaborative robot Sawyer, which is amongst the newest collaborative robots on the market. The main goal was to endow Sawyer with the ability to learn a sequential task from tutors' demonstrations, through a natural and efficient process.

The developed work can be divided into three main tasks: (1) first, a previously developed neuro-cognitive control architecture for extracting the sequential structure of a task was implemented and tested in Sawyer, combined with a Short-Term Memory (STM) mechanism to memorize a sequence in one-shot, aiming to reduce the number of demonstration trials; (2) second, the previous model was extended to incorporate workspace information and action selection in a Human-Robot Collaboration (HRC) scenario where robot and human coworker coordinate their actions to construct the structure; and (3) third, the STM mechanism was also extended to memorize ordinal and temporal aspects of the sequence, demonstrated by tutors with different behavior time scales.

The models implemented contributed to a more intuitive and practical interaction with the robot for human co-workers. The STM model made the learning possible from few demonstrations to comply with the requirement of being an efficient method for learning. Moreover, the recall of the memorized information allowed Sawyer to evolve from being in a learning position to be in a teaching one, obtaining the capability of assisting inexperienced co-workers.

Key Words: Collaborative Robots; Human-Robot Interaction; Learning from Demonstration; Coordination of actions and sub-goals; Dynamic Neural Fields.

Resumo

Nos últimos anos, tem havido uma crescente procura por robôs colaborativos capazes de interagir e cooperar com pessoas comuns em vários ambientes, partilhando espaço físico e trabalhando em conjunto, tanto em ambientes industriais como domésticos. Em alguns cenários, estes robôs serão confrontados com tarefas que não podem ser previamente planeadas, o que resulta numa necessidade de existir flexibilidade e adaptação ao ambiente que se encontra em constante mudança.

Esta dissertação pretende dotar robôs com a capacidade de adquirir conhecimento de tarefas sequenciais utilizando técnicas de Programação por Demonstração. De forma a continuar o trabalho desenvolvido no Laboratório de Robótica Móvel e Antropomórfica da Universidade do Minho, esta dissertação visa estender os modelos de aprendizagem previamente desenvolvidos ao robô colaborativo Sawyer, que é um dos mais recentes no mercado. O principal objetivo foi dotar o robô com a capacidade de aprender tarefas sequenciais por demonstração, através de um processo natural e eficiente.

O trabalho desenvolvido pode ser dividido em três tarefas principais: (1) em primeiro lugar, uma arquitetura de controlo baseada em modelos neurocognitivos, desenvolvida anteriormente, para aprender a estrutura de uma tarefa sequencial foi implementada e testada no robô Sawyer, conjugada com um mecanismo de *Short-Term Memory* que permitiu memorizar uma sequência apenas com uma demonstração, para reduzir o número de demonstrações necessárias; (2) em segundo lugar, o modelo anterior foi estendido para englobar informação acerca do espaço de trabalho e seleção de ações num cenário de Colaboração Humano-Robô em que ambos coordenam as suas ações para construir a tarefa; (3) em terceiro lugar, o mecanismo de *Short-Term Memory* foi também estendido para memorizar informação ordinal e temporal de uma sequência de passos demonstrada por tutores com comportamentos temporais diferentes.

Os modelos implementados contribuíram para uma interação com o robô mais intuitiva e prática para os *co-workers* humanos. O mecanismo de *Short-Term Memory* permitiu que a aprendizagem fosse realizada a partir de poucas demonstrações, para cumprir com o requisito de ser um método de aprendizagem eficiente. Além disso, a informação memorizada permitiu ao Sawyer evoluir de uma posição de aprendizagem para uma posição em que é capaz de instruir *co-workers* inexperientes.

Palavras-chave: Robôs Colaborativos; Colaboração Humano-Robô; Aprendizagem por Demonstração; Coordenação de ações e tarefas; Campos Dinâmicos Neuronais.

Contents

List of Abbreviations

AEL	Action Execution Layer
ANN	Artificial Neural Networks
APN	Adaptive Petri-Net
AROS	Anthropomorphic Robotic System
CSGL	Common Sub-Goals Layer
DNF	Dynamic Neural Field
ETL	Executable Tasks Layer
HHO	Human Hands Over
HI	Human Inserts
HMM	Hidden Markov Models
HRC	Human-Robot Collaboration
HRI	Human-Robot Interaction
LfD	Learning from Demonstration
LMS	Least-Mean Square
MARLab	Mobile and Anthropomorphic Robotics Laboratory
OML	Object Memory Layer
PbD	Programming by Demonstration
RHO	Robot Hands Over
RI	Robot Inserts
ROS	Robotic Operation System

SL	Sequence Layer
SLL	Sequence Learning Layer
STM	Short-Term Memory

Nomenclature

x,y	Spatial dimension
u	Neuronal activation
t	Time variable
τ	Time constant
h	Neuronal resting level
s	Neuronal input
f_0	Heaviside step function
w	Interaction kernel
A	Amplitude of a Gaussian bump
σ	Standard deviation of a Gaussian bump
w_{inh}	Global inhibition of an interaction kernel
B	Amplitude of the negative Gaussian bump of a *Mexican-Hat* kernel function
a_i	Weight of a synaptic connection in a single neuron
m	Total number of input nodes in a discrete neural network
q_i	Value of an input node i
$\varphi(.)$	Activation function of a neuron
$a_{i,j}$	Weight of the synaptic connection $\Delta a_{i,j}$
A	Matrix of synaptic weights
$\Delta a_{i,j}$	Weight variation in the synaptic connection of the input i to the neuron j

η	Rate parameter for a neural network learning rule
x^{in}	Spatial dimension of a neural network input
x^{out}	Spatial dimension of a neural network output
τ_a	Time scale of the differential equation defining a synaptic weight variation
$u^{pa}(x^{pa},t)$	Neuronal activation of the Past Field
$u^{pr}(x^{pr},t)$	Neuronal activation of the Present Field
λ^{pr}	Learning threshold of the Present field
λ^{pa}	Learning threshold of the Past field
h^{pr}	Neuronal resting level of u^{pr}
$h^{pa}(x^{pa},t)$	Neuronal resting level of u^{pa}
h^{pa}_{low}	Lower value for the resting level of u^{pa}
h^{pa}_{high}	Higher value for the resting level of u^{pa}
ς^{k}_{stoch}	Stochastic noise function
$s^{pa}(x^{pa},t)$	Input to the Past field $u^{pa}(x^{pa},t)$
$v^{pa}(x^{pa},t)$	Input from the Vision System to $s^{pa}(x^{pa},t)$
$C^{v\rightarrow pa}$	Gain parameter controlling the strength of contribution v^{pa} to s^{pa}
τ^{pa}_h	Time constant of $h^{pa}(x^{pa},t)$
$v^{pr}(x^{pr},t)$	Input from the Vision System to $s^{pr}(x^{pr},t)$
$C^{v\rightarrow pr}$	Gain parameter controlling the strength of contribution v^{pr} to s^{pr}
$C^{rec\rightarrow pr}$	Gain parameter controlling the strength of the Recall contribution to s^{pr}
$s^{pr}(x^{pr},t)$	Input to the Present field $u^{pr}(x^{pr},t)$
$\underset{pa\rightarrow pr}{A}(x^{pa},x^{pr},t)$	Adaptable excitatory synaptic interaction from u^{pa} to u^{pr}
$e(x^{pr},t)$	Difference between desired and actual value of the activation
$u^{stm}(x^{stm},t)$	Neuronal activation of the STM Field
$v^{stm}(x^{stm},t)$	Input from the Vision System to $s^{stm}(x^{stm},t)$
$C^{v\rightarrow stm}$	Gain parameter controlling the strength of contribution v^{stm} to s^{stm}
$s^{stm}(x^{stm},t)$	Input to the STM field $u^{stm}(x^{stm},t)$
τ^{stm}_h	Time constant of $h^{stm}(x^{stm},t)$
$h^{stm}(x^{stm},t)$	Neuronal resting level of u^{stm}

H_{dec}^{stm}	Baseline value of $h^{stm}(x^{stm}, t)$
$C^{stm \to pr}$	Gain parameter controlling the strength of contribution of the STM Layer to s^{pr}
b^{reh}	Flag active during the Rehearsal mode
$\underset{pa \to oml}{I}(x^{pa}, x^{oml})$	Predefined inhibitory synaptic interaction from u^{pa} to the OML fields
$\underset{pa \to pr}{I}(x^{pa}, x^{pr})$	Predefined inhibitory synaptic interaction from u^{pa} to u^{pr}
$u^{hw}(x^{hw}, t)$	Neuronal activation of the Human Workspace field
$u^{rw}(x^{rw}, t)$	Neuronal activation of the Robot Workspace field
$u^{sw}(x^{sw}, t)$	Neuronal activation of the Shared Workspace field
$v^{oml}(x^{oml}, t)$	Visual input to the OML fields
$C^{v \to oml}$	Gain parameter controlling the strength of contribution v^{oml} to s^{oml}
$u^{etl}(x^{etl}, t)$	Neuronal activation of the Executable Tasks Layer
$s^{etl}(x^{etl}, t)$	Input to the ETL field $u^{etl}(x^{etl}, t)$
$u^{ael}(x^{c}, x^{a}, t)$	Neuronal activation of the Action Execution Layer
x^{c}	AEL dimension representing “color”
x^{a}	AEL dimension representing “action”
$s^{ael}(x^{c}, x^{a}, t)$	Input to the AEL field $u^{ael}(x^{c}, x^{a}, t)$
$C^{c \to ael}$	Gain parameter controlling the strength of contribution “action” to s^{ael}
$C^{a \to ael}$	Gain parameter controlling the strength of contribution “color” to s^{ael}
$\underset{pa \to ael}{I}(x^{pa}, x^{ael})$	Predefined inhibitory synaptic interaction from u^{pa} to u^{ael}
$u^{SM}(x, t)$	Neuronal activation of the Sequence Memory field
$u^{SR}(x, t)$	Neuronal activation of the Sequence Recall field
$u^{PE}(x, t)$	Neuronal activation of the Past Events field
h_0^{SM}	Neuronal resting level of $u^{SM}(x, t)$
$\tau^{h^{SM}}$	Time constant of h^{SM}
$\tau^{h^{SR}}$	Time constant of h^{SR}
τ^{pa}	Time constant of u^{pa}
τ^{pr}	Time constant of u^{pr}

τ^{stm}	Time constant of u^{stm}

List of Figures

List of Tables

Part I

Introduction

Chapter 1

Introduction

Future generations of robots are expected to assist ordinary people in tasks that cannot be programmed beforehand. Thus, endowing robots with learning mechanisms and cognitive capabilities has become a priority in the robotics field. These abilities should allow robots to adapt to a wide variety of environments and tasks, responding to the specific needs and preferences of the human partners with whom they interact. This dissertation aims to provide new insights into natural and efficient Human-Robot Interaction (HRI)/Human-Robot Collaboration (HRC) with focus on the task-learning-from-demonstration paradigm, by continuing the work developed with collaborative robots at the Mobile and Anthropomorphic Robotics Laboratory (MARLab), University of Minho. Specifically, the work proposed in this dissertation focus on the development and testing, on Robot Sawyer, of efficient learning models which are based on Dynamic Neural Fields (DNFs). The ultimate goal is to endow Sawyer with the ability to learn sequential tasks, from tutor's demonstration and verbal feedback, without previously programming task-relevant information.

1.1 Motivation

Robots are today an increasingly important part of the modern world, not only because they are a fundamental tool in many industries, but also due to their ability to assist humans in hazard environments and dangerous tasks (Wiese et al., 2017). However, despite the continued research and the amount of progress made in their development, their presence in our everyday lives is still rather reduced, since their ability to interact with humans in a natural and efficient way is still limited, highlighting how challenging it is to design robots that can interact socially, perceiving intentions, learning new tasks and adapting to different users and workplace changes (Wiese et al., 2017).

So far, real-world examples of HRI have been mostly limited to scenarios in which the robot still requires some kind of human assistance such as teleoperation, manipulation/remote control or human supervision (see e.g. Scholtz, 2003; Sheridan, 2016). Some researchers analyzed the impact of the way humans and robots communicate, focusing on the influence of communicating through gestures and verbal commands in the HRI process. It was concluded that *"when the robot used co-verbal gestures during interaction, it was anthropomorphized more, participants perceived it as more likeable, reported greater shared reality with it, and showed increased future contact intentions"* (Salem et al., 2013). Moreover, the integration of robotic partners in human-dominated areas, such as in industrial settings, is only likely to be accepted by their biological colleagues if they allow for natural cooperation in which co-workers are not required to learn new forms of interaction (Wise, 2018). Thus, for fluent interaction and for building trust, the robot should actively contribute to the work and continuously communicate its reasoning and decisions to its human co-workers (Ackerman, 2019).

Furthermore, it has been stated that the majority of HRI applications have been designed by experts in the field for their own use, restricting the background of operators able to work efficiently with such machines. Hence, developing control architectures able to accommodate cognitive skills and new task-knowledge held by any operator remains an important research goal for achieving an efficient HRC (Argall et al., 2009; Khamassi et al., 2016). Consequently, roboticists have started to question how to endow robots with an adaptive, efficient and natural interaction method to allow ordinary people, with no expert knowledge in the field, to teach and work with robots without necessarily having to program the machine for every requested task (Sousa et al., 2014).

Learning from Demonstration (LfD) and imitating others behaviors have been proved to be effective tools of social learning mechanism for transferring knowledge between humans (Bandura, 1977; Lind et al., 2019). Considering how effective we are when transferring knowledge from one another, it is reasonable to expect that this approach could also bring good results with robots learning from humans, the same way humans learn from each other (Sousa, 2014). Plus, it is also believed that the concept of efficient learning relates to the assumption that "good" interaction with a robot must reflect natural (human-human) interaction and communication as closely as possible in order to ease people's need to interpret the robot's behavior (Sandry, 2015). Therefore, to earn the designation of a collaborative (intelligent) robot, the machine must be provided with mechanisms to adapt its behavior to new tasks and co-workers without having to be programmed for such actions (Salem et al., 2013). Although LfD has been already used in several applications such as real-time motion capture (Koenemann et al., 2014), domestic tasks, like filling a cup with tap water (Misra et al., 2016), among others, only a few authors, such as Erlhagen et al. (2005, 2006), Sandamirskaya and Schöner (2010) and Sousa et al. (2009, 2014, 2015) have based their work on cognitive models of sequence learning in humans that reflect the way humans learn from observation. This will be one of the main topics of this dissertation.

1.2 Human-Robot Interaction (HRI)

Since the 1940s, robots and humans have been interacting with each other. Over the years, this interaction has evolved from the use of basic interfaces, such as joysticks and teleoperation, to speech commands and others (Sheridan, 1997). In 2001, Fong et al. proposed the notion of "robot as a partner" instead of "robot as a tool", while also pointing out the need for robots to have some kind of human assistance when facing unstructured environments as a way of increasing their efficiency in being autonomous and making decisions. Their work introduced the term "collaborative control", to define an interaction paradigm in which humans interacted with "social aware" robots to achieve common goals (Fong et al., 2003b). This new paradigm has motivated researchers to develop new robots, focusing not only on their motor ability to execute a specific goal but also on developing the cognitive skills necessary to achieve it through decision making and intention recognition in cooperation with a human partner. Moreover, Fong et al. classified social robots into two categories regarding their design approach: "functionally designed" and "biological inspired." The former describes robots that may appear socially capable but whose design is not based on any biological theory (Fong et al., 2003a) (see also Kim and Suzuki, 2010). The latter refers to robots whose implementation takes into consideration biological principles such as psychological or neuro-cognitive findings regarding human join action. Examples of biologically inspired architectures can be seen in Breazeal et al. (2009), where an anthropomorphic robot, *Leonardo* (Figure 1.1)[1], is used to interact with a human partner to test principles of the theory of mind. Also, in the work of Dautenhahn et al. (2009), an architecture to test psychological behavior with autistic children was developed and tested in the robot *KASPAR* (Figure 1.2). Finally, relevant research was also developed by Erlhagen and Bicho (Erlhagen and Bicho (2006, 2014), Erlhagen et al. (2006), Bicho et al. (2011a, 2011b)) with the use of neurodynamic fields to build bio-inspired goal-oriented architectures for learning and decision-making on HRI scenarios. These architectures

[1] **Personal Robots Group – MIT Media Lab:** https://robotic.media.mit.edu/portfolio/leonardo/

were implemented and tested on the Anthropomorphic Robotic System (AROS) (Figure 1.3). AROS was built in the MARLab, at University of Minho, with an anthropomorphic shape that allows it to cooperate with humans in join tasks in a more human-like way (Silva, 2008). Several cognitive models were implemented and tested in AROS in the scope of the EU Project JAST [2], which brought relevant knowledge to the field (Erlhagen & Bicho, 2014).

Figure 1.1: Robot Leonardo. Developed by MIT (Massachusetts Institute of Technology) Media Lab. Retrieved from Breazeal et al. (2009).

Figure 1.2: KASPAR Robot. Retrieved from Dautenhahn et al. (2009).

[2]**Project JAST** - Joint Action Science and Technology (proj. Nr. 003747)

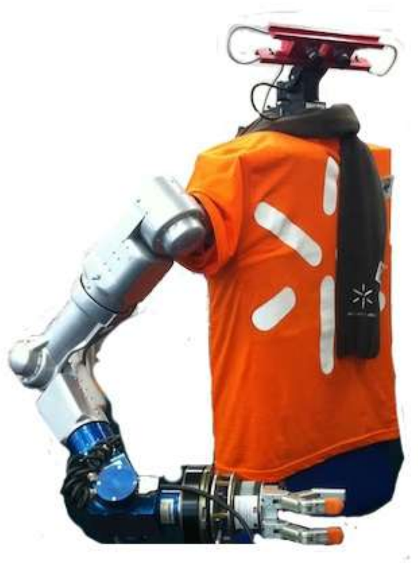


Figure 1.3: AROS. Anthropomorphic robot developed at the MARLab.

1.3 Learning from Demonstration

The Learning-from-Demonstration paradigm has been used in robotics to address the issue of enabling robots with learning capabilities. This technique allows ordinary people - with no expert knowledge in the robotics field - to interact and program those machines without having any programming background. Moreover, HRI researchers have also been employing LfD to endow robots with the capability of learning tasks with requirements that cannot be programmed beforehand (unstructured environments), increasing efficiency by correcting the robot's behavior and providing feedback, for example (Lee, 2017). Some application examples are the learning of a sequential task of picking, carrying and dropping a ball inside a box (Veeraraghavan & Veloso, 2008), the study of active learning in HRI through the means of asking/answering questions (Cakmak & Thomaz, 2012) and learning a multi-step task on a mobile manipulator (Niekum et al., 2012).

There are several approaches inside the LfD framework (see Dillmann, 2004). For the purpose of this work, task learning to extract information at the plan level - representing a task as a sequence of sub-actions -, is taken into consideration. Sousa et al. (2014) separates the question of developing a model for learning a task representation at the plan level into three main issues: i) selecting the method used by the tutor to provide information to the learning system, as well as selecting which inputs and variables to use; ii) segmenting the end-task into sub-goals that can be encoded as units of action, observed through multiple demonstrations and reproducible by the robotic platform; and, iii), how to acquire the sequential order of the tasks and encode the various units of action. It is worth noting that most tasks can be performed through different sequences of order. Take the example of making coffee: one can insert the coffee capsule in the first place and then turn the machine on and put the cup in place; another can put the cup in place first and then turn the machine on and insert the capsule. Learning flexible task representations requires structures able to encode different rules regarding the relations between the different unions of action and of extracting the necessary information from several observations. Chapter 3 will focus on this matter.

1.4 Framework

In the last years, researchers from the MARLab, at the University of Minho, have been grounding their work on the Dynamic Neural Fields theory (Amari, 1977) to endow autonomous robots with cognitive and learning capabilities (Erlhagen and Bicho (2006); Bicho et al. (2009, 2011a, 2011b); Sousa et al. (2009, 2014)). Erlhagen and Bicho (2006) tested a cognitive architecture to reproduce an observed grasping–placing sequence, stressing the importance of common task knowledge in shared tasks. Later, Bicho and colleagues explored the use of non-verbal commands in HRI by extending the architecture to integrate contextual cues, shared task knowledge and to predict the outcome of the user's motor behavior (Bicho et al., 2009). Contributing to the improvement of HRI in social scenarios, the previous work in MARLab resulted in the cognitive robot control architecture displayed in Figure 1.4, which was motivated by neuro-cognitive findings in joint action in humans and it is mathematically formalized using Dynamic Neural Fields (Bicho et al., 2011b; Erlhagen & Bicho, 2014). The model implements the association between observed and executed actions and uses information of the inferred goal of the co-worker, contextual cues and shared task knowledge. The Vision System incorporated in the robot recognizes the co-worker's motor action, e.g. reaching an object, triggering a supra-threshold activity in the neuronal population corresponding to the same action on the (observing) robot. The architecture was successfully implemented in the Anthropomorphic Robotic System, allowing the robot to predict the Human's motor intention, predict and detect errors in the sequences and select an appropriate complementary behavior in collaborative tasks (Bicho et al., 2012). However, this was possible only because AROS had predefined – i.e., preprogrammed - knowledge about the sequential structure of the tasks, encoded in the Common Sub-Goals Layer. To overcome this limitation, Sousa et al. (2009, 2014, 2015) applied associative learning mechanisms on robot AROS thus allowing it to learn

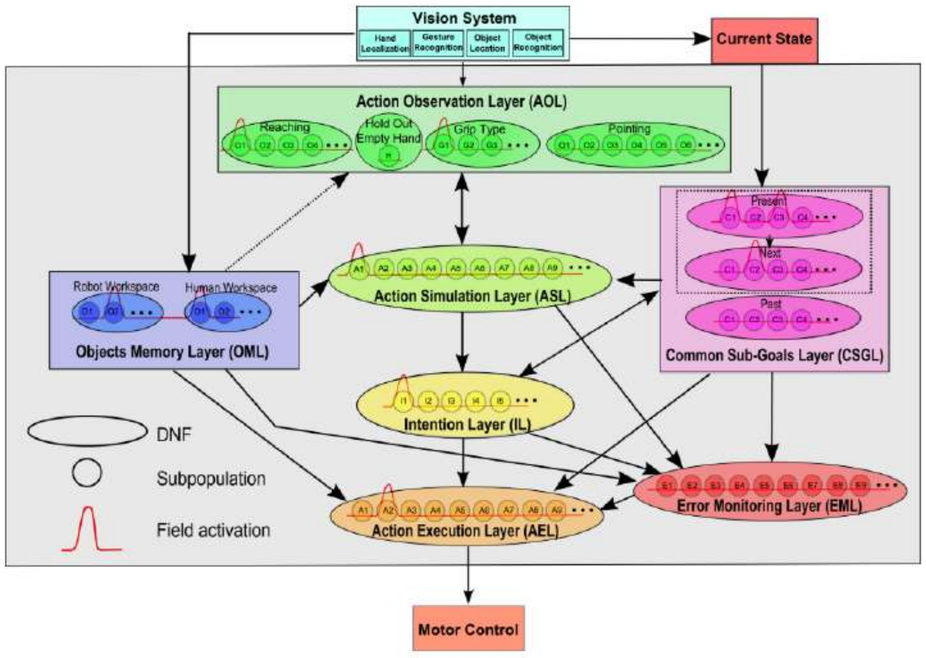


Figure 1.4: Cognitive architecture for human-robot interaction in joint tasks. It implements the relation between observed actions (AOL layer) and complementary actions (AEL layer), taking into account the inferred action-goal of the co-worker (IDL layer), contextual cues (OML layer) and shared task knowledge (CSGL layer). The goal inference ability is based on motor simulation (ASL layer) (Erlhagen & Bicho, 2014). Retrieved from Malheiro et al. (2017).

sequential tasks from tutors' demonstrations and verbal feedback. However, it remained to be seen how the joint action architecture and the learning models can be translated to other robots such as the robot Sawyer.

1.5 Main Goals and Contributions of this dissertation

This dissertation aims to contribute to the development of natural and efficient HRI/HRC by endowing robots with the ability to learn sequential tasks without pre-programmed knowledge of the same. It extends the work developed by Sousa et al. (2009, 2014, 2015) to the robotic system Sawyer, a "collaborative" robot from Rethink Robotics[3]. Thus, the main goal of this work was to endow Sawyer with learning mechanisms and allow it to acquire knowledge regarding the structure of sequential tasks and to make decisions in join cooperation with a human co-worker.

[3]**Robot Sawyer:** http://www.rethinkrobotics.com/sawyer/

Subsequently, the architecture implemented for Sequence Learning was extended to incorporate workspace information and action selection in a real Human-Robot Collaboration scenario, where Robot and Human co-workers coordinated their actions to accomplished sub-goals in a specific sequential task - “what to do” and “who does it”.

Additionally, a model for memorizing sequences with time constraints based on the work of Ferreira et al. (2014) was implemented and tested. This learning mechanism allowed the robot not only to memorize long sequences of sub-goals in a task, but also the time interval between them. The integration of these two characteristics - ordinal and temporal information - allowed the robot to memorize in one shot “what to do” and “when to do” in the HRI task scenario adopted. The implemented system capacitated the robot to instruct inexperienced workers or to make decisions when playing the role of an active assistant.

The following publications resulted from the work developed during this dissertation:

- Cunha, A., Ferreira, F., Erlhagen, W., Sousa, E., Louro, L., Vicente, P., Monteiro, S., and Bicho, E. (2020). **Towards endowing collaborative robots with fast learning for minimizing tutors’ demonstrations: What and when to do?** In Silva, M. F., Luís Lima, J., Reis, L. P., Sanfeliu, A., and Tardioli, D., editors, *Robot 2019: Fourth Iberian Robotics Conference*, pages 368–378, Cham. Springer International Publishing, DOI: https://doi.org/10.1007/978-3-030-35990-4_30

- Cunha, A., Ferreira, F., Sousa, E., Louro, L., Vicente, P., Monteiro, S., Erlhagen, W. ,and Bicho, E. (2020). **Towards Collaborative Robots as Intelligent Co-workers in Human-Robot Joint Tasks: what to do and who does it?** (Submitted to *ISR 2020; 52th International Symposium on Robotics*)

1.6 Dissertation Outline

The reminder of this dissertation is organized as follows:

Part II, consists on the state of the art relevant for this dissertation:

- Chapter 2 gives an overview of the Learning from Demonstration paradigm, focusing on learning a task as a sequence of sub-goals;

- Chapter 3 describes the existing models for extracting and encoding task-relevant information, starting by presenting some of the structures used to store sequential tasks followed by an general analysis of the cognitive models for serial order.

Part III, presents the theoretical background and materials used in this work:

- It starts with Chapter 4 where the framework of Dynamic Neural Fields is presented, together with the key equations that formalize the models;
- Next, Chapter 5 introduces the concept of Artificial Neural Networks as well as the key learning rules used in this dissertation;
- Finally, Chapter 6 describes the robotic platform and the Human-Robot Collaboration scenarios used to validate the DNF-models.

Part IV, describes the Implementation and Results of the several experiments made to validate the DNF-models. It consists of three Chapters with three different experiments:

- Chapter 7 presents the models for sequence learning with the integration of the Short-Term Memory mechanism, where Sawyer learns the sequential structure of the task from tutor demonstration;
- Chapter 8 extends the DNF-models to an architecture that incorporates workspace information and allows Human and Robot to conjugate their motor actions to construct the task;
- Finally, Chapter 9, presents a new model for learning sequences with time constraints.

Part V, is composed of Chapter 10 that concludes this dissertation and points out some of the future work that can be developed.

Part II

State of the Art

Chapter 2

Learning from demonstration in Robotics

This chapter presents the Learning from Demonstration paradigm and gives an overview of the work that has been developed using this technique in the field of robotics, specifically in Human-Robot Interaction scenarios.

2.1 General Introduction

"Robots are coming out of their cages to work alongside humans. This has significant implications for how work, and organizations, will be structured in the future", claims the International Federation of Robotics (2018). In the last years, workplace environments have been changing from scenarios where robots replace the work of humans to others where robots and humans collaboratively work together in joint tasks.

Traditionally, programming a robotic system can be a very hard working and time-demanding task. Furthermore, it is unrealistic to expect that every human, either in domestic or industrial environments, can have the knowledge and technical background required to instruct/program a robot to perform a new task. Let's imagine that a robot is in charge of assisting a human through its domestic household chores. Every human has different needs and every house a different layout. The robot needs to be flexible and adaptable to fit their user's needs and environment, hence programming such systems beforehand is not feasible (Dillmann, 2004). Therefore, the traditional way of fully programming machines to do a particular task cannot answer the challenge of coping with unstructured environments and unpredictable human behaviors. This creates a demand for a different approach into programming robots, specifically to allow the machines to learn from humans in a natural, easy, humanfriendly and intuitive way (Chernova & Thomaz, 2014). Learning from Demonstration is an approach that enables non-expert workers in the robotics field to teach a robot complex tasks so that they can autonomously perform tasks that were not pre-programmed in the machine beforehand. Furthermore, it allows end-users to teach robots (through demonstration) according to the real-world requirements where the machine needs to operate (Lee, 2017).

2.2 Human-Robot Collaboration in industrial applications

Having the power to replace unsafe and heavy tasks for humans, companies are more and more interested in incorporating robots into their assembling lines to work side by side with human operators (Krüger et al., 2006). Today, the traditional robotic manipulators used in the industry are extensively incorporated into production and assembling lines due to their robustness and capabilities such as precision and repeatability. Those machines are, however, pre-programmed "by hand" by human workers with programming capabilities and/or robotics knowledge to execute specific tasks in structured work environments and assembling lines. This method leaves reduced space to accommodate changes in the place of work that were not designed in advance by the programmer. For that, the programmer would have to predict every possible scenario and instruct the robot to act according to each specific condition, which would result in a large amount of code, testing and time spending. Moreover, while the programming is being conducted, the machine stays out of service, decreasing the efficiency and productivity in the overall process (Kyrarini et al., 2018).

Another issue worth pointing out is the way humans and robots interact in such scenarios. In order to make human co-workers more comfortable with their robotic partners and to maintain a natural and intuitive interaction, those machines should be designed to recognize human-like signs such as desires and intentions (Lee, 2017). However, in most applications, there is no direct communication between the user and the machine, so the human partner's actions, feelings and intentions during the joint cooperation are not taken into consideration by the robot. For an effective cooperation during the joint construction of a task, for example, when the human partner grabs an object, the robot should be able to recognize if its partner's intention is grabbing to insert or grabbing to pass it to the robot itself (see Bicho et al. (2012) for an extended review on this matter).

2.3 The Learning from Demonstration paradigm

The Learning from Demonstration paradigm - firstly known as Programming by Demonstration (PbD) - started to attract researchers in the early 1980s as a way of reducing the programming workload and its costs in factories (Billard et al., 2008). The first developed methods were based on teleoperation, e.g., in scenarios where the robot had to reproduce specific movements (see Levas and Selfridge, 1984). Since then, this method has been used in a variety of robotics-related tasks such as navigating through unknown environments (Bicho, 1999; Ollis et al., 2007), teaching human-like gestures to anthropomorphic robots (Calinon et al., 2007), learning goal-direct motor primitives (Erlhagen et al., 2006) and object manipulation tasks (Kyrarini et al., 2018), to name a few.

Within the LfD framework, only a few studies, such as Kuniyoshi et al. (1994) and Sato et al. (2002), have focused on the learning of task representations from human observation, using methods of extracting information from demonstration to acquire task-related information in assembly work. Despite the number of different points of view and approaches (see Billard et al. (2008), Argall et al. (2009) and Chernova and Thomaz (2014) for a more extensive review), there are common challenges that need to be tackled when designing systems capable of extracting task-relevant information. Sousa (2014) separated these challenges into three main fundamental issues: (1) selecting the most suitable method for the tutor to transfer information to the learning system, as well as which inputs and variables (e.g., vision, speech, gestures) to use, in order to provide relevant and clear information for learning; (2) the separation of the task into units of sub-goals/sub-tasks - what precisely the machine will learn - e.g, a goal (yellow column inserted), a motor action to reach the goal (pick the pink column), that can be recognized by the system through the multiple demonstration trials and can be reproduced by the robot; (3) how to encode the sequential order through which the various sub-tasks need to be performed and such information can be used. The next chapter provides an extended overview of this issue.

Learning a task as a sequence of sub-goals

Learning from Demonstration provides a useful framework to address learning issues in human-robot joint tasks. Therefore, it is important to have in mind what exactly is the human tutor teaching the robot. For the relevance of this work, a task is divided into a sequence of sub-goals that must be achieved through a specific order, respecting the inherent task-constraints. While some tasks may require a single order of execution, there are several tasks/end-goals that may be achieved through different sequences. Learning such tasks requires ways of extracting and encoding not only a sequence but also the precedence relations between the several sub-goals, as well as analyzing each sequence, selecting each ones fulfill the environmental constraints.

Let's say that, during an assembly-like task, the next object to be mounted (according to the pre-programmed sequential steps) is missing. There may be other objects available that can be inserted subsequently without compromising the final structure. If the system had learned the task itself - with all the constraints necessary to build the final task -, instead of just performing a set of preprogrammed steps, the robot would be able to recognize that there was no need to stop the work and would be able to continue doing its chore while waiting for the missing piece. This way, the assembling line would not be stopped while waiting for the missing object and it could continue the task at least until the point where it would no longer be possible to continue without the missing item. Concluding, knowing the task brings greater autonomy and flexibility to the machine. Moreover, endowing these robots with cognitive and learning capabilities allows them to make decisions in the absence of a human guide as well as instruct other inexperienced workers with knowledge about the tasks to be performed (Cunha et al., 2020)

Chapter 3

Models for extracting and encoding task-relevant information

In this dissertation, learning task-related knowledge from demonstration is one of the core paradigms. Those tasks are encoded as a sequence of sub-goals that must be achieved, respecting its serial order constraints. Acquiring task-relevant information often implies storing sequential information as a sequence of sub-tasks that need to be achieved through one (or more) sequential orders. This requires that the system has to learn the relations between sub-goals/sub-tasks when executing the task/end-goal. In this chapter, an overview of the most commonly used methods for storing information will be discussed, as well as some examples of structures used for encoding task-knowledge. The chapter ends with an overview of sequential behavior models.

3.1 Diagrams for encoding multi-path task information

3.1.1 Precedence graphs

Precedence graphs are the most common structures for encoding multi-sequence paths. These are formed by a set of nodes connected through edges and are commonly used to encode sequences of multiple paths (Figure 3.1). Each node represents a sub-goal or sub-action to be achieved while the edges delineate the precedence relations between them, that should be respected through the various steps, until the end-node/end-goal is reached. In any of the cases, the robot must adopt a behavior that will allow it to achieve each goal, always respecting the precedent actions. Those relations can be the result of pre-programmed control structures (Ekvall & Kragic, 2006) or learning mechanisms consisting of algorithms used to extract a control model from demonstration (Sousa et al., 2014). In order to build the precedence graph of a certain task, the conditions leading to each node (pre-conditions) have to be analyzed, as well as the output-conditions that result from reaching the same node (post-conditions).

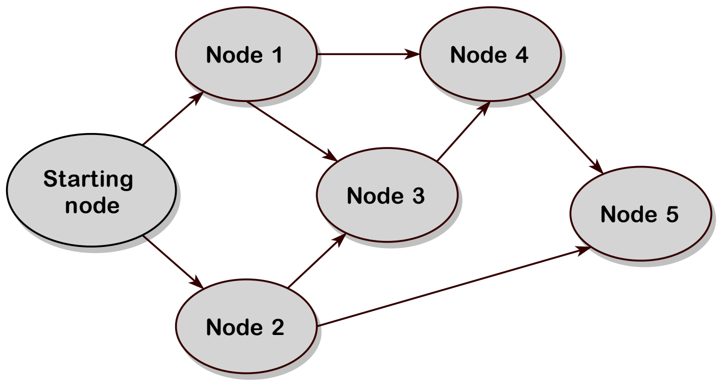


Figure 3.1: Precedence graph describing the relation between six nodes. Each node represents a sub-goal of an end-task that is represented by the total set of nodes.

Several studies have been conducted using this type of structure in robotic learning applications. Examples can be found in Nicolescu and Mataric (2003), where the authors used a similar approach to teach a robot in a pick and place navigation task. A slightly different approach can be seen in Veeraraghavan and Veloso (2008), where a precedent analysis is also used, combined with the addition of a mechanism to distinguish actions from objects, allowing actions to be assigned to different objects throughout the task. In the work of Konidaris et al. (2012), a different structure is used in which the authors assume multiple starting nodes for the same end-task in a robot navigation system. Their algorithm builds a tree-like instance where each edge represents a trajectory path. Through the observations, similar trajectories are merged and new trajectories are created from new branches.

Petri-Nets

Another type of precedence graphs used in LfD are Petri-Nets. These are graphs that encode not only nodes/states and precedence relations but also the transitions leading from one node to another. Those transitions can be, for example, the trajectory needed to reach the node corresponding to a Point A starting from a Point B. Chang

(2013) used this method to introduce error recovering capabilities in its learning by imitation model, to allow a robot to adapt the sequence of actions needed during task execution. However, this required a human user to manually edit the same structure to add additional nodes and transitions. Later, to overcome this limitation, Zhao et al. (2017) suggested and Adaptive Petri-Net (APN) approach (Figure 3.2) to allow changes in the original net structure that can be added autonomously by the robotic system.

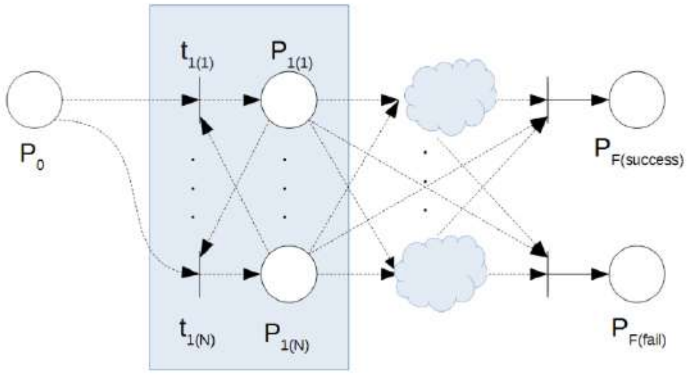


Figure 3.2: Adaptive Petri-Net. The shaded areas correspond to unknown places and transitions that can be increased. First trial begins at P_0. When the state variables satisfy the necessary conditions, the system moves forward to make a stochastic decision on each transition. The procedure continues until a final place ($P_{F(success)}$) is achieved. Trials ending in $P_{F(fail)}$ indicate error. Retrieved from Zhao et al. (2017).

Hidden Markov Models

Hidden Markov Models (HMM) (Figure 3.3) are another type of graphs based on transition probabilities that can be used to encode task knowledge, following two key ideas: the first one states that the probability of transition from one state to another depends exclusively on that current state, not being dependent on any past events; the second one affirms that current and past states always remain unidentified, while what is visible is an observation that depends on a distribution probability specific to each state. Therefore, these structures are helpful in cases where the observed information is noisy, faulty or does not correspond to reality. Examples of research using these models are the work of Amit and Mataric (2002) and Butterfield et al. (2010).

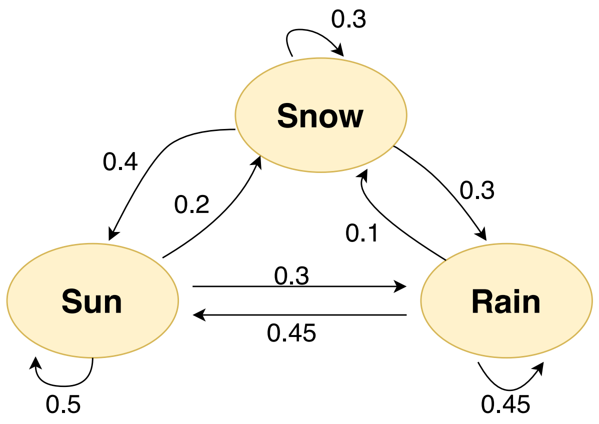


Figure 3.3: HMM Chain example with three states. It describes the transition probabilities between Snow, Sun and Rain. If the transition probability between Snow and Rain is 0.3, that means that if it was snowing yesterday, there is a 30% probability that it could rain today.

3.1.2 Hierarchical representations

Hierarchical representations consist of a top-bottom cascade structure describing the hierarchical organization of task representations (Figure 3.4). They are organized into a chain of different levels of abstraction and generally start with primitive actions such as motor actions (grab, plug, reach, pass, etc...) in the bottom (A-G), continuing with a division of the main task into representations of sub-tasks (e.g., red column plugged, yellow nut inserted, green ball inside the box) in the middle level, leading to an end-goal on the top (task completed). These structures are typically used to encode greater flexibility and facilitate changes in the demonstration order of the task and execution of more than one sub-task at the same time (Diedrichsen & Kornysheva, 2015). Nevertheless, accord-

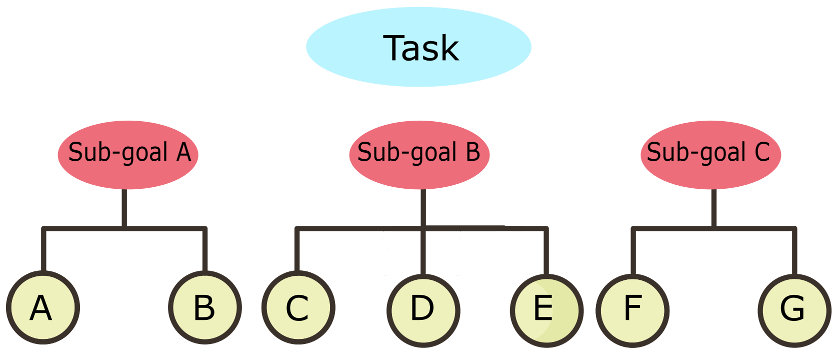


Figure 3.4: Hierarchical structure of a task divided into three levels.

ing to Sousa (2014), hierarchical representations might be hard to acquire from demonstrations, often depending on predefined task-knowledge or further clues given by the tutor. One example is the work of Miura et al. (2005), where the authors used a predefined hierarchical model organized from higher to low-level operations in a navigation task where the robot had to drive towards an elevator and move to the interior of it. The system questioned the human tutor to acquire specific scenario information - knowledge that could not be pre-programmed - while the human answered by giving specific directions. Garland and Lesh (2003), contrarily, used an annotations system in a cooking task simulation using labeled demonstrations that provided hierarchical information.

3.2 Cognitive models of sequential behavior and order

Sequential behavior in humans, a debate initiated by Lashley (1951), has been widely discussed in the fields of psychology, neuroscience and robotics, leading to several studies regarding multiple aspects of serial order. While acquiring and encoding task-knowledge for sequential tasks through demonstration, it is important to look at the cognitive models of sequential action to better understand how the stored information can be retrieved. This section intends to give an overview of the main cognitive approaches for serial order: chaining, ordinal and positional (Henson, 1998).

3.2.1 The chaining method

The chaining theory presumes that serial order is the result of associations between successive elements in a sequence. By retrieving a specific element, the next one is triggered according to a chaining process. This theory follows the stimulus-response theory in which each stimulus causes a response that, on its own, can turn into the stimulus for the next event (Lashley, 1951). This approach can be seen in the work of Lewandowsky and Murdock (1989), for example. Some issues were pointed to this theory for failing to answer some critical aspects of sequential behaviour in humans such as: 1) explaining how the first element can be triggered since, according to the same theory, events are cued by the ones proceeding it; 2) handling repetitions; and 3) how to recover from errors in the retrieving process (Lashley, 1951; Brown et al., 2000). The first issue was addressed by Murdock (1995) who proposed the use of external cues to trigger the first event in a recall sequence, while the next ones are addressed with the use of a Compound Chaining approach (Figure 3.5) where each event of a sequence is triggered by several past events as opposing to just the preceding one.

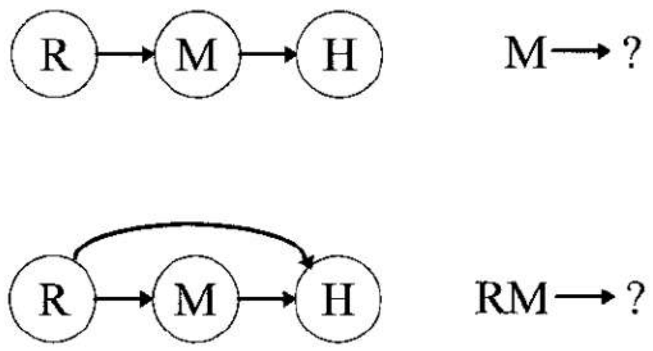


Figure 3.5: Sequence retrieval according to the Chaining approach. The Simple Chaining (top) implies that an event is only trigged by its predecessor while the Compound Chaining (bottom) method admits that each element can be cued by several preceding events. Retrieved from Henson (1998).

3.2.2 The ordinal theory

According to the ordinal theory, elements are represented in a single dimension where each event has a height value associated. The serial order is retrieved by relatively comparing and ordering the values associated to each element which results in a gradient from highest associated values to lower associated values, where the highest should be the next event in the recall process (Henson, 1998). A concrete implementation of this theory can be seen in the Competitive Queuing model proposed by Houghton (1990), in which competition between events and subsequent inhibition of activated events is implemented, being the element with the highest value the first to be recalled and so on. This theory is also the one used in the work of Ferreira et al. (2011, 2014) and Cunha et al. (2020) to learn the sequence order and timing of sequential tasks. For further reading, see Page and Norris (1998).

3.2.3 The positional theory

The positional theory assumes that serial order is stored by associating each event within a sequence with a position in an ordered list. The serial order is retrieved by using each position to trigger its associated event (position-item association) (Henson, 1998). This theory differs from the chaining approach in the fact that associations are not created between sequence events but between the positional representations of them. This approach handles the existence of repetitions by design, since sequence events can be associated with more than one position in the list. However, it also raises several critiques, mostly concerned with the lack of explanation on how the positional list itself is created (Brown et al., 2000).

3.3 Final notes

Learning from Demonstration has been proved to be a framework capable of providing mechanisms to endow robots with task-knowledge. Although several authors have investigated algorithms to encode sequential tasks based on demonstrations (see Pardowitz et al., 2007), only a few authors, such as Erlhagen et al. (2005, 2006), Ferreira et al. (2011, 2014), Sandamirskaya and Schöner (2010) and Sousa et al. (2009, 2014, 2015) have based their work on cognitive models of sequence learning in humans. Two of these works works form the basis of this dissertation: the work of the work of Ferreira et al. (2014), in which the robot AROS learns the sequence order and timing in a music task and Sousa et al. (2014) which also applied a DNF framework to allow the same robot to acquire task-knowledge in assembling a toy vehicle. The research carried on by these two authors and their colleagues motivates this work as a way to extend their models and further implement the same in different scenarios.

Part III

Materials and Methods

Chapter 4

Dynamic Neural Fields

The computational models implemented in this dissertation are based on the Dynamic Neural Fields theory. The following chapter provides an overview of the concepts and theoretical framework behind DNFs, as well as the key mathematical equations necessary to formalize the models.

4.1 General Introduction

Dynamical Neural Fields were primarily used in the robotics field by Schöner et al. (1995) to demonstrated the advantages in navigation and obstacle avoidance tasks within a behavior-based vehicle. Their work was followed by several authors in the robotics field, such as Bicho (1999), Bicho et al. (2000) and Monteiro et al. (2004) to name a few. Their approach presupposes that the system's behavior is internally represented by a set of behavioral variables, each one illustrating an individual component of behavior. For a more complete review on how behavior-based robots can be modeled by dynamical systems see Bicho (1999).

Engels and Schöner (1995) proposed a DNF-inspired solution to control the behaviour dynamics because of the cognitive skills that they allow to model. It was proved that DNF-framework provided key processing mechanisms for applications in cognitive robotics, such as memory, prediction, and decision making (Erlhagen & Bicho, 2006). The concept behind these models is that task-relevant information is expressed in the form of bumps above a threshold level of neural activation where each bump represents a specific behaviour. In the context of this work, each activity bump will encode a specific sub-task or sub-goal. The neuronal activity is initially triggered by external input sources - e.g, input from a vision system (Figure 4.1). Additionally, DNFs admit the existence

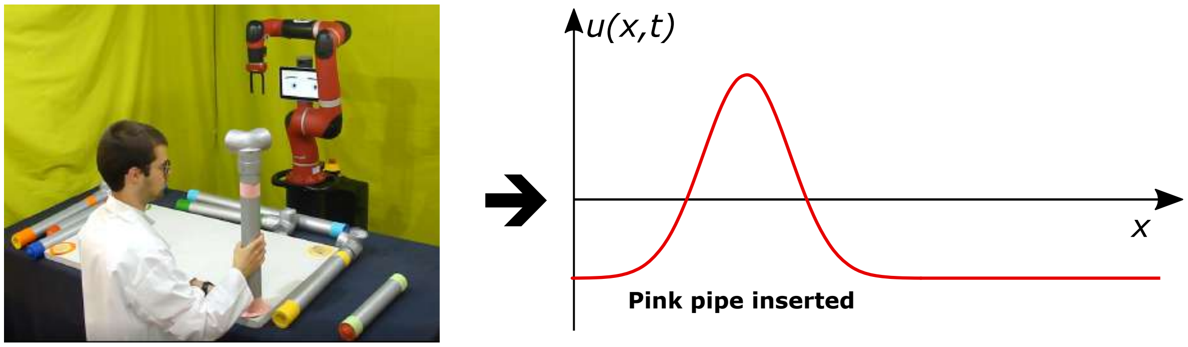


Figure 4.1: Example of a bump of activity triggered from visual input. The robotic system observes a pink pipe being inserted which generated a bump in the neuronal population encoding the pink pipe.

of auto-sustained bumps of neuronal activity that remain in the field in the absence of external input, due to the lateral interactions between neighbor neurons of the same layer (Amari, 1977). This feature enables the models with the capacity to accommodate cognitive skills that are fundamental when talking about robot learning in a more intuitive and human-like manner, such as working memory, decision making and learning associations between separated events (Curtis & Lee, 2010). The next sections describe the mathematical basis to implement DNF-models.

4.2 The Amari Equation

Proposed by Amari (1977) to formalize the pattern formation and interaction dynamics in cortical surface, equation (4.1) has been employed by robotic researchers to model processes of memory and decision making, since it can be analyzed analytically, it reduces computational complexity and its properties are capable to model cognitive abilities (Bicho et al., 2010). This approach follows the assumption that strong recurrent excitatory and inhibitory interactions in local neural populations form a basic mechanism for handling cortical information which causes dynamic behavior in neural aggregations.

The integro-differential equation bellow formalizes the dynamics of a single-dimension field where x,y symbolize spatial dimensions. The activity of neurons is summarized into an activation function $u(x,t)$ where u represents the activation of a neuron positioned in the location x at time t, and the constant parameter $\tau > 0$

controls the time scale of the field:

$$\tau \frac{\partial u(x,t)}{\partial t} = -u(x,t) + h + s(x,t) + \int w(x-y) f_0[u(y,t)]dy. \tag{4.1}$$

The equation is divided into a linear (4.2) and a non-linear part (4.3). To facilitate its comprehension, one can start by analyzing its linear part, where $s(x,t)$ corresponds to the input given to a neuron x at the time t. The parameter $h < 0$ determines the resting level to which the neuronal activity relaxes without the external input s. $s(x,t) > 0$ happens when information corresponding to the variable encoded in a specific sub-neuronal population presents activity, that is, when external input triggers excitatory activity in the same. Contrarily, $s(x,t) \leq 0$ occurs when the same activity is non-existent or presents inhibitory behaviour:

$$\tau \frac{\partial u(x,t)}{\partial t} = -u(x,t) + h + s(x,t). \tag{4.2}$$

In addition, the features of Amari's equation that allows it to model cognitive behaviour result from its non-linear term that describes the interactions between the sub-neuronal populations of the same field (Erlhagen & Bicho, 2014):

$$\int w(x-y) f_0[u(y,t)]dy. \tag{4.3}$$

Depending on the distance between them, those sub-populations of neurons can either excite or inhibit each other (Figure 4.2), following an interaction pattern that is defined by the functions f_0 and w (Amari, 1977). $f_0(u)$ is

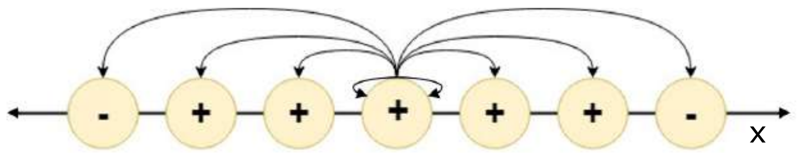


Figure 4.2: Relations between neurons of the same layer. Populations of neurons closer to the ones at location x are excited while others at longer distances are inhibited.

used as a gating function to ensure that only neurons with activity above a specific threshold contribute to the interaction ($f_0[u(y,t)] > 0$). In this case, following Amari's formulation, the function can be mathematically implemented using a Heaviside step (threshold value = 0):

$$f_0(u) = \begin{cases} 1, & u \geq 0 \\ 0, & u < 0 \end{cases} . \tag{4.4}$$

4.2.1 Types of Interaction kernels

The strength of each interaction is described by the kernel term function $w(x - y)$. Bellow, three types of coupling kernel functions are described and may be used according to the desired type of lateral interaction between neurons (Ferreira et al., 2016).

Gaussian kernel with global inhibition

Figure 4.3 depicts the Gaussian kernel function which is defined by the following equation:

$$w(x - y) = Ae^{\frac{-(x-y)^2}{2\sigma^2}} - w_{inh}, \tag{4.5}$$

where $A > 0$ and $\sigma > 0$ control the amplitude and standard deviation of the gaussian curve, respectively, and w_{inh} represents a positive constant quantifying the inhibition strength (Erlhagen & Bicho, 2006). This means

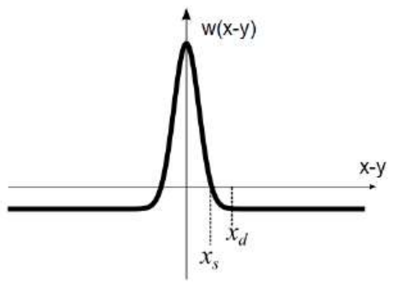


Figure 4.3: Gaussian kernel. A neuron excites another if the distance between them is smaller than x_s and inhibit it otherwise. If the distance is bigger than x_d, the inhibition strength is constant.

that, when the lateral interaction is defined by this type of kernel, a neuron will cause excitatory behaviour in neurons where the distance is less than x_s and will cause inhibitory behaviour otherwise, with an approximately constant inhibition strength for neurons at distances bigger than x_d. Coupling a Dynamic Neural Field with this

Gaussian function enables the existence of competition between sub-neuronal populations from where only one localized peak of activity should evolve.

Mexican-hat kernel

Continuing, the equation bellow implements a *Mexican-Hat* kernel which is given by the difference of two Gaussian functions, having the amplitudes $A > B > 0$ and the standard deviations $\sigma_b > \sigma_a > 0$ (Ferreira et al., 2016):

$$w(x-y) = Ae^{-\frac{(x-y)^2}{2\sigma_A^2}} - Be^{-\frac{(x-y)^2}{2\sigma_B^2}}. \tag{4.6}$$

This type of kernel allows the existence of simultaneous peaks of activity if the distance between the corresponding sub-populations of neurons is larger than x_d (Figure 4.4).

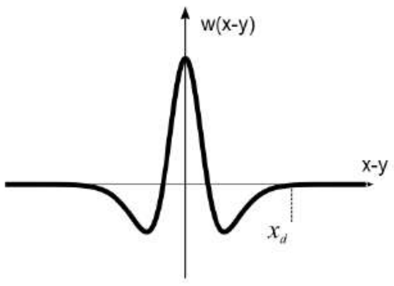


Figure 4.4: Mexican-hat kernel. It admits the coexistence of more than one peak of activity if they are located at distances longer than x_d.

Oscillatory kernel

The third and last kernel function (Figure 4.5) is an Oscillatory kernel defined by the equation (4.7), where $k > 0$ determines the rate at which the oscillations in w decay with distance. $A > 0$ and $0 < \alpha \leq 1$ control the amplitude and the spatial phase of w, respectively:

$$w(x-y) = Ae^{-k|x-y|}\left(k \sin|\alpha(x-y)| + \cos(\alpha(x-y))\right). \tag{4.7}$$

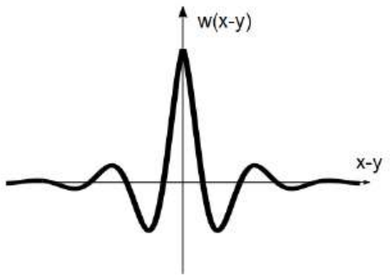


Figure 4.5: Oscillatory kernel. It guarantees the existence and stability of multiple activity bumps with smaller excitatory interactions at larger distances.

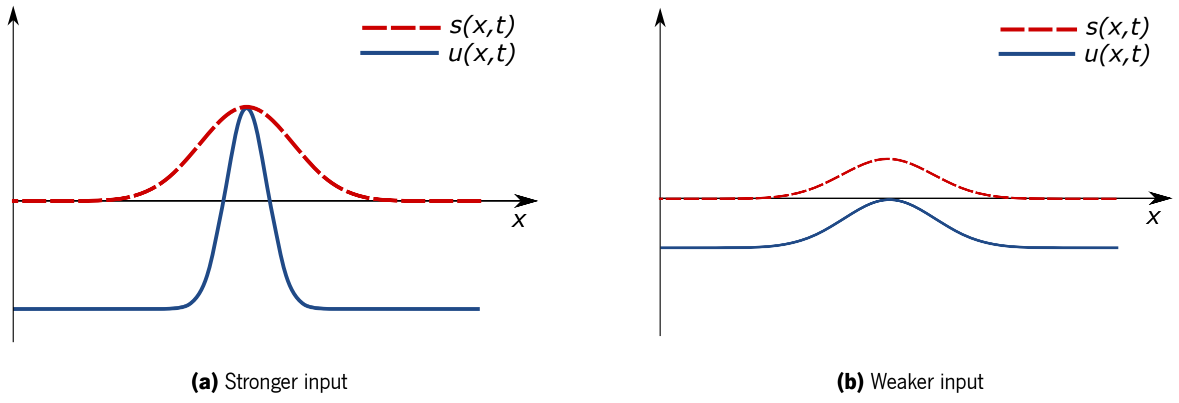
This coupling function admits the coexistence and stability of multiple bumps of activity, allowing the existence of excitatory interactions in neurons located also at larger distances (Laing et al., 2002; Ferreira et al., 2016).

4.3 Cognitive capabilities modeled through DNFs and the Amari equation

In 2006, Erlhagen and Bicho proposed a DNF-based architecture demonstrating how the mathematical framework of Dynamic Neural Fields can be adopted to implement cognitive functions in the field of robotics, such as memory and forgetting, decision making, goal inference and prediction, which inspired several related work in the topic, like Bicho et al. (2010), Ferreira et al. (2014), Sousa et al. (2015), to name a few. For the relevance of this dissertation, the focus lays on the cognitive capabilities of memorizing, forgetting and decision making.

Self-sustained bumps of neuronal activity

As mentioned in previous sections, the lateral interactions between neighbor sub-populations of neurons allow the field to hold self-sustained activity bumps in the lack of external input. However, this only happens if the strength of the input driven by external sources takes the corresponding population of neurons to a value above the predefined threshold level, triggering the interaction between neuronal populations and allowing the bump to remain self-sustained (Figure 4.6).



(a) Stronger input **(b)** Weaker input

Figure 4.6: Illustration of the interaction between $u(x,t)$ and $s(x,t)$. In (a) the input is sufficiently strong to drive the sub-neural population above the threshold level (zero) and a bump of activity evolves, while in (b) the external input is weaker, thus not able to form a self-sustained bump.

Memory and forgetting

The ability to auto-sustain an activity bump allows the field to behave as a memory representation, since it can represent the activation corresponding to a feature even after the input that caused it is no longer available.

Similarly, for cases where this memory representation is no longer necessary, it is also possible to eliminate the memory representation by suppressing the same activity. This can be achieved by controlling the global inhibition input parameter h (Figure 4.7). For sufficiently negative values, $u(x,t)$ is too low and the self-excitation is no longer sufficient to sustain a positive activity bump (Figure 4.7a) (Amari, 1977). For levels of h closer to zero, the field can have two different states of stability (Figure 4.7b): if the input has triggered a positive peak of activation and is then removed, the field maintains a localized stable bump of activity. If no input has been active, the general resting level persists. To summarize, a neuron will present activity if the total balance of excitatory and

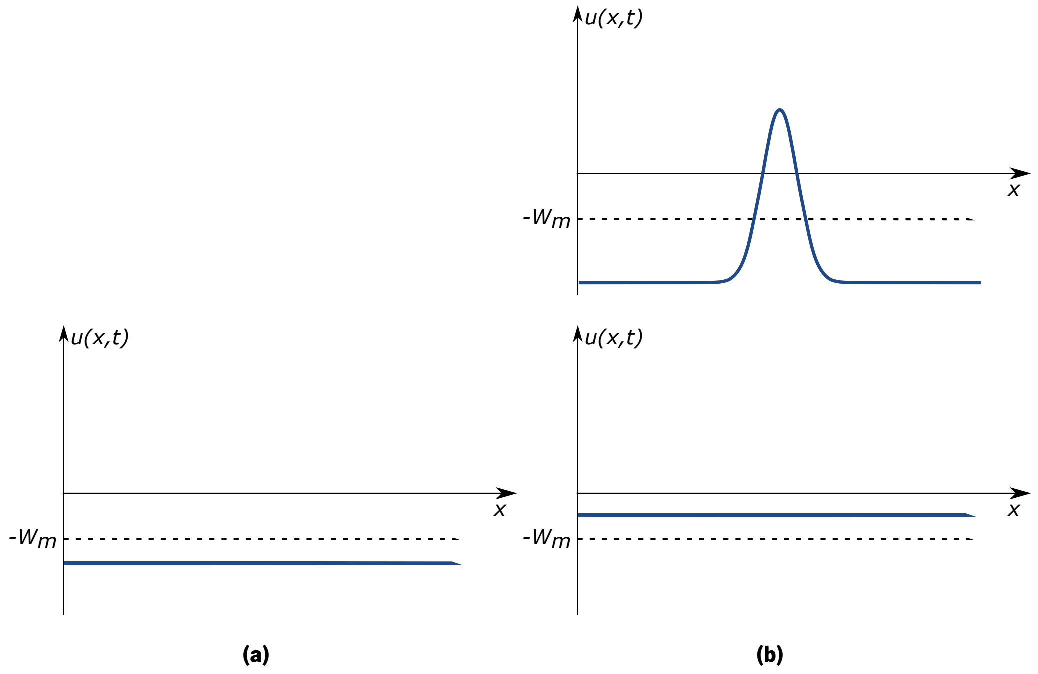


Figure 4.7: Different stable states of the field dynamics, in the absence of external input, as a function of the resting level. In (a), $h < -W_m$, the field cannot self-sustain a peak of activity without input, so $u(x,t) = h$. In (b), $h > -W_m$, the field is able to sustain a bump of activity after the triggering input is no longer available.

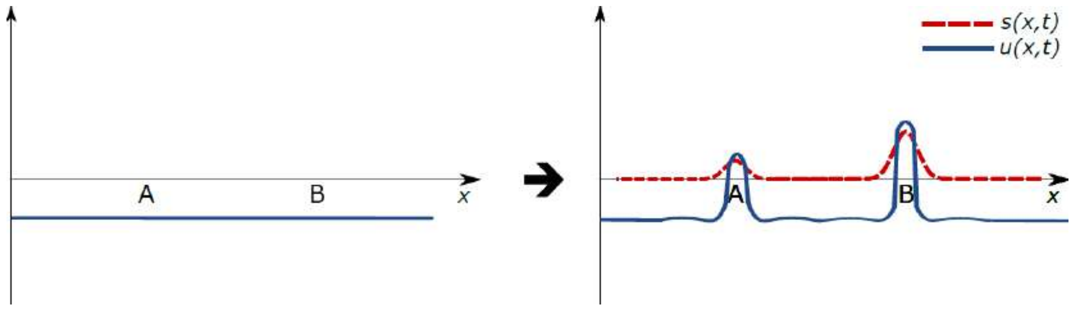
inhibitory input is strong enough to keep it above the threshold level 0. This means that there is a “critical” value for h bellow which the field is not able to maintain a self-sustained bump:

$$h < -W_m = -max_x \left\{ \int_0^x w(x-y)dx \right\}. \tag{4.8}$$

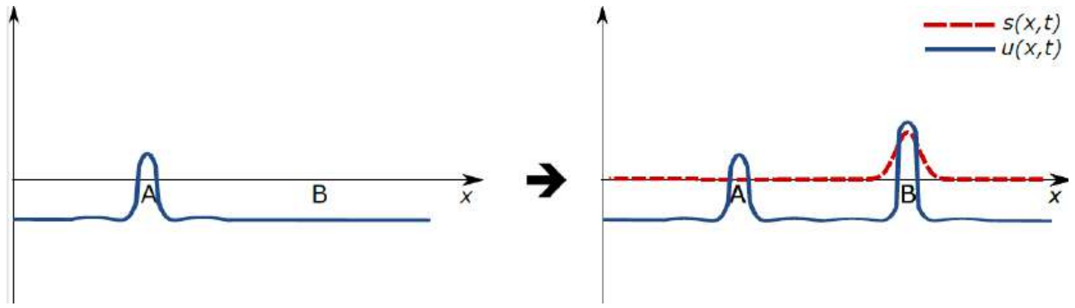
Thus, a localized memory bump can be eliminated by lowering h below the level of $h < -W_m$ (Bicho et al., 2000).

Multi-item Memory

If the field is coupled with an Oscillatory kernel function (equation(4.7)), it admits the coexistence and stability of multiple bumps of activity, so multi-items can be memorized in the same field. Figure 4.8 presents a DNF field able to store multiple peaks. In Figure 4.8a, two simultaneous inputs trigger activity above the threshold level 0. Since the field was coupled with an Oscillatory kernel, both peaks of activity generated were sustained. In Figure 4.8b, the field already had memorized an event when the second input appeared, and once again, both bumps are able to coexist in the field.



(a) Memorizing two simultaneous events

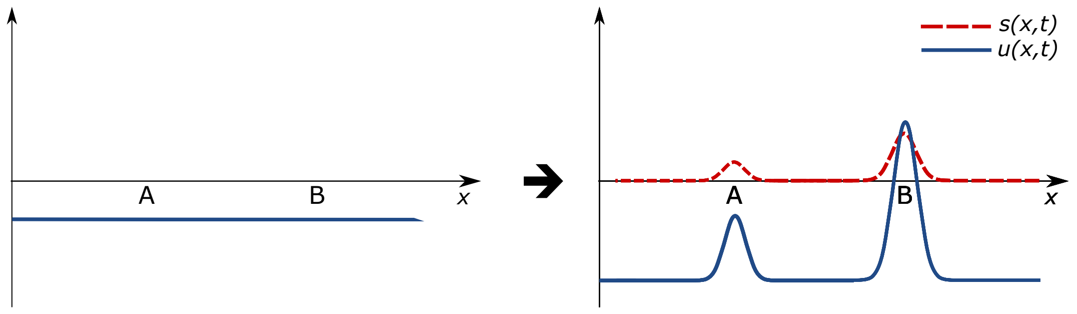


(b) Memorizing a second event

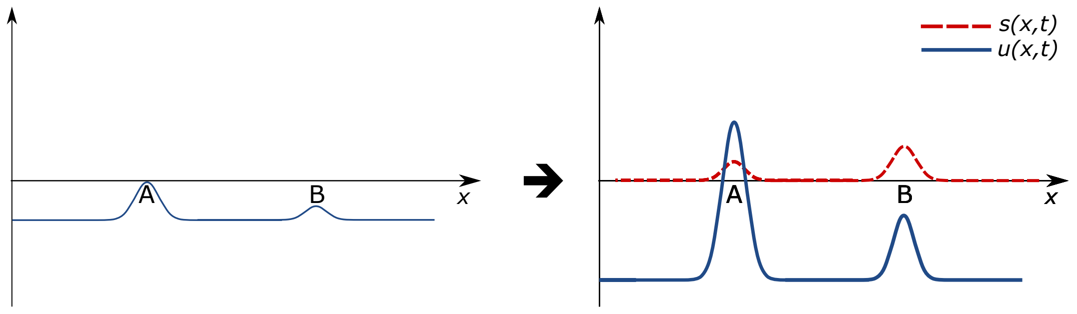
Figure 4.8: Example of DNF field with the ability to store multi-bumps of activity. In (a), two simultaneous inputs triggered two different bumps of activity who were sustained in the field. In (b), although the field already had a memorized bump, when the second one appears, both are kept in the field in the form of memory.

Decision Making

In the presence of sufficiently strong lateral inhibition, another cognitive skill that can be implemented is the ability of making decisions. To formulize this cognitive skill, it is necessary to use a Gaussian kernel function with global inhibition (equation (4.5)) to trigger the competition between neighboring populations. Figure 4.9 shows two different cases where two peaks compete to win the decision in the process, leading to a single location peak of activation positioned either in the sub-neuronal population A or B. Since both bumps are sufficiently distant, the



(a) Winining decision: B



(b) Winining decision: A

Figure 4.9: Example of decision making using DNFs. Initially, in (a) there is no pre-shaping activity so the peak with the highest input (B) wins the lateral competition. In (b) the field already has a pre-shaping value, so although the highest input was in B, the total activity favored the decision to A.

lateral interactions will result in one inhibiting the other. The "winning" bump will emerge based on two factors: (1) the bump with the highest input level will have advantage over the other (Figure 4.9a); and (2) the initial state of the field may give the advantage to one peak over the other (Figure 4.9b).

Chapter 5

Artificial Neural Networks

Chapter 4 provided the necessary background to understand how neuronal populations in the DNFs theory can model cognitive behaviour. This chapter introduces Artificial Neural Networks (ANN) as a tool to accommodate learning capabilities in the DNFs-models. It presents the basic functioning of neural networks and how ANN can be integrated with Dynamic Neural Fields.

5.1 Introducing Neural Networks

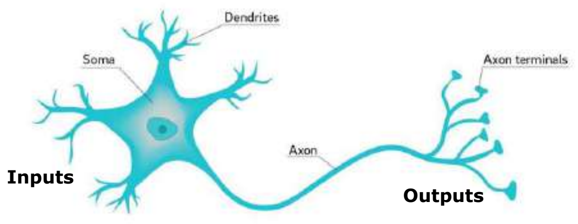
The interest on neural networks emerged after McCulloch and Pitts (1943) presented a simplified model of biological neurons, demonstrating how they could perform computational tasks. ANN may be described as *"intelligent, thinking machines, working in the same way as the animal brain"* (Lek & Guégan, 1999), able to learn from experience, extracting complex features from observable data that would be rather difficult or even impossible to encode "by hand" and thus working as powerful classifying mechanisms. They can be used to solve complex computational problems and process data with great speed. In these networks, calculations are done in a parallel chain, reducing the necessary time to process large amounts of data. Due to those characteristics, Artificial Neural Networks have been extensively used as a computational tool in several fields, such as banking, computer sciences, financial, medical and speech (Lek & Park, 2008), to name a few.

The relevance for the computer engineering fields lays on the possibility to process large amounts of sensorial data and apply learning capabilities to extract information and make predictions (Zador, 2019), thus allowing the development of more accurate control policies when dealing with noisy information. Furthermore, when associated to Dynamic Neural Fields, the neural interactions can be shaped as a function of input data, conditioning the network response, meaning that an input-output mapping function can be recreated through learning from examples, in the absence of explicit knowledge of the function (Bicho et al., 2010; Sousa et al., 2015).

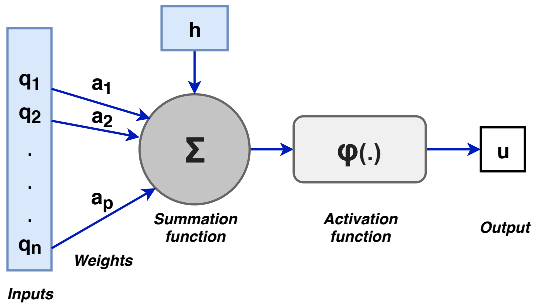
5.2 Architecture of Neural Networks

5.2.1 Basic functioning

The basic unit of the brain is the neuron. Likewise, the basic computational unit of any neural network is also a neuron (also referred as node). Figure 5.1 depicts a schematic of a biological neuron (Figure5.1a) and its mathematical representation (Figure 5.1b). Each input source is represented by a node connected to the neuron through the corresponding synaptic connection, weighted by a_i. Additionally, an external input h is applied to introduce a positive or negative shift to the total input s. Following the aforementioned Amari's equation, h represents a negative shift to the input s ($h < 0$). The activity level u depends on the external input and input from neighboring neurons, with m being the total number of input sources. Each one of those sources, q_i, contributes to the total input with a different weight, a_i, with $i = 1, 2, 3...n$. These relations can be described



(a) Biological neuron



(b) Mathematical model

Figure 5.1: Schematic comparing the structure of a biological neuron and its mathematical model. The neuron receives several inputs from different sources and computes an output. Each input has an associated weight, a, based on its relative importance to other inputs. After, a function is applied to the weighed sum of inputs.

by the equation below:

$$s = \sum_{i=1}^{m} q_i a_i. \tag{5.1}$$

Subsequently, the signal resulting from the previous sum, s, is modeled by an activation function, $\varphi(.)$, which limits the output of the neuron and characterizes its overall response:

$$u = \varphi(\sum_{i=1}^{m} q_i a_i + h). \tag{5.2}$$

Several formulations can be used for this function, being the most prevalent ones the Heaviside and the Sigmoid functions. A simple network can be modeled by a binary activation function of two states - "On" (1) or "Off" (0) - where the output will depend on the value of the several inputs, following a linear separation such as the one modeled by the Heaviside step function (4.4). However, with an Heaviside function, a small change in weights or bias can drastically change the output value, going from 0 to 1 or vice versa. Therefore, to obtained a smoother network behavior, it is necessary to select an activation function where small deviations in the weight or bias provokes small corresponding deviations in the network output. That way, it is possible to adjust, step by step, the values of weights and bias towards the best approximation. A sigmoid function (5.3) can be used to model the former behaviour since it presents smoother deviations with a range of outputs from 0 to 1, with σ being the slope value of the function:

$$\varphi(s) = \frac{1}{1 + e^{-\sigma s}}. \tag{5.3}$$

5.2.2 Organization

Artificial Neural Networks are formed by several neurons organized into layers (Figure 5.2). Input signals may be shared by several neurons that can function independently or interact with other neurons, similarly to what happens in a DNF (Sousa, 2014). The basic mathematical formulation of a single neuron (5.2) can be lengthened to target a set of neurons in the same layer:

$$u_j = \varphi(s_j + h_j), \tag{5.4}$$

with u_j being the activity in a neuron j that results from the input s_j, given by the equation

$$s_j = \sum_{i=1}^{m} q_i a_{i,j}, \tag{5.5}$$

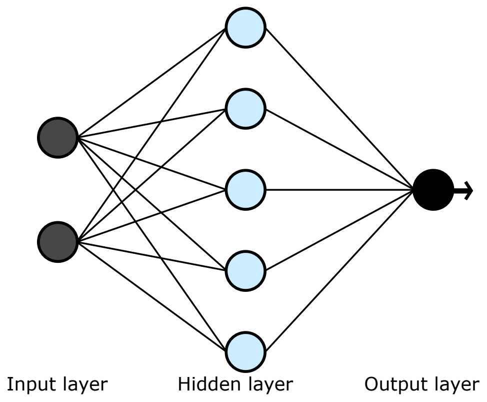
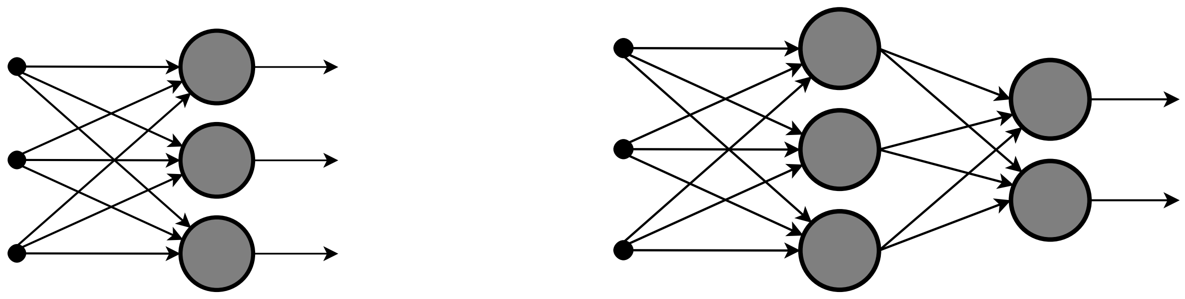


Figure 5.2: Example of a neural network with two input nodes. The leftmost layer is called the input layer containing the input neurons contributing to the network. The rightmost is the output layer that holds the output neurons (in this case there is only a single output neuron but there could be multiple). The middle layer is labelled as hidden layer - designation for any layer between the input and output layers.

where $a_{i,j}$ represents the weight of the synaptic connection of the input i to the neuron j. Generalizing, the former equation can be written in the form of a matrix, resulting in the matrix of synaptic weights **A**:

$$\mathbf{A} = \begin{bmatrix} a_{1,1} & a_{1,2} & \dots & a_{1,n} \\ a_{2,1} & a_{2,2} & \dots & a_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m,1} & a_{m,2} & \dots & a_{m,n} \end{bmatrix}. \tag{5.6}$$

ANN can be categorized into single-layers or multi-layer networks (Figure 5.3). The neuronal activity in single-layer networks is directly propagated from input nodes to output nodes, thus each neuron fires immediately in the presence of a specific value of input (Figure 5.3a). In a multi-layer topology, there are hidden layers between input and output nodes where the output of each hidden-layer works as input for the next layer - another hidden-layer or the output layer (Figure 5.3b). Those are used in applications where a linear separation provided by a single-layer network topology is not sufficient to classify the data. Worth noting, different network organizations require different analysis and learning rules to adjust the synaptic weights. Although the architecture followed in this work incorporates several layers, the synaptic weights are only adjusted between neighbor layers, so the learning rules are directed to a single-layer topology. The next sections introduce some of the most common types of learning in neural networks.



(a) Single-layer network **(b)** Multi-layer network

Figure 5.3: Single-layer and multi-layer networks.

5.3 General paradigms of learning

This section gives a quick overview of three common mechanisms used for adapting the weights of a network to achieve the desired behaviour. Each one of the approaches is based on a different learning paradigm.

Supervised Learning

The aim of supervised learning is to build a model that makes predictions, based on evidences, in the presence of uncertainty. In this type of learning paradigm, there is a desired output for each particular input. This means that the training data is already "labeled" with the correct answers for each output. The network receives the input and produces an output response that is compared to the desired output to generate an error value. Then, the weights are adjusted in order to minimize the value of that error function (Argall et al., 2009).

Unsupervised Learning

This learning method opposes the previous one since the learning takes place without supervision, meaning that there is not a predefined output intended for each input. The learning mechanism looks for statistical patterns and adjust the weights accordingly. Clustering is the most common unsupervised learning technique, used to find hidden patterns or groupings in the training data. This can be helpful in situations where it is either impossible or impractical for a human to cluster the data by its own, thus unsupervised learning provides initial observations that can then be used to assess individual hypotheses (Sammut & Webb, 2017).

Reinforcement Learning

Reinforcement learning seeks to maximize a numerical reward signal. The networks does not know what is the desired output and instead must discover which paths bring the biggest reward value, using a trying-error search. The network is inserted in a closed-loop with the environment where the output influences the environment state which is then feed into the network. At the same time, an evaluation of the state is performed, providing a reinforcement signal to control the weight adaptation (Sutton & Barto, 1998).

5.4 Hebbian Learning

"Neurons that fire together wire together", Hebb (1949). Hebbian Learning - also knowns as associative learning - was introduced by Donald Hebb as a learning paradigm inspired in the biological procedure of the neural weight adaptation, more concretely, in the dynamics of synaptic adaptation between two cells. In his postulate, the author affirms:

"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."

His research indicates that when two cells are simultaneously active, the synaptic connection between them is reinforced so that activity in one expedites activity in the other. That means that the weight value connecting two neurons increases if they are activated simultaneously and decreases otherwise. The simplest mathematical formulation of the Hebbian rule is described by following equation:

$$\Delta a_{i,j} = \eta q_i u_j, \tag{5.7}$$

affirming that the synaptic weight variation, $\Delta a_{i,j}$, connecting the pre-synaptic neuron i to the post-synaptic j is equal to the multiplication of the input q_i and output u_j signals by a rate parameter η. Although this equation obeys to the Hebbian principle and forms the ground for several weight adaptation rules, this rule is rarely used since it implies an exponentially increase or decrease in the weights, which brings limitations in terms of stability (Chen et al., 2001). Its continuous application turns the network unstable since the weights grow indefinitely, regardless of their current value. An alternative weight adaptation is presented in section 5.5.1.

5.5 Learning in a Dynamic Neural Field

Although in Artificial Neural Networks neurons are analyzed as discrete units, ANN formulations can be applied to the continuous dimensions modeled in Dynamic Neural Fields, permitting the accommodation of neuronal adaptation rules normally adopted in neural networks. Erlhagen and Bicho (2006), Sousa et al. (2009), Sandamirskaya (2014) and Sousa et al. (2015) have applied neuronal-based learning mechanisms to DNFs-based models, forming the groundwork of learning mechanisms and adaptation rules used in this dissertation. In his work, Sousa (2014) explains that, when transferring adaptation rules designed for neural networks to make associations between DNFs, two features must be adapted: 1) the equations that define the activity propagation from pre to post-synaptic neurons; 2) the adaptation rules controlling the weights of the synaptic connections. Both aspects are addressed by the following equations which are based in the work developed by Sousa (2014).

To initiate the transition from discrete neurons to the behavioral dimensions of DNFs, it is necessary to consider the behavioral dimensions x^{in} - for input nodes - and x^{out} - for output nodes - as a line of discrete locations x_i with a constant spatial distance between neurons. Consequently, the subsequent equivalences can be made:

$$u_j^{out} \equiv u(x_j^{out}), \tag{5.8}$$

$$s_j^{out} \equiv s(x_j^{out}), \tag{5.9}$$

$$q \equiv q(x_i^{in}), \tag{5.10}$$

being u_j^{out} and s_j^{out} the activation and the input, respectively, of a neuron at location x_j^{out}. Next, equation (5.5) that describes the propagation of activity from an input layer to an array of neurons, can be rewritten as:

$$s(x_j^{out}) = \sum_{i=1}^{m} q(x_i^{in}) a(x_i^{in}, x_j^{out}), \tag{5.11}$$

where $a(x_i^{in}, x_j^{out})$ represents the synaptic weight connection from the input node at the location x_i^{in} and the output node at x_j^{out}. Continuing, it is also necessary to adapt equation (5.11) to accommodate the propagation of activity between two fields. For that, let's represent the activation of two coupled fields by $u^{in}(x_i^{in})$ - input field - and $u^{out}(x_i^{out})$ - output field. The propagation from the input field to the output field can be formulized by replacing $q(x_i^{in})$ with $u^{in}(x_i^{in})$ in the former equation:

$$s^{out}(x_j^{out}) = \sum_{i=1}^{m} u^{in}(x_i^{in}) a(x_i^{in}, x_j^{out}). \tag{5.12}$$

Another feature that must be encoded in the former equations is the incorporation of a gating mechanism to control the propagation of activity from on field to another. This is achieved by introducing a threshold function $f_{\lambda^{in}}$ implying that only activity above a certain level, λ^{in}, contributes to the interaction between fields:

$$s^{out}(x_j^{out}) = \sum_{i=1}^{m} f_{\lambda^{in}}[u^{in}(x_i^{in})] a(x_i^{in}, x_j^{out}). \tag{5.13}$$

Finally, there are a few steps left to allow the former equation to reflect the continuous domain of Dynamic Neural Fields:

1. Minimize the distance between the spatial location of the neurons to reach a boundlessly distribution;
2. Replace the summation in the equation (5.13) for the integration of the dimension x^{in};
3. Introduce the time dependency according to the Amari field equation.

Thus, the result can be formulized by the following equation:

$$s^{out}(x^{out}, t) = \int f_{\lambda^{in}}[u^{in}(x^{in}, t)] \underset{in \to out}{a}(x^{in}, x^{out}, t) dx^{in}, \tag{5.14}$$

which computes s^{out} as a result of the propagation of activity from another field. Also, the matrix of synaptic weights was altered to incorporate the variation in time of the weight values, which will vary according to the learning rule adopted, explored next.

5.5.1 The Delta Rule

A different weight adaptation rule, incorporating information about the current network state and the desired output, was suggested by Widrow and Hoff (1988). The authors proposed a Least-Mean Square (LMS) algorithm, commonly used in optimization problems where the objective is to minimize a cost function. The learning policy adopted uses the Delta Rule, which differs from the Hebbian rule (equation (5.7)) in the replacement of the output u_j by an error term e_j :

$$\Delta a_{i,j} = \eta q_i e_j, \tag{5.15}$$

in which e_j computes the difference between the target output u_j^{tar} and the total amount of input, given by the equation (5.5):

$$e_j = u_j^{tar} - \sum_{i=1}^{m} q_i a_{i,j}. \tag{5.16}$$

Following the analysis made in section 5.5, once again it is necessary to transit the former equation to the continuous domain. Similarly to the equivalences (5.8) and (5.9), the error term will be given by:

$$e_j \equiv e(x_j^{out}). \tag{5.17}$$

Subsequently, since the input considered for the network is given by the activation of the field, the input term q_i is replaced by $f_{\lambda^{in}}[u^{in}(x_i^{in})]$, and the Delta Rule can be now written as:

$$\Delta a(x_i^{in}, x_j^{out}) = \eta f_{\lambda^{in}}[u^{in}(x_i^{in})]e(x_j^{out}). \tag{5.18}$$

Then, the previous equation is formalized in the continuous space by the equation

$$\tau_a \frac{\partial}{\partial t} \underset{in \to out}{a} (x^{in}, x^{out}, t) = f_{\lambda^{in}}[u^{in}(x^{in}, t)]e(x^{out}, t), \tag{5.19}$$

where the rate term is replaced by the time constant $\tau_a > 0$ that defines the rate variation in the synaptic weights. Considering that the error term is defined by the difference between target output and the total amount of input to a neuron, given by equation (5.14), the last step is to represent the target output. As stated before, a threshold must be applied to ensure that only the desired activation values influence the variation of the synaptic weights. For that, the target output can be given by $g_{\lambda^{out}}[u^{tar}(x^{out}, t)]$ with $g_{\lambda^{out}}$ being defined by the function

$$g_{\lambda^{out}}[u] = \frac{[u - \lambda^{out}]^+}{1 + [u - \lambda^{out}]^+}, \tag{5.20}$$

where λ_{out} is the threshold value. Finally, the error equation can now be fully written:

$$e(x^{out},t) = g_{\lambda^{out}}(u^{tar}(x^{out},t)) - \int f_{\lambda^{in}}[u^{in}(x^{in},t)] \underset{in\rightarrow out}{a}(x^{in},x^{out},t)dx^{in}. \tag{5.21}$$

5.6 Final remind

The core methods that form the theoretical background of this work - Dynamic Neural Fields in Chapter 4 and Artificial Neural Networks in the current Chapter - were reviewed, demonstrating how it is possible to incorporate learning in the interaction between fields. Next are introduced the experimental setup and the robotic platform used to implement the work featured in the following Parts of this dissertation.

Chapter 6

Robotic platform and joint construction task

As a concrete scenario to implement and test the models developed, a human-robot join task scenario was selected. This chapter starts by describing the robotic platform used in this work and some of its features. Next, the joint construction task is presented, as well as the key questions addressed during the collaborative work between robot and human tested in this dissertation: 1) what do to (what part should be used next); 2) who does it and how (what action should be done over the selected part, and who should do it); and 3) when to do.

6.1 The Robot Sawyer

Sawyer (Figure 6.1) is a robotic platform designed by the company "Rethink Robotics" and commercialized since March 2016. It consists of an articulated arm with 7 degrees of freedom and 1.26 meters reach, designed to perform collaborative tasks (Rethink Robotics, 2018). The difference between traditional industrial robots and collaborative robots like Sawyer is the way they are presented to people. While traditional robots are usually placed inside cages, having no contact with human operators, Sawyer has inherent safety features that allow it to be operated alongside people[1]. This safety net, together with his appealing visual look, makes this robot better accepted between human operators. The LCD display that sits on its top (its "head") depicts a pair of animated

[1]Sawyer is certified according to ISO requirements by TÜV Rheinland (ISO 10218-1:2011 and PLd Cat. 3).

eyes capable of representing different emotions that give Sawyer a humanoid appearance and make it appealing to work with, contributing to a more natural and user-friendly interaction (Rethink Robotics, 2018). The robot also has two cameras, one in the head (RGB) and one in the arm (Grayscale) that may be used to provide information about object color and type. Table 6.1 shows some of Sawyer's specifications.

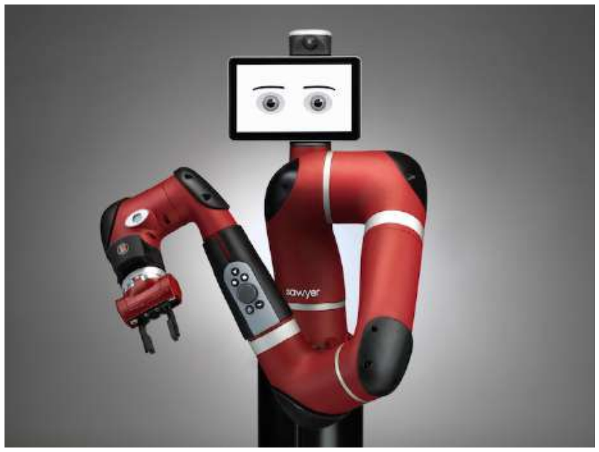


Figure 6.1: Sawyer: Robotic platform from "Rethink Robotics".

Table 6.1: Main specifications of Robot Sawyer

Pay Load	Max Reach	Gantry Reach	Task Repeatability	Typical Tool Speed	DoF	Robot Weight
4kg	1260mm	900mm	$\pm$ 0.1mm	1.5 m/s	7	19kg

The arm movements and trajectories used in the experiments result from the trajectory replication demonstrated by the human through "Kinematics teaching", available in Sawyer's software. In the first stage, the DNF-models used in this work were implemented and simulated in Matlab. A Python script was developed to make the communication between the Matlab computations and the Robot Sawyer in real-time, using the Robotic Operation System (ROS) framework. Also, a speech synthesizer was used to allow the Robot to communicate with the human by reproducing pre-defined words or sentences provided as text strings.

6.2 Joint Construction Task

As a concrete application scenario to validate the work developed, a task consisting of building a structure of eight different-colored thrusters was selected. The task was segmented into eight sub-tasks/sub-goals, where inserting each thruster on top of the Base corresponds to one sub-goal. The structure is composed by the Base (BA), four bottom-parts - Green (G), Orange (O), Pink (P) and Yellow (Y) - and four other pieces that are inserted in the top - Blue (B), Light Blue (LB), Light Green (LG) and Red (R) -, forming the structure displayed in Figure 6.2. The eight thrusters must be all in place to finalize the structure, which can be done through any order, as long as the physical constraints of the structure are respected. For example, the Light Blue thruster cannot be inserted without placing the Orange and the Pink thrusters that form its support. This particularity makes the construction task a very flexible one that can be achieved through multiple construction sequences.

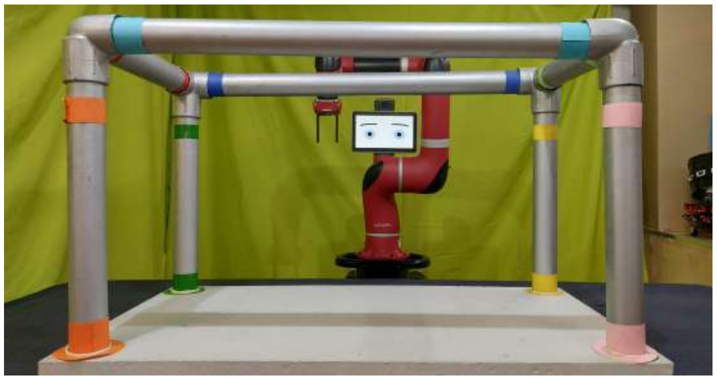


Figure 6.2: Structure consisting of one Base and eight colored parts: Green, Orange, Pink, Yellow (bottom) and Blue, Light Blue, Light Green, Red (top).

6.2.1 Learning the sequential structure of the task - What piece should be placed next?

The first part of the experiment focuses on acquiring, from tutor demonstration, the sequential structure of the task. The schematic displayed in Figure 6.3 shows eight possibilities of different sequences of steps to assemble the structure. One can start by inserting the Pink thruster (Figure 6.4a) and other can start by the Orange one (Figure 6.4b), for example. Therefore, for an efficient Human-Robot Collaboration, the robotic co-worker needs to be able to assist its human partners in every possible scenario, that is, regardless of the sequence of steps used by its different co-workers. Consequently, when working together with a human, it is fundamental that the robot knows the sequential structure of the task, that is, the precedent relations between the several sub-tasks, to be able to assist its co-worker, independently of the sequence of steps he may use.

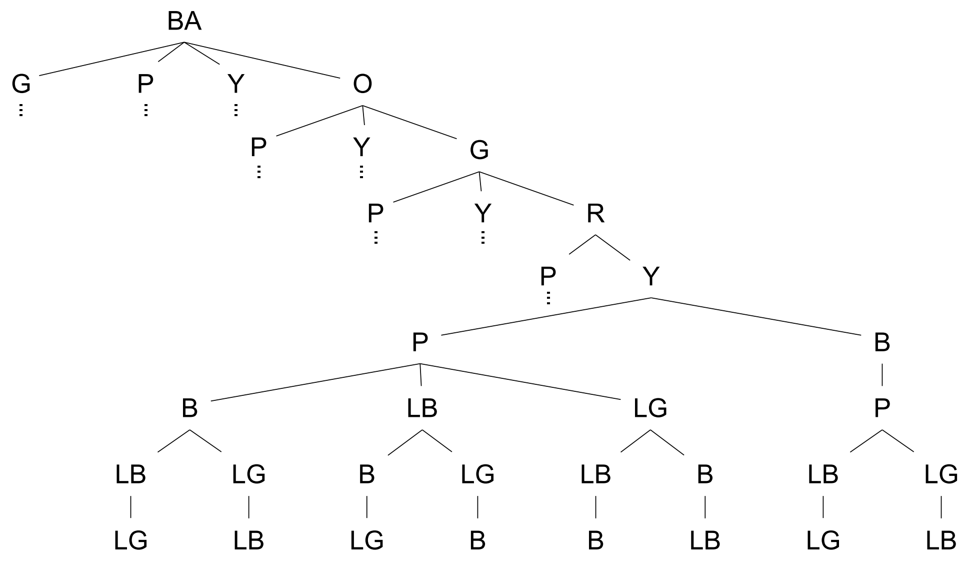


Figure 6.3: Schematic of the possible sequences to construct the structure. For simplicity, the schematic only shows 8 of the 928 possibilities of different sequences that could be used.

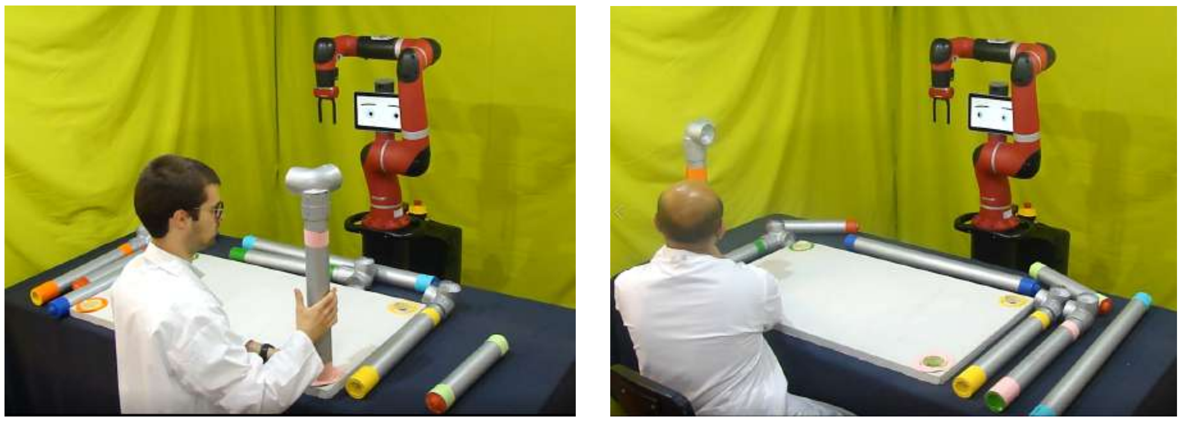


Figure 6.4: Different operators building the structure starting from different parts of the structure. In (a), the co-worker starts to construct the right side of the structure by inserting the Pink thruster. In (b), a different operator starts building the left side of the structure by inserting the Orange thruster.

6.2.2 Incorporating workspace information and action selection - What action should be done over the selected part and who should do it?

In the second part of the experiment, the pieces were distributed through three different workspaces: Robot Workspace, Human Workspace and Shared Workspace (Figure 6.5). In this phase, besides selecting the next piece to be inserted, the robot also needs to select the best suitable action to take, considering the end purpose (insert a piece in a given place), which will depend on two factors:

1. Who should grab the next piece? - Sawyer can grab pieces from the Robot and Shared Workspaces. The Human can reach pieces in the Human and Shared Workspaces;

2. Who should insert the piece? - Due to its physical constraints, the robot can only reach the placing location of the Green, Orange and Yellow thrusters. The other parts are assumed to be inserted by the Human.

Consequently, it is expected that Sawyer and its human co-workers work together and coordinate their actions to insert all the parts and complete the task.

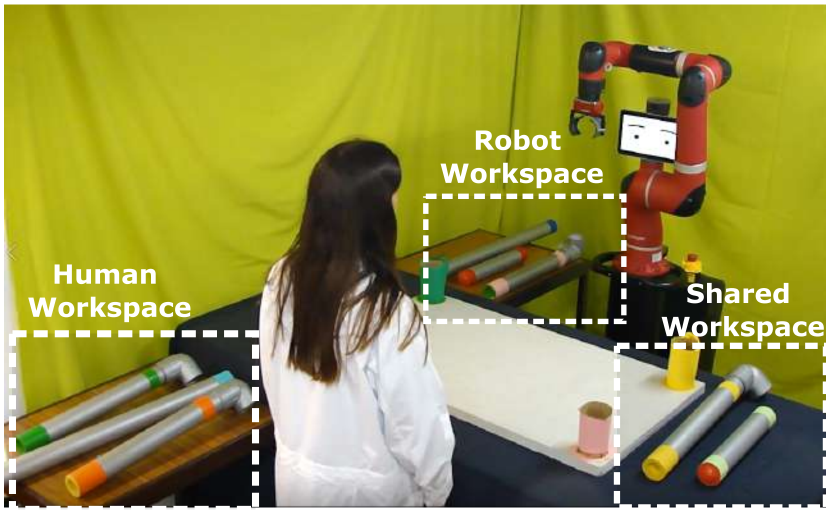


Figure 6.5: Human, Robot and Shared Workspaces. The Human and Robot Workspaces consist of pieces placed in the tables next to the human operator and next the robot, respectively. They cannot reach pieces in the other co-worker's workspace. The Shared Workspace is formed by the pieces laying on the construction table - where lays the Base -, reachable by both partners.

6.2.3 Memorizing sequences with time constraints - When to do?

A third experiment was conducted to validate a model for memorizing ordinal and temporal information of the sequential task in a single demonstration. The model was tested as a primary step towards learning sequences that may have time constraints. A tutor starts by demonstrating the sequence of steps to build the structure, starting from different layouts of the pieces in the construction table (Figure 6.6 - left). Each different layout implies the use of a different construction sequence.

In each trial, the (observing) robot memorizes the serial order and the time interval between each assembly step and associates the initial layout and the characteristic of the tutor that performed the demonstration - for that, tutors with different behavior time scales (older/younger) were selected. When facing a new (inexperienced) operator (Figure 6.6 - right), the robot selects one of the memorized trials according to the layout of the pieces and the characteristics of the new co-worker and recalls the memorized information according to the serial order and time previously demonstrated.

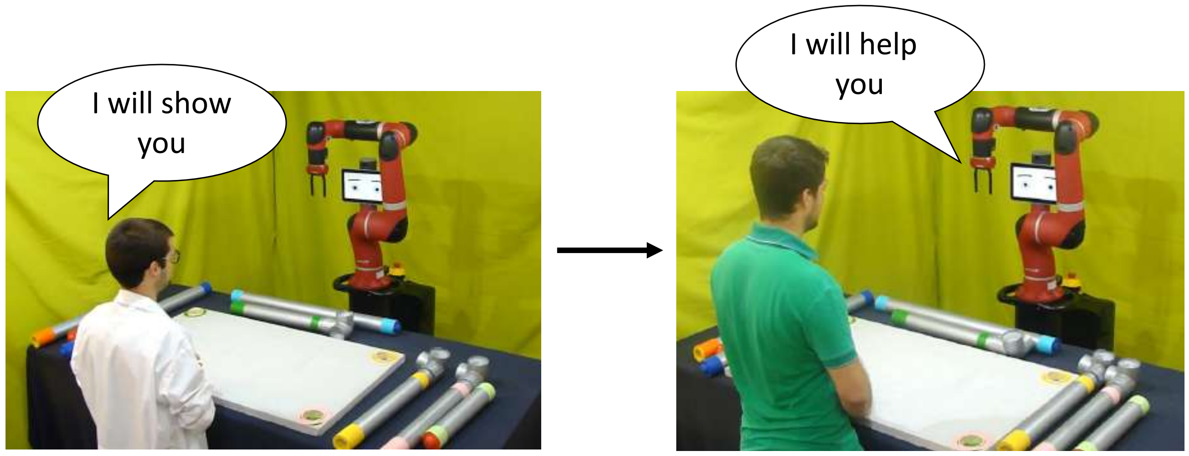


Figure 6.6: Sawyer evolves from a learning position to a teaching position. On the left, Sawyer receives information from an experienced tutor. On the right, Sawyer uses the memorized information to instruct inexperienced co-workers facing the same trial context.

Part IV

Implementation and Results

Chapter 7

Learning the Sequential Structure of an Assembly Task

The present chapter describes the implementation and results of the learning model that allow Sawyer to acquire task-relevant information of sequential tasks. It begins with a description of the mathematical implementation of the model, followed by the results of learning the sequential structure of the construction task described in Chapter 6.

7.1 Dynamic Neural Field Model for Sequence Learning

In section 2.3, it was seen that task-relevant information could be learned as a sequence of sub-goals/sub-tasks that must be achieved, respecting its serial order constraints. Facing a flexible task with multiple assembly sequences (Figure 6.3), the method selected for encoding task-knowledge was a compound chaining approach (see section 3.2.1). The activation of a specific sub-goal is triggered by several previously achieved sub-goals, bringing flexibility to the model to deal with the several possible sequences. As a method to acquire task information, the Learning from Demonstration paradigm is applied. Figure 7.1 depicts a flowchart of the system used to teach the robot. First, during the Learning Mode, the tutor demonstrates to the robot the necessary steps to complete the task using different serial orders. The Rehearsal mechanism allows the robot to internally "think" about the observed sequence and use that information to set the weights of the learning rule. This process will be explained in further sections. Then, during the Recall Mode, the robot attempts to reproduce the task and receives verbal feedback from the tutor. If the feedback is positive, it means that the robot was able to retrieve all sub-goals and successfully learned the task. Negative feedback from the tutor indicates that the robot made an error during the recall process and it will imply adjusting the learning mechanism and repeat the demonstration trial. This process continues until the robot makes no more errors.

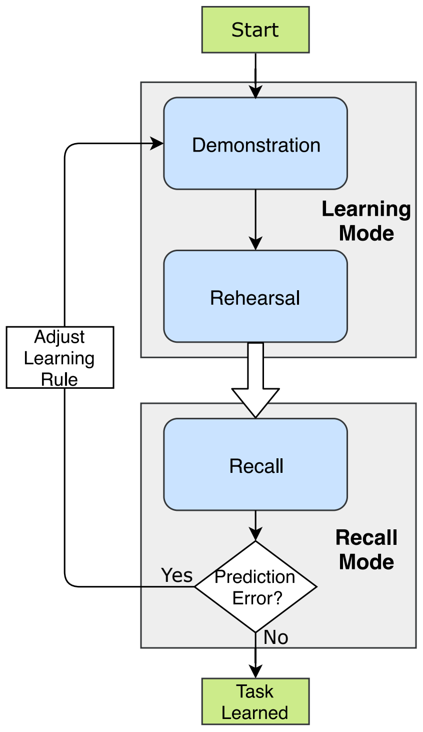


Figure 7.1: Flowchart of the learning process. It starts with the Learning mode, where the tutor demonstrates the sequence - Demonstration -, followed by the (internal) Rehearsal of the learned steps by the robot. Then, in the Recall mode, the robot retrieves the learned sequence, and the tutor provides feedback to adjust (or not) the weights of the learning rule until the robot retrieves the task without fail.

In order to acquire the necessary sequence of sub-goals, the first step is to differentiate sub-tasks that were already performed (e.g., the already assembled pieces) from the ones that still need to be executed. Following the work of Sousa et al. (2014, 2015), an architecture for representing "Past" and "Present" sub-tasks was implemented, separating already completed sub-goals from the ones yet to be achieved. The model consists of two interconnected dynamic neural fields that reflect changes in the task observed by the robot and form the base of the Sequence Learning Layer (SLL) (Figure 7.2). The Past field behaves as a "working" memory of the current state of the task while the Present field marks observed or predicted sub-goals. The dynamics connecting both fields allow learning the task by associating sequential achieved sub-goals. This mechanism will be explained further ahead.

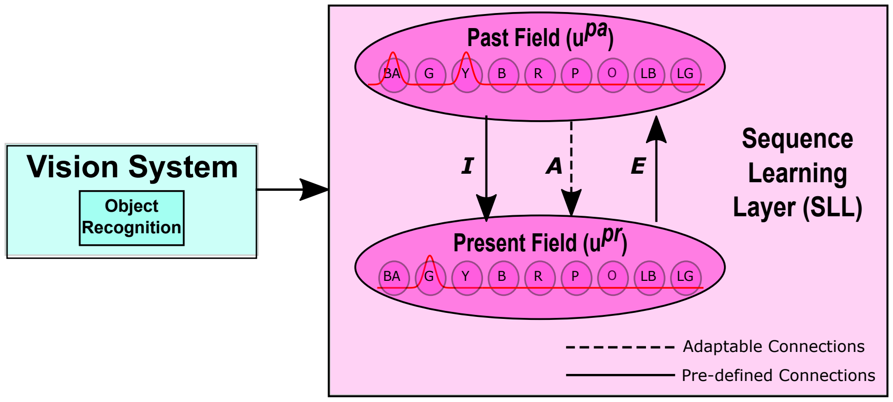


Figure 7.2: Diagram of the Sequence Learning Layer (SLL). It consists of two coupled dynamic neural fields u^{pa} - Past Field and u^{pr} - Present Field. I and E correspond to the inhibitory connections from u^{pa} to u^{pr} and the excitatory connections from u^{pr} to u^{pa}, respectively. A represents the adaptable connections from u^{pa} to u^{pr} encoding the relations between sub-goals learned through the various observations.

7.1.1 Input from the Vision System

The Vision System is the main source of sensorial input in this work, providing information about the state of the task over time. In the Sequence Learning Layer, the Vision System allows the recognition of the objects that are being inserted, triggering the achievement of each sub-goal (Figure 7.3). The execution of sub-goals is codified in the DNF model. Each field contains sub-neuronal populations that encode the current state of each sub-goal, in the form of a Gaussian function centered in each sub-neuronal population. The visual input is mathematically described as $v^k(x^k, t)$, where $k = pa$ or $k = pr$ if the input contributes to the neuronal activation of the Past Field $u^{pa}(x^{pa}, t)$ or to the Present Field $u^{pr}(x^{pr}, t)$, respectively.

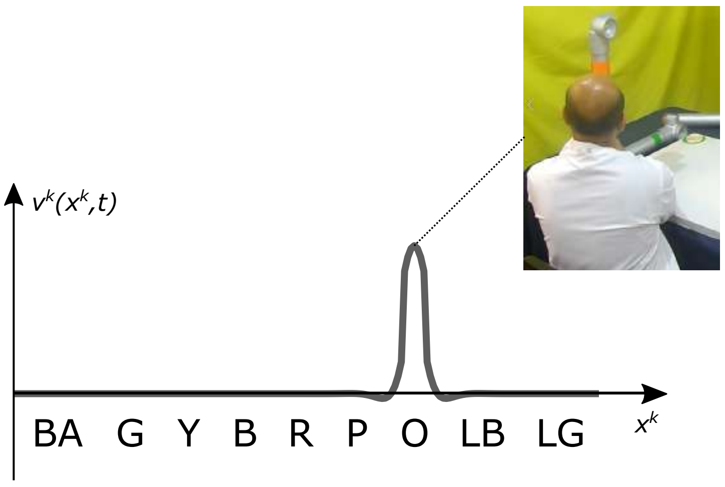


Figure 7.3: Example of input from the Vision System. The observation of a sub-goal triggers activity in the sub-neuronal population encoding the specific event, represented by a Gaussian pattern.

7.1.2 Sequence Learning Layer

The Sequence Learning Layer incorporates the base learning model of this work with the purpose of encoding task-relevant information. Figure 7.4 shows a diagram of the connections between the two fields. Three different types of synaptic interactions connect the two dynamic neural fields u^{pa} and u^{pr}.

Predefined connections

- $\underset{pa\to pr}{I}(x^{pa}, x^{pr})$: Inhibitory synaptic interaction from u^{pa} to u^{pr};
- $\underset{pr\to pa}{E}(x^{pr}, x^{pa})$: Excitatory synaptic interaction from u^{pr} to u^{pa}.

The predefined connections couple sub-neuronal populations in u^{pa} and u^{pr} corresponding to the same sub-goal. A sub-goal with activity above a threshold level $u^{pr} > \lambda^{pr} > 0$ in the Present Field - observed through the visual input - will trigger activity in the corresponding sub-population of neurons in the Past Field through the excitatory connection $\underset{pr\to pa}{E}$ - blue arrows in Figure 7.4. Likewise, sub-goals with activation in the Past field above the threshold level 0 - $u^{pa}(u^{pa} > 0)$ - are inhibited in the Present Field by the inhibitory connection $\underset{pa\to pr}{I}$ - red dashed arrows in Figure 7.4 -, marking the accomplishment of the corresponding sub-task.

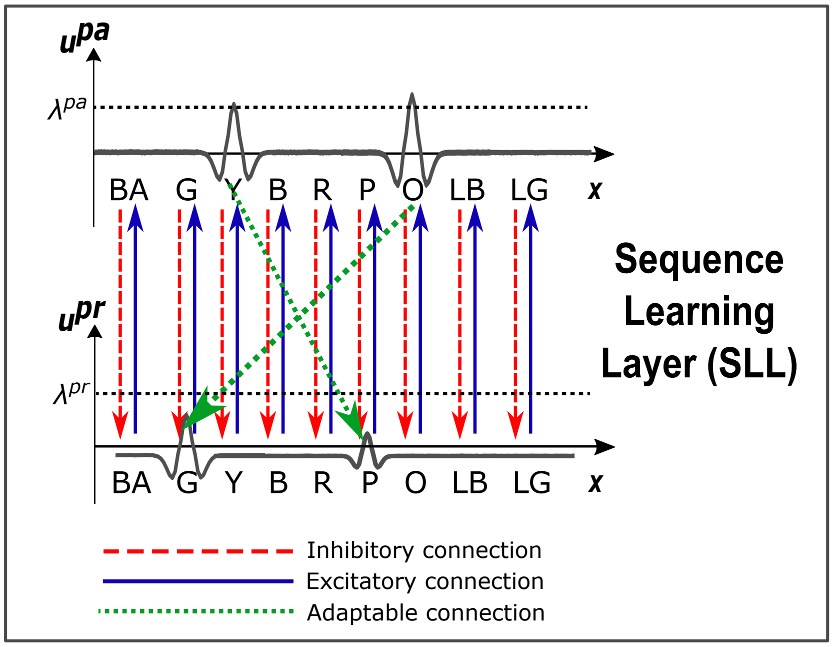


Figure 7.4: Diagram of the connections between u^{pa} and u^{pr} in the Sequence Learning Layer.

Adaptable Connections

- $\underset{pa \to pr}{A}(x^{pa}, x^{pr}, t)$: Excitatory synaptic interaction from u^{pa} to u^{pr}.

The adaptable synaptic connection from the Past to the Present fields $\underset{pa \to pr}{A}$ - green dashed arrows in Figure 7.4 - encodes the relations between the several sub-goals. The Delta rule is applied (see section 5.5.1) to establish the synaptic connections between sub-neuronal populations from u^{pa} to u^{pr}. The learning and adaptation of those connections happens when the same sub-population presents simultaneously activity above the threshold levels λ^{pa} and λ^{pr}, in u^{pa} and u^{pr} respectively (Seitz & Dinse, 2007).

Through the dynamic behavior between both fields, the SLL models the two key working modes of the system (see Figure 7.1):

- **Learning Mode** - SLL acquires the sequential structure of the task from tutor's demonstrations, encoded in the Past and Present Fields;

- **Recall Mode** - After learning, and having the information about already achieved sub-goals (Past Field), the model can make predictions (in the Present Field) regarding what should be the next sub-goal.

Both fields receive input from the Vision System - v^{pa} and v^{pr}. However, the field's response to the visual input is not the same and depends on the resting level h of each field. In the Present field, the resting level has a constant value h^{pr} that ensures that the visual input by itself is sufficient to trigger activity above the threshold level λ^{pr}. Similarly, λ^{pr} allows distinguishing activity representing observed events that will contribute to the Learning mechanism from smaller peaks of activity that may result from predictions made by the Recall mechanism. In the Past field, the resting level is a function $h^{pa}(x^{pa}, t)$ that admits two different baseline values, depending on each mode (Learning or Recall) the system is operating.

Learning Mode

In the Learning mode, the baseline value h^{pa}_{low} is lower than in the Recall mode, which means that the u^{pa} response to the visual input is slower, allowing the corresponding bump in the u^{pr} to reach the threshold λ^{pr} before the same sub-population evolves in u^{pa} (Figure 7.5). These slower dynamics permits the existence of a time window during which sub-populations in u^{pa} and newly evolving ones in u^{pr} present activity above the corresponding threshold levels simultaneously and become linked. The visual input triggers activity in $u^{pr}(t_0)$. Then, u^{pr} reaches λ^{pr} (t_1) and excites the corresponding population in u^{pa} that continues to grow due to the

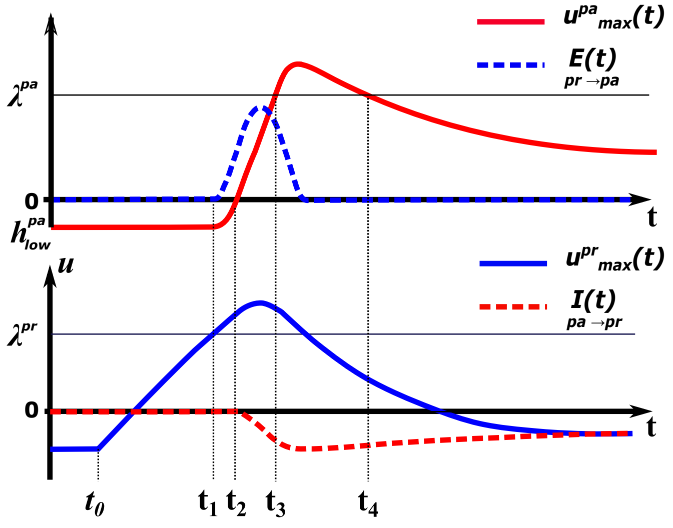


Figure 7.5: Lower resting level in u^{pa} during Learning.

recurrent interactions within the population. When u^{pa} reaches the threshold 0 (t_2), it starts to inhibit thes activity in u^{pr} that converges to its resting level. From t_3 to t_4, u^{pa} is above the threshold level λ^{pa}, forming the time window where new emerging sub-goals in u^{pr} become associated to the former sub-goal.

Recall Mode

Contrarily, during the Recall mode, the robot must have a faster response to take an active role in retrieving the learned sub-goals. For that, the resting level is placed higher h^{pa}_{high} (Figure 7.6) so the u^{pa} field responds faster to the visual input and the formed bump instantly inhibits the corresponding sub-population in u^{pr}, preventing the formation of a peak and accelerating the recall process (Sousa et al., 2015). A higher resting state of u^{pa}

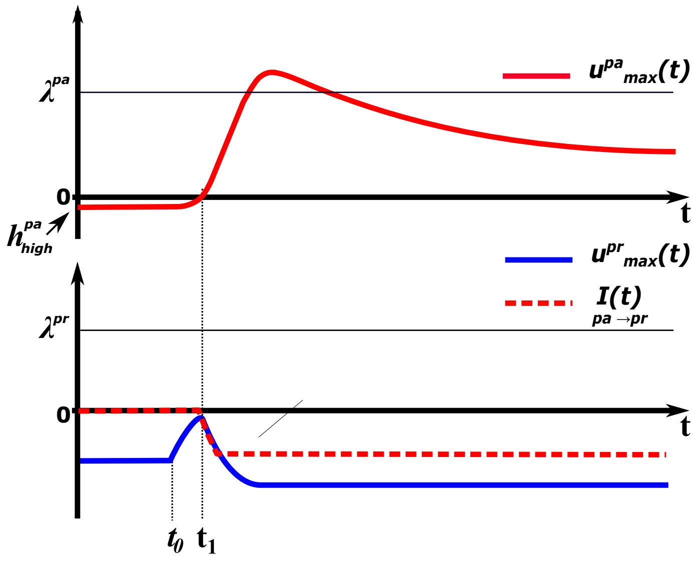


Figure 7.6: Higher resting level in u^{pa} during Recall.

brings populations closer to the activation threshold and thus accelerates the formation of a bump in response to visual input (t_0). In (t_1), u^{pr} excites the corresponding population in u^{pa} that instantly inhibits the activity of u^{pr}.

After achieving λ^{pa}, the dynamics of the field, in both cases, causes a continuous decay in the activity of the corresponding sub-neural populations. However, the activity stabilizes above threshold level 0 due to the recurrent interactions within the populations, symbolizing the achievement of the sub-goal, and maintaining a working memory of the task.

7.1.3 Associative Learning

It has been shown that learning the task is more than just memorizing an observed sequence of sub-goals. To acquire the sequential structure of the task, an associative learning mechanism based on Hebbian learning is applied, allowing the system to acquire the dependences between the several sub-goals. Those connections are saved in the form of a matrix $\underset{pa \to pr}{A}$. The basic concept of this mechanism relies on the existence of an interval of time during a demonstration where sub-populations of neurons have activity above the Learning thresholds - λ^{pr} and λ^{pa}- in both fields.

Since the activity in u^{pa} decreases over time, the memory representation of the performed sub-goals becomes weaker, so only recently performed events will become linked to the next one observed. Depending on how long is the interval of time during which an activity bump stays above the threshold level in u^{pa}, more or less already achieved sub-goals will become associated with the next one observed in u^{pr}. Several methods can be used to control this association: changing the decay rate of u^{pa} (Sousa et al., 2014), controlling the gain of the response to the visual input or changing the value of the learning threshold λ^{pa}, for example. For simplicity in the visualization, the latter method was selected. Figure 7.7 illustrates the associative process with two different values of λ^{pa}. Lower threshold values will allow the bumps of activity to remain longer above the threshold line so more past sub-goals in u^{pa} will become linked to the next one in u^{pr} (Figure 7.7b). When demonstrating different sequences, different associations will be made, extracting the several precedence relations between sub-goals.

During the Recall mode, the robot retrieves the sequence of sub-goals observed during the demonstration period and learned during the Rehearsal mode. If the robot makes a wrong prediction (Figure7.8), the learning threshold λ^{pa} is lowered to allow more past events to become associated with a specific prediction in the next demonstration. This process of learning-recalling is repeated until the robot makes no more errors.

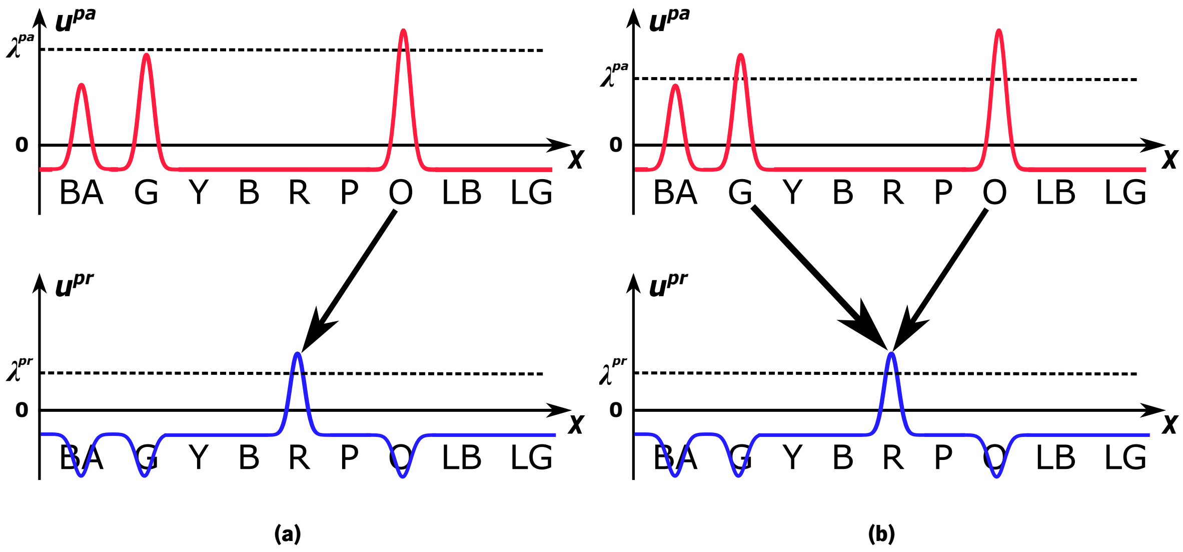


Figure 7.7: Associative learning controlled by the threshold level λ^{pa}. In (a), with a higher threshold level, the model only makes a simple chaining association (O$\longrightarrow$ R). By lowering the threshold value (b), the time window for associative learning is increased, allowing a compound chaining association (O$\longrightarrow$ R) and (G$\longrightarrow$ R).

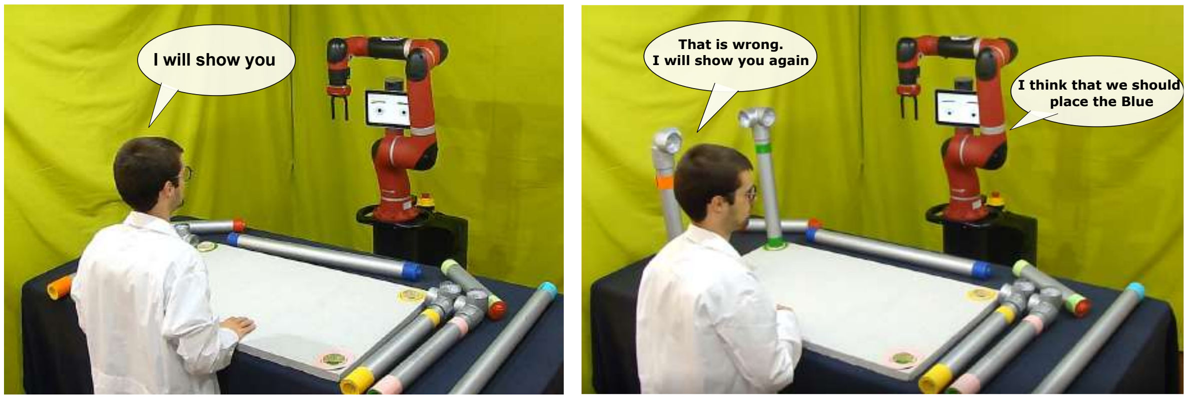


Figure 7.8: Learning and Recalling Process. If during the Recall Mode (on the right) an error occurs, the threshold value λ^{pa} is lowered to allow more associations between sub-goals in the next demonstration trial.

7.1.4 Mathematical formulation of the Sequence Learning Layer

The mathematical base of each layer of the model is a Dynamic Neural Field, formalized by an adaptation of the Amari equation (see section 4.2):

$$\begin{aligned}\tau^k \frac{\partial u^k(x^k,t)}{\partial t} &= -u^k(x^k,t) + h^k(x^k,t) + \varsigma^k_{stoch}(x^k,t) \\ &+ f_\beta[u^k(x^k,t)]\left(s^k(x^k,t) + \int w_k(x^k - y^k) f_0[u^k(y^k,t)]dy^k\right),\end{aligned} \tag{7.1}$$

with k representing each field and u_k being the activation of a neuron x_k. $\tau_k > 0$ and h_k are the time constant and the field resting level to witch u_k converges when there is no external input, respectively. To ensure that u_k does not take excessively negative values facing strong inhibitory inputs, an additional gating term is employed $f_\beta[u^k(x^k,t)]$, ensuring that the field's activity never reaches values below $\beta < 0$. Additionally, a noise function ς^k_{stoch} is applied to force the competition between neurons with identical input values.

Next follows a detailed explanation of the mathematical formulation of the Past and Present fields.

Past Field

The equation below describes the activity of the Past field u^{pa}:

$$\begin{aligned}\tau^{pa} \frac{\partial u^{pa}(x^{pa},t)}{\partial t} &= -\, u^{pa}(x^{pa},t) + h^{pa}(x^{pa},t) + \varsigma^{pa}_{stoch}(x^{pa},t) \\ &+ f_\beta[u^{pa}(x^{pa},t)]\left(s^{pa}(x^{pa},t) + \int w_{pa}(x^{pa} - y^{pa}) f_0[u^{pa}(y^{pa},t)]dy^{pa}\right).\end{aligned} \tag{7.2}$$

The interaction kernel function w_{pa} used is an Oscillatory kernel (equation (4.7)) since it admits the coexistence and stability of multiple bumps of activity. The implementation parameters for this field can be consulted in the Appendix A.1.

Input to the Past field: s^{pa}

The input $s^{pa}(x^{pa},t)$ is defined by the sum of the external visual input $v^{pa}(x^{pa},t)$ with a hand-coded gain $C^{v\rightarrow pa}$ and the excitatory contribution from the Present field given by the integral term:

$$s^{pa}(x^{pa},t) = C^{v\rightarrow pa} v^{pa}(x^{pa},t) + \int f_{\lambda^{pr}}[u^{pr}(x^{pr},t)] \underset{pr\rightarrow pa}{E}(x^{pr},x^{pa})dx^{pr}. \tag{7.3}$$

The integral term formalizes the propagation of the excitatory connections from u^{pr} to u^{pa} where only activity above the threshold λ^{pr} is propagated to u^{pa}, applied through the function $f_{\lambda^{pr}}$.

Resting level: $h^{pa}(x^{pa}, t)$

Following aection 7.1.2, the baseline activity of u^{pa} differs in the Learning and Recall modes. The equation below controls the adaptation of the resting level value h^{pa} towards which the field rests in the absence of input, depending on which mode the robot is operating:

$$\begin{aligned}\tau_h^{pa}\frac{\partial h^{pa}(x^{pa},t)}{\partial t} &= \Big(1 - f_0[u^{pa}(x^{pa},t)]\Big)\Big(H_{level}^{pa} - h^{pa}(x^{pa},t)\Big) + \\ &+ f_0[u^{pa}(x^{pa},t)]\Big(H_{dec}^{pa} - h^{pa}(x^{pa},t)\Big),\end{aligned} \tag{7.4}$$

where τ_h^{pa} is the time constant of the resting level decay and H_{level}^{pa} defines the value of the resting level depending on the mode, given by:

$$H_{level}^{pa} = \begin{cases} H_{high}^{pa}, & \text{if mode} = \text{recall} \\ H_{low}^{pa}, & \text{if mode} = \text{learning} \end{cases}. \tag{7.5}$$

When $u^{pa} > 0$, the value of $h^{pa}(x^{pa}, t)$ decays from H_{level}^{pa} to H_{dec}^{pa}. Contrarily, when $u^{pa} \leq 0$, it converges to H_{level}^{pa}.

Present Field

The Present Field is formalized by the following equation:

$$\begin{aligned}\tau^{pr}\frac{\partial u^{pr}(x^{pr},t)}{\partial t} =& - u^{pr}(x^{pr},t) + h^{pr}(x^{pr},t) + \varsigma_{stoch}^{pr}(x^{pr},t) \\ &+ f_\beta[u^{pr}(x^{pr},t)]\Big(s^{pr}(x^{pr},t) + \int w_{pr}(x^{pr}-y^{pr})f_0[u^{pr}(y^{pr},t)]dy^{pr}\Big),\end{aligned} \tag{7.6}$$

where h^{pr} is a constant function and the interaction kernel w_{pr} is modeled by a Gaussian curve with global inhibition (equation (4.5)) to enable the competition between sub-neuronal populations.

Input to the Present Field: s^{pr}

The input to the field u^{pr} has three different sources:

1. the visual input $v^{pr}(x^{pr}, t)$ mediated by the gain constant $C^{v \to pr}$;
2. the propagation of the inhibitory interaction from u^{pa} to u^{pr} where only activity above the threshold level 0 is propagated to u^{pr};
3. the "recall" contribution from $\underset{pa \to pr}{A}$, with a strength $C^{rec \to pr}$.

The equation below describes the input $s^{pr}(x^{pr}, t)$:

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

where $\underset{pa \to pr}{A}(x^{pa}, x^{pr}, t)$ represents the synaptic adaptation rule storing the associative learning connections from u^{pa} to u^{pr}, detailed next.

Adaptation rule: $\underset{pa \to pr}{A}$

$\underset{pa \to pr}{A}$ stores the connections from u^{pa} to u^{pr} established during the Learning mode through the set of demonstrations provided by the human tutor. The weights are adapted when sub-neuronal populations present activity above the learning thresholds - $u^{pa} > \lambda^{pa}$ and $u^{pr} > \lambda^{pr}$ -, according to the Delta Rule adaptation for the continuous space (equation (5.19)), now formalized as:

$$\begin{aligned} \tau_a \frac{\partial \underset{pa \to pr}{A}(x^{pa}, x^{pr}, t)}{\partial t} &= f_{\lambda^{pr}}\Big[u^{pr}(x^{pr}, t)\Big] f_{\lambda^{pa}}\Big[u^{pa}(x^{pa}, t)\Big] \\ &\times \Big[e^{pr}(x^{pr}, t) - \underset{pa \to pr}{A}(x^{pa}, x^{pr}, t)\Big], \end{aligned} \tag{7.8}$$

with the error term being formulized similar to equation (5.21):

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

The error term $e(x^{pr},t)$ is the difference between activity above the threshold level λ^{pr} given by $g_{\lambda^{pr}}$ (see equation (5.20)) and the total amount of input given through the connections $\underset{pa\to pr}{A}$. During the Recall mode, if the robot makes a wrong prediction, the learning threshold λ^{pa} is placed lower (Figure 7.7). That way, more past sub-populations will be above λ^{pa} in the next trial (see Sousa, 2014).

7.1.5 Short-Term Memory Layer (STM)

Although the models presented in section 7.1 can achieve the goal of learning a sequential task from demonstration, the adaptation of the weights calls for a large number of demonstration trials in order to extract the correct weight values stored in $\underset{pa\to pr}{A}$. When working with neural networks where the training is performed off-line, this is not an issue. However, in this particular Programming by Demonstration scenario, it is the human tutor the one performing the training trials in an online process, so having multiple demonstration trials may become annoying, unpractical, and time-consuming (Cunha et al., 2020).

To overcome this limitation, Ferreira et al. (2014) proposed a neural-inspired fast learning system capable of storing an observed sequence in a single demonstration. Later, Sousa et al. (2015) combined the Sequence Learning Layer model with this Short-Term Memory (STM) mechanism to work as two-complementary processes, inspired by the work of Mcclelland et al. (1995). The STM layer is formed by a single Dynamic Neural Field that stores the sequence of sub-goals demonstrated as a multi-bump pattern. This pattern is later used as input to the SLL to train the adaptable synaptic connections during the Rehearsal mode (Figure 7.9).

Each bump represents an event triggered through excitatory input from the Vision System. The strength of each memory representation reflects the time elapsed since each event was observed (e.g., blue thruster inserted), resulting in an activation gradient from the first to the last sub-goal observed (Figure 7.10). This way, the STM model can store in one-shot a sequence of sub-goals performed by the tutor, which is later used as input to the Present field u^{pr}.

Since the STM model can store a sequence of steps in a single demonstration, this model can be used to replace the human tutor during the Rehearsal mode. The sequence encoded in the STM layer is used as input

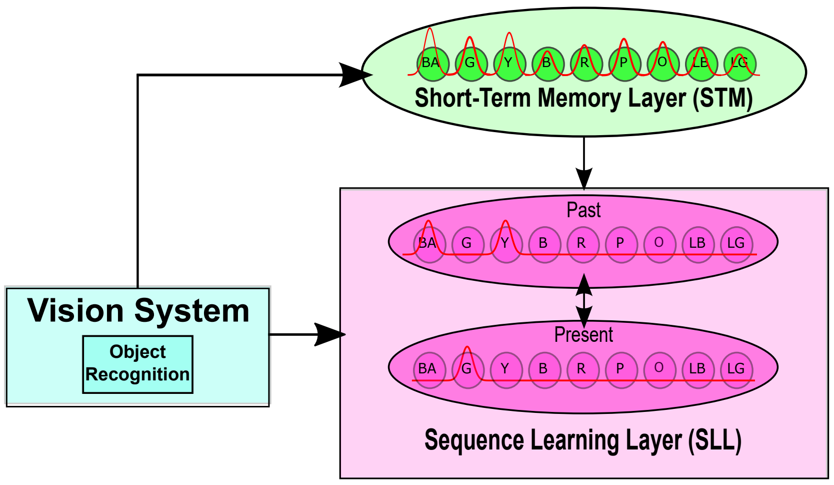


Figure 7.9: SLL architecture extended to incorporate the Short-Term Memory Layer (STM). The STM Layer is formalized as a single Dynamic Neural Field. During the Rehearsal mode, the input from the Vision System to SLL is replaced by the input from the STM layer.

to the Sequence Learning Layer, replacing the input that was previously being provided by the Vision System. The Rehearsal process consists of feeding the Present Field u^{pr} in the SLL, several times in a loop, with the multi-bump pattern (Figure 7.10) stored in the STM, providing the training trials necessary to adapt the synaptic weights. The process is repeated until a pre-defined number of rehearsal trials is reached. The implementation parameters of this mode can be seen in Appendix A.3.

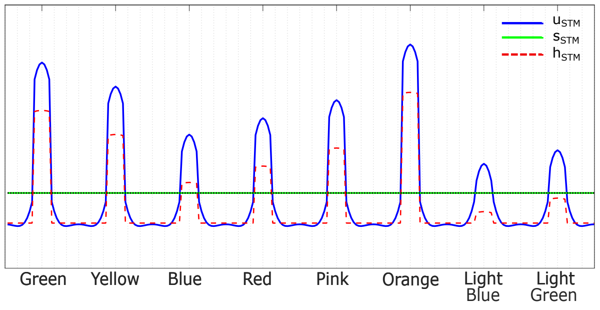


Figure 7.10: Example of a multi-bump pattern stored in STM. The amplitude of the bumps encodes the serial order of the inserted thrusters, with the highest peak (Orange) being the color of the first inserted piece and the shortest one (Light Blue) being the last.

7.1.6 Mathematical formulation of the Short-Term Memory Layer and the Rehearsal mechanism

STM Field

The equation below describes the activity of the STM field u^{stm}:

$$\begin{aligned}\tau^{stm}\frac{\partial u^{stm}(x^{stm},t)}{\partial t} &= -u^{stm}(x^{stm},t) + h^{stm}(x^{stm},t) + \varsigma^{stm}_{stoch}(x^{stm},t) \\ +f_{\beta}[u^{stm}(x^{stm},t)]\bigg(&s^{stm}(x^{stm},t) + \int w_{stm}(x^{stm}-y^{stm})f_0[u^{stm}(y^{stm},t)]dy^{stm}\bigg).\end{aligned} \tag{7.10}$$

The interaction kernel function w_{stm} used is an Oscillatory kernel (equation (4.7)). The implementation parameters for this field can be consulted in the Appendix A.2.

Input to the STM Field: s^{stm}

The STM field $u^{stm}(x^{stm},t)$ receives input from the Vision System $v^{stm}(x^{stm},t)$ with a strength weighted by $C^{v\rightarrow stm}$. The input $s^{stm}(x^{stm},t)$ is formulized as:

$$s^{stm}(x^{stm},t) = C^{v\rightarrow stm}v^{stm}(x^{stm},t). \tag{7.11}$$

Resting level: h^{stm}

The adaptation of the resting level of the STM field is the mechanism that allows a sequence of sub-tasks to be represented in the activation gradient of the field (Ferreira et al., 2014). The next equation governs the adaptation of h^{stm}:

$$\begin{aligned}\tau^{stm}_{h}\frac{\partial h^{stm}(x^{stm},t)}{\partial t} &= \Big(1 - f_0[u^{stm}(x^{stm},t)]\Big)\Big(H^{stm}_{dec} - h^{stm}(x^{stm},t)\Big) \\ &\quad + f_0[u^{stm}(x^{stm},t)],\end{aligned} \tag{7.12}$$

where τ^{stm}_{h} defines the growth rate of $h^{stm}(x^{stm},t)$ and H^{stm}_{dec} is the baseline value. When $u^{stmn} < 0$, the resting level decays to the baseline value. Otherwise, $h^{stm}(x^{stm},t)$ follows a linear grow (see red dashed line in Figure 7.10), which allows the generation of the amplitude gradient encoding serial order.

Input to the Present Field: s^{pr}

So far, the input to the present field has been defined as (7.7). For simplicity, let's write the previous input as $s^{SLL}(x^{pr}, t)$. With the implementation of the Short-Term Memory mechanism, the input to the Present field can be now written as

$$s^{pr}(x^{pr}, t) = s^{SLL}(x^{pr}, t) + b^{reh} C^{stm \to pr} u^{stm}(s^{stm}, t), \tag{7.13}$$

with the second term of the equation being the contribution from the STM Layer, mediated by the gain $C^{stm \to pr}$. The parameter b^{reh} works as a flag that is only active during the Rehearsal mode, so the STM Layer only contributes to the Present field input s^{stm} during Rehearsal.

7.2 Results: What piece should be inserted next?

This section presents the tests performed to validate the models described in section 7.1 as well as the results obtained, using the joint-task scenario presented in section 6.2.1. The goal was to test and validate the DNF-models in a real-world case where a robot learns the task demonstrated by a human tutor.

7.2.1 Experimental Setup

The experimental paradigm used in this work consists of a scenario where Human and Robot are placed facing each other on opposite sides of a table. The interaction starts with the demonstration performed by the human tutor. In the beginning, all the thrusters are laying on the workspace table and the tutor verbalizes the sentence "I will show you" to initiate the trial (Figure 7.11 - left), which sets the system's state to the Demonstration mode. While the human performs the demonstration, the robot observes and the Vision System captures the insertion of each piece recognized by its color (Figure 7.12). Each thruster has a fixed location in the structure and can be placed at any moment, as long as the state of the constructions allows its insertion. The visual information is translated into Gaussian inputs v^{pa}, v^{pr} and v^{stm} that will trigger the representation of the achievement of each sub-goal (e.g, Pink thruster inserted) in the respective fields, u^{pa}, u^{pr} and u^{stm}.

The verbalization of the sentence "I have finished" (Figure 7.11 - right) by the tutor ends the demonstration

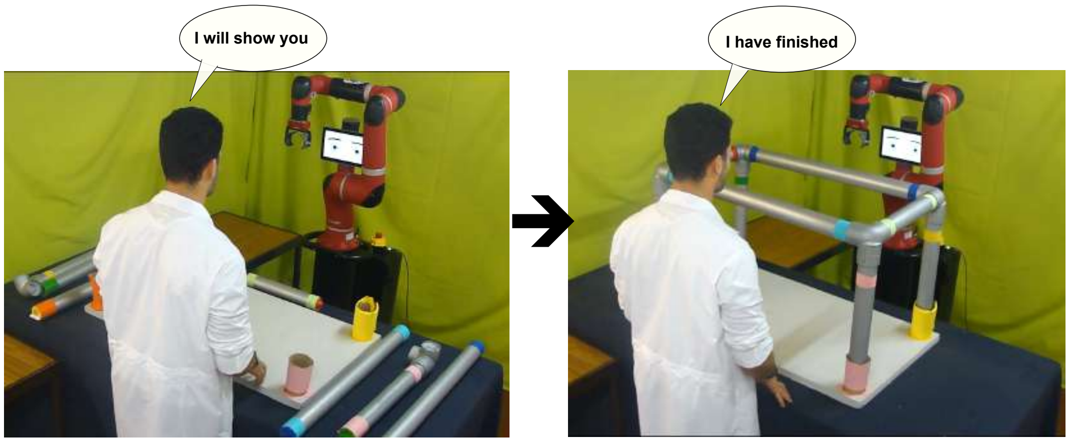
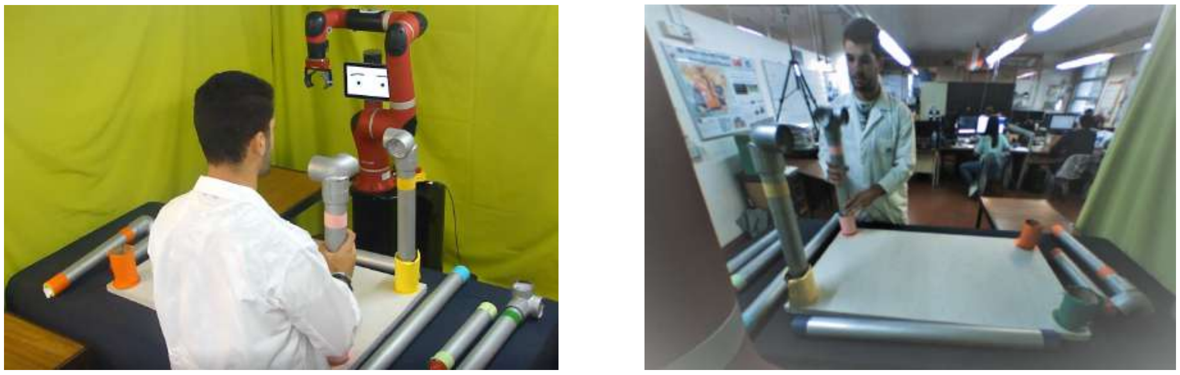


Figure 7.11: Beginning and ending of one demonstration trial. To begin the trial, the tutor says "I will show you". After demonstrating the sequence of sub-goals, the human verbalizes "I have finished" to indicate that the task is completed.

trial and switches the system to the Rehearsal mode. Subsequently, the resulting multi-bump pattern stored in the STM layer is used to train the synaptic weights of the learning matrix $\underset{pa\rightarrow pr}{A}$. Before initiating the Rehearsal, Sawyer verbalizes the sentence "Let me think about it", and at the end of the process it verbalizes "I am ready" to call the tutor's attention that the robot is ready to proceed to the Recall mode.



(a) Outside camera view

(b) Head Robot camera view

Figure 7.12: Insertion of the Pink thruster. Image (a) is a snapshot of the outside camera recording the experiment where it can be seen the tutor inserting the Pink thruster. Snapshot (b) corresponds to Sawyers' head camera view that can be used to provide the visual input.

In the Recall mode, Sawyer starts retrieving each sub-goal of the task and verbalizes the result of its decision process consisting on which piece should be inserted next, considering the current state of the construction. The

Base is already in place at the beginning of each trial to provide the first input and trigger the chaining mechanism. During the process, human feedback is provided at each step, which can be either positive - “That is correct” - or negative - “That is wrong”. In case Sawyer makes a wrong prediction, negative feedback is provided (Figure 7.13), the learning threshold value λ^{pa} is lowered and a new demonstration is requested.

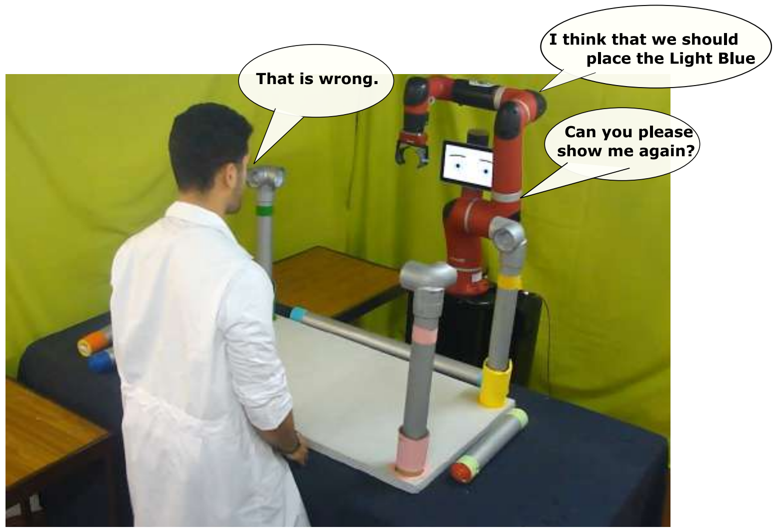

That is wrong.
I think that we should place the Light Blue
Can you please show me again?

Figure 7.13: Error in the Recall process. If the robot makes a wrong prediction, the tutor provides negative feedback which will trigger the routine that places the threshold value λ^{pa} with a lower value to allow more associations between sub-goals in the next demonstration trial. Facing the negative feedback, Sawyer requests a new demonstration by verbalizing the sentence “Can you please show me again?”.

The Demonstration $\rightarrow$ Rehearsal $\rightarrow$ Recall process is repeated until, after observing different sequences, the robot is able to retrieve the task with no errors. In the end, it is expected that the robot acquires the precedent relations between the several sub-goals, extrapolating task knowledge. Next are described the set of tests performed to validate the models.

7.2.2 Test number 1: Learning a sequence

In the first test, the human tutor demonstrated the task by inserting each piece using the following sequence :

Orange $\Rightarrow$ Yellow $\Rightarrow$ Green $\Rightarrow$ Red $\Rightarrow$ Blue $\Rightarrow$ Pink $\Rightarrow$ Light Green $\Rightarrow$ Light Blue

The demonstration trial resulted in the multi-bump pattern depicted in Figure 7.14, stored in the STM Layer. A video of this trial can be seen at https://youtu.be/DgLnFuOkwvs. The amplitude of the peaks encodes the serial order demonstrated by the tutor. The sub-population corresponding to Base will always have the highest amplitude since the Base is already in place at the beginning of each demonstration. Apart from that, the next highest peak (Orange) corresponds to the first inserted piece and the lowest bump (Light Blue) to the last inserted thruster.

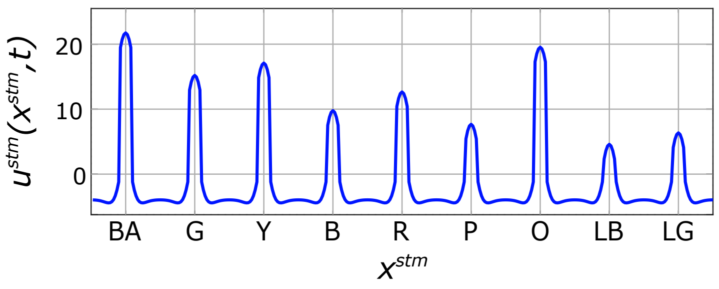
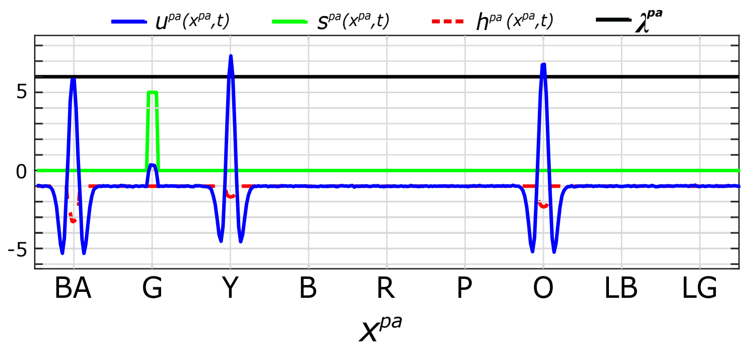


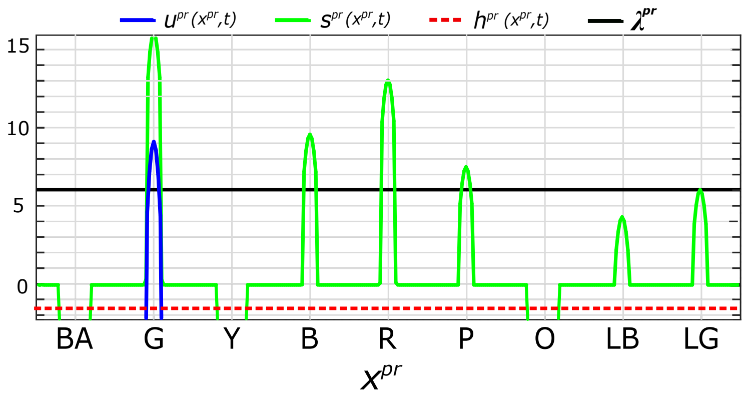
Figure 7.14: Activation of the STM field u^{stm} resulting from first demonstration.

Next follows the Rehearsal process. In total, 40 rehearsal trials were made in a loop, in which the STM pattern provides input to the fields in the Sequence Learning Layer to train the synaptic connections of $\underset{pa\to pr}{A}$. During this test, the Past learning threshold λ^{pa} was equal to 6. Each one of the rehearsal trials had a total of 400 epochs. This value was set heuristically to allow all the sub-neural populations in u^{stm} to trigger activation above the threshold level in the Present field λ^{pr}. After each trial, the activation of the SLL fields u^{pa} and u^{pr} are re-set to the respective resting level values. During rehearsal, the tutor dismounted the structure, which had no implications in the system since the input from the STM field replaced the visual input.

Figure 7.15 is a snapshot of the SLL fields at one instant of the Rehearsal trial. At that moment, the Base and the Yellow and Pink thrusters are sub-goals already marked as completed in the Past field (blue line in Figure 7.15a) and therefore inhibited in the Present field (Figure 7.15b). When the input for the Green thruster is given in u^{pr}, it reaches λ^{pr} (dark line), and the sub-populations in u^{pa} above the learning threshold λ^{pa} - Yellow and Orange - become linked to the sub-population encoding the Green. Immediately, the active bump in u^{pr} will trigger the corresponding population in u^{pa}. This is the mechanism behind the adaptation of the synaptic weights of $\underset{pa\to pr}{A}$. When the number of total rehearsal loops is reached, $\underset{pa\to pr}{A}$ will encode in their weights the connections between the sequential observed sub-goals, according to the time window define by the the value of λ^{pa}.



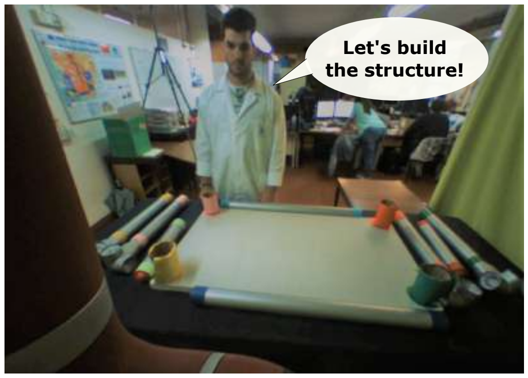
(a) Past Field



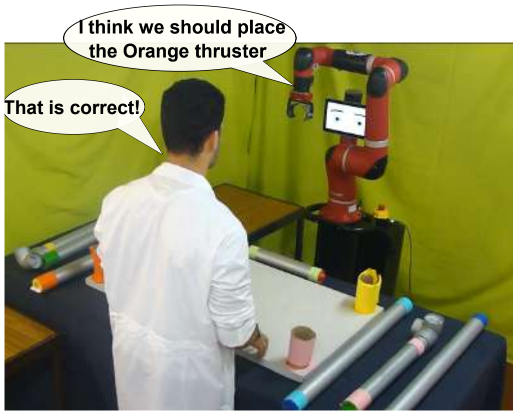
(b) Present Field

Figure 7.15: Snapshot of the associations between sub-goals during Rehearsal. Completed sub-goals are marked in the Past field (a). When a new bump emerges in the Present field (b), sub-populations above λ^{pa} become linked to the active population in u^{pr} while the same is above λ^{pr} - associative learning.

With the Rehearsal process completed, the experiment continued to the Recall mode. The human tutor verbalized to the robot "Let's build the structure" to initiate the Recall procedure (Figure 7.16a). The Base was already placed in the table which leads to the appearance of activity in the corresponding sub-population in u^{pa} (Figure 7.17a). Next, the propagation of the adaptive connections $\underset{pa\rightarrow pr}{A}$ from u^{pa} to u^{pr} generated activity in s^{pr} where two prediction peaks - corresponding to the Orange and Yellow - were developed (Figure 7.17b). The appearance of two prediction bumps results from the associations made during rehearsal, where in each step, in general, two sub-goals became linked to the next. Since the amplitude of the bump encoding the Orange was higher than the one encoding the Yellow, the prediction verbalized by the robot laid towards the insertion of the Orange thruster (Figure 7.16b).

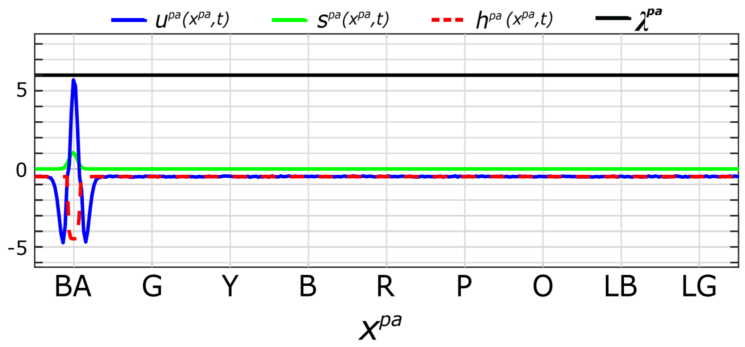


(a) Beginning of the Recall mode: Visual input

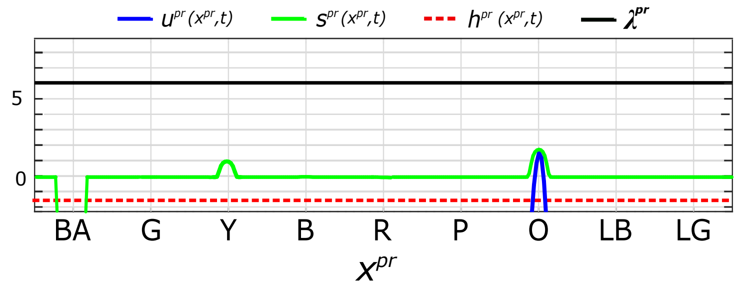


(b) First prediction made by the Robot

Figure 7.16: Illustration of the Recall Mode. The tutor gives the command to initiate the task recall (a). The visual input triggered the first prediction - insertion of the Orange thruster (b).



(a) Past Field



(b) Present Field

Figure 7.17: SLL profile during the first prediction. The model recalls the Orange thruster. A smaller bump appears in the sub-population encoding the Yellow piece due to the associations made during the rehearsal who also linked BA to Y.

The process continued until all the pieces were inserted. The robot recognizes a period without activity in u^{pr} as the end of the task since the model predicts no more sub-goals. Then Sawyer verbalizes "I think we have finished", marking the end of the recall process without any error.

7.2.3 Test number 2: Learning an alternative sequence

In the second experiment, the tutor demonstrated the task using a different assembling sequence:

Pink $\Rightarrow$ Green $\Rightarrow$ Yellow $\Rightarrow$ Blue $\Rightarrow$ Orange $\Rightarrow$ Light Blue $\Rightarrow$ Light Green $\Rightarrow$ Red

A video of this demonstration can be seen at https://youtu.be/5CwwHiO1NcA.Figure 7.18 shows the multi-bump pattern stored in the STM layer that encodes the serial order of sub-goals of this trial.

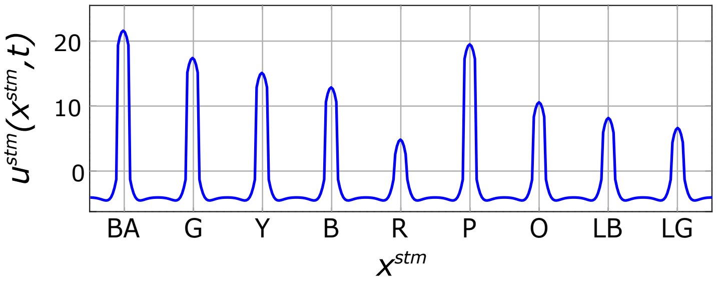
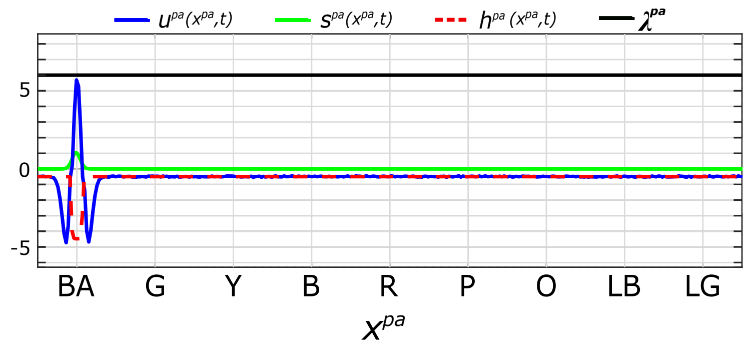


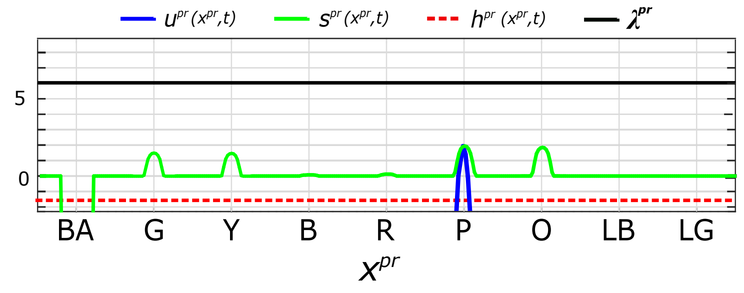
Figure 7.18: Activation of the STM field u^{stm} resulting from the second demonstration.

Following the same learning process, the Rehearsal took place after the Demonstration. Since the robot did not make a mistake during the recall of the first learned sequence, the learning threshold level was kept in the same value ($\lambda^{pa} = 6$). During the rehearsal of the new observed sequence, the synaptic weights of $\underset{pa \to pr}{A}$ are readjusted to accommodate the new connections established between the several observed sub-goals in the new demonstration.

Figure 7.19 depicts the activation profiles of u^{pa} and u^{pr} in the beginning of the recall process. The main difference between this test and the previous one is that the Base triggered four predictions in s^{pr}, corresponding to the insertion of the Green, Yellow, Pink and Orange thrusters (Figure 7.19b). The associations (P and $O \to BA$) were already established during the first trial (see Figure 7.17b). Then, in the second demonstration, the Base was followed by the Pink and the Green thrusters, forming the associations (P and $G \to BA$). It is worth to note that each one of the pieces could be the next inserted in the construction. In this case, the decision of which sub-goal becomes active depends on the stochastic noise ς^{pr}_{stoch} introduced in equation (7.6). The sub-goal recalled was the Pink thruster.



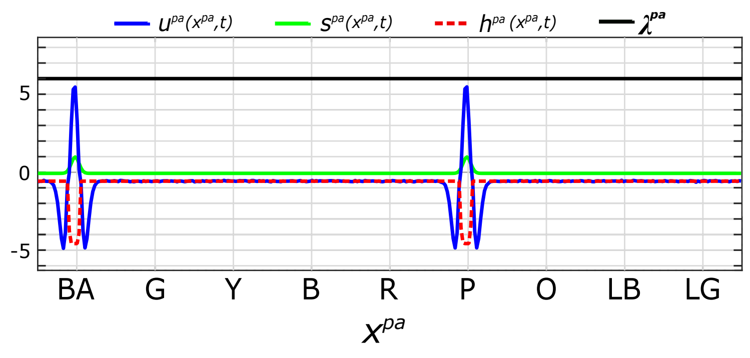
(a) Past Field



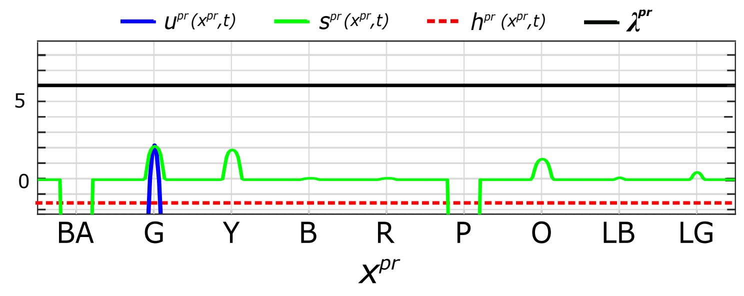
(b) Present Field

Figure 7.19: SLL profile during the first prediction. Four prediction bumps are triggered in u^{pr} corresponding the sub-populations encoding the Green, Yellow, Pink and Orange pieces. The model recalls the Pink thruster.

Next, the tutor provided positive feedback and continued by placing the Pink thruster. Figure 7.20 shows the activation of the SLL fields during the second step of recall. In Figure 7.20a it is visible that the Base and the Pink thruster are marked as achieved sub-goals in u^{pa}. The insertion of the Pink thruster triggered activity in the populations of the sub-goals observed after, both during the previous ($P \rightarrow LG$) and the current trial ($P \rightarrow G$). Since the Green had already received previous input from the Base, the total input formed the highest peak in the sub-population encoding the Green, which won the competition over the other active predictions (Figure 7.20b). Moreover, the second observed sub-goals after the Pink ($P \rightarrow LG \rightarrow$ **LB**) and ($P \rightarrow G \rightarrow$ **Y**) triggered a smaller bump in the Light Blue sub-population and increased the prediction input of Yellow, respectively. This happened since the learning threshold value in u^{pa} allows, in general, a compound association of two sub-goals (see Figure 7.15b).



(a) Past Field



(b) Present Field

Figure 7.20: SLL profile during the second prediction. The model recalls the Green thruster.

The complete sequence retrieved by the robot during the rehearsal process was the following:

Pink $\Rightarrow$ **Green** $\Rightarrow$ **Yellow** $\Rightarrow$ **Orange** $\Rightarrow$ **Red** $\Rightarrow$ **Blue** $\Rightarrow$ **Light Blue** $\Rightarrow$ **Light Green**

All the recalled sub-goals were correct, so Sawyer received positive feedback through all the steps.

7.2.4 Test number 3: Learning a third sequence

In the third test, the tutor demonstrated the task by inserting each piece using the following sequence :

Yellow $\Rightarrow$ Pink $\Rightarrow$ Light Green $\Rightarrow$ Orange $\Rightarrow$ Green $\Rightarrow$ Red $\Rightarrow$ Light Blue $\Rightarrow$ Blue

The demonstration resulted in the multi-bump pattern displayed in Figure 7.21 A video of the third demonstration can be found at https://youtu.be/RO2MHIKubvU. As the previous experiments, the demonstration is followed by the rehearsal process where the weights of the adaptable connections $\underset{pa\to pr}{A}$ from the Past sub-goals to the Present are readjusted.

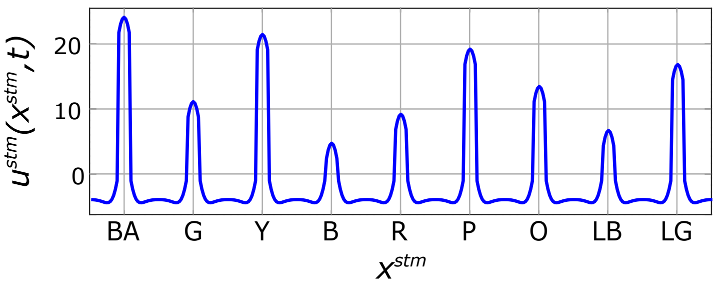
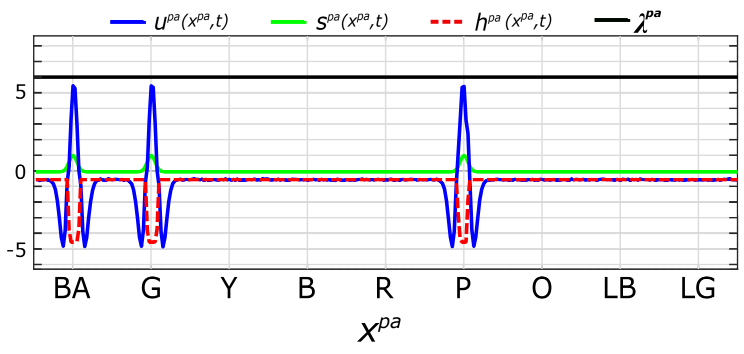
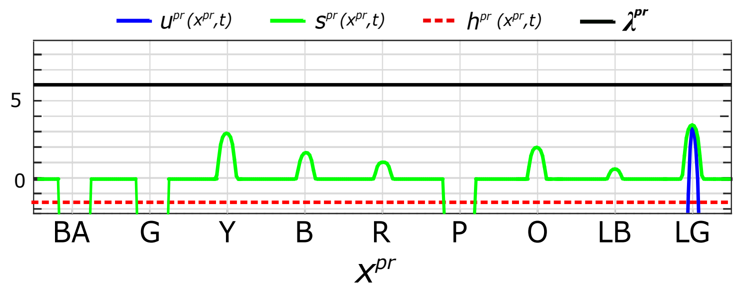


Figure 7.21: Activation of the STM field u^{stm} resulting from the third demonstration

During the recall mode, the robot starts by selecting the Pink thruster to be inserted after the Base, followed by the Green. Figure 7.22 shows the activation profile of the SLL fields after the tutor inserted the Green thruster.



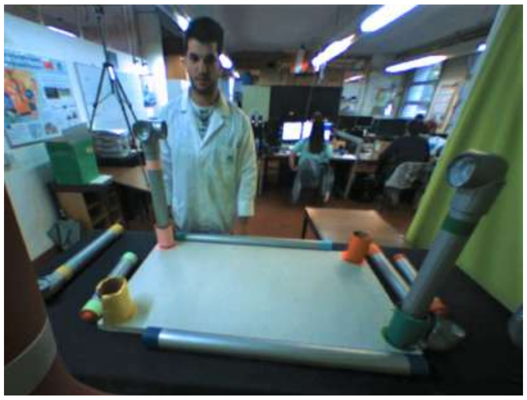
(a) Past Field



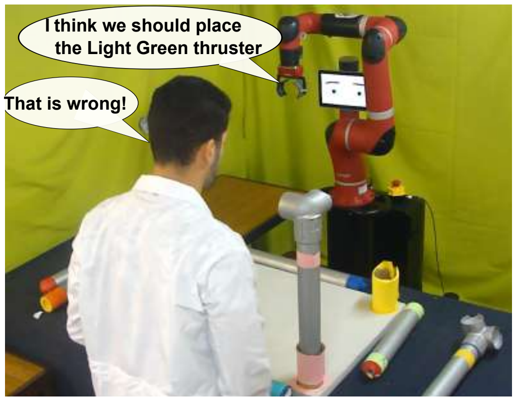
(b) Present Field

Figure 7.22: Error in the recall process. After inserting the Pink and Green thruster, the robot makes a wrong decision, recalling the Light Green.

At this moment, it is visible that all the remaining sub-goals in s^{pr} (Figure 7.22b) have a peak (bigger or smaller), due to the propagation of the connections in $\underset{pa \to pr}{A}$ that resulted from learning all the previous sequences. The next sub-goal recalled by the model was the Light Green (Figure 7.23). Taking into consideration the physical constraints of the structure, the Light Green thruster cannot be inserted at this stage of the construction. The decision made by the robot is wrong, so Sawyer receives negative feedback from the tutor (Figure 7.23b), which will lower the learning threshold to the next value: $\lambda^{pa} = 5.5$. Because it received negative feedback, the robot asks the tutor to repeat the demonstration (see Figure 7.13). A video of the complete trial can be found at https://youtu.be/CLAZJOg1kys. After the tutor demonstrates the sequence of steps once again, the rehearsal



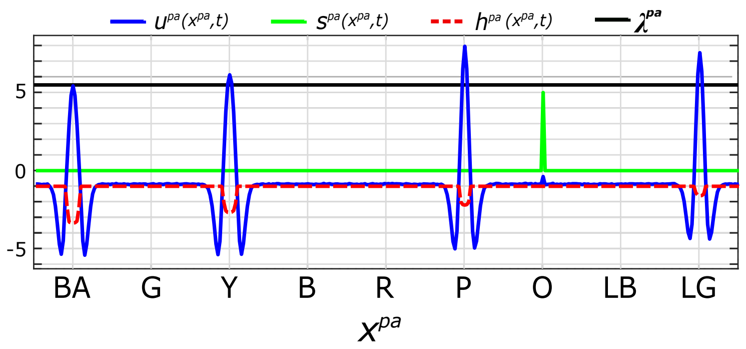
(a) Robot Head Camera view: Visual input



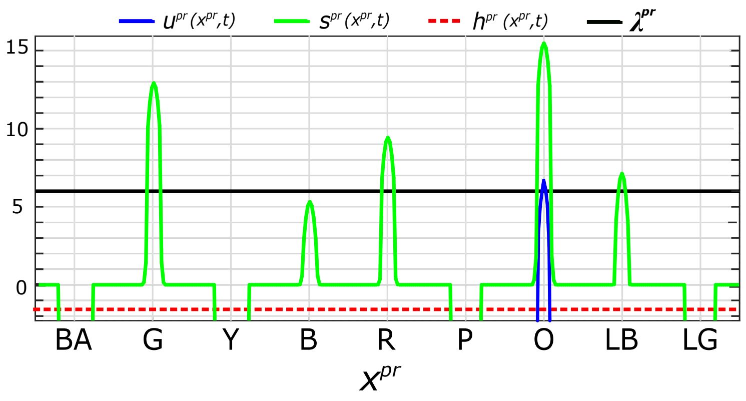
(b) Robot makes a wrong prediction

Figure 7.23: Illustration of the Recall Mode - Sawyer makes a wrong prediction. After recalling the Pink and Green pieces (a), the robot recalls the Light Green thruster to be placed next (b). The physical constraints of the structure do not allow the insertion of Light Green at this step of the construction.

process takes place, this time with a lower threshold value. The lower threshold allows the association of more achieved sub-goals in u^{pa} to become linked to the next sub-goal observed in u^{pr} (Figure 7.24). Also, the relations observed during the demonstration are reinforced in $\underset{pa \to pr}{A}$ during the new rehearsal trial.



(a) Past Field



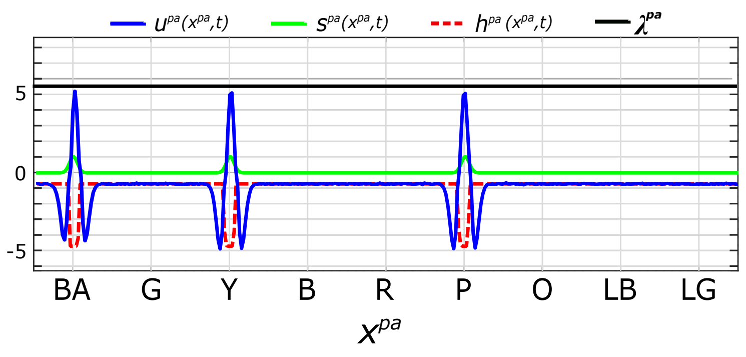
(b) Present Field

Figure 7.24: Snapshot of the associations between sub-goals during Rehearsal with $\lambda^{pa} = 5.5$. During this rehearsal trial, the learning threshold is placed lower (a). When a new bump emerges in the Present field (b), sub-populations above λ^{pa} (Pink, Light Green and Yellow) become linked to the active population in u^{pr} (Orange).

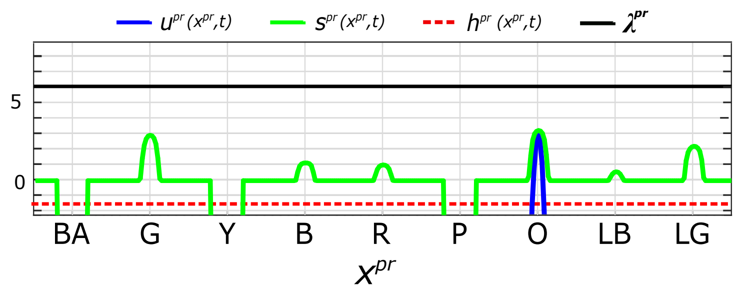
Going back to the Recall mode, the robot starts by recalling the Yellow piece, followed by the Pink. Figure 7.25 shows the activation profile of the SLL fields after achieving these two sub-goals. Contrarily to the previous trial, this time, the robot retrieves the Orange piece to be inserted next, making a valid decision (Figure 7.25b). Then, the human tutor provides positive feedback and the recall of the next sub-goals is due with no more errors. The final sequence retrieved by the robot was the following:

Yellow $\Rightarrow$ Pink $\Rightarrow$ Orange $\Rightarrow$ Light Green $\Rightarrow$ Green $\Rightarrow$ Red $\Rightarrow$ Blue $\Rightarrow$ Light Blue

A video of the trial can be seen at https://youtu.be/nG2s79gdquQ.



(a) Past Field



(b) Present Field

Figure 7.25: SLL profile during the third prediction after repeating the demonstration. With the Yellow and Pink thrusters inserted (a), the robot makes a valid decision and retrieves the Orange piece (b).

7.3 Discussion

The model for sequence learning presented in this chapter and the experimental results show that it was possible to capacitate Sawyer with the ability to acquire information regarding the sequential structure of a task, using learning from demonstration and tutor feedback, in a real-time Human-Robot Interaction scenario. First of all, it was proved that the Dynamic Neural Field framework could be used to represent information regarding the state of the execution of a task, through the interactions between two coupled DNF-fields. Moreover, it was possible to store serial order information in a single dimension field where the order is represented in the amplitude of the sub-neuronal populations encoding specific sub-goals. The former could be translated into a Short-Term Memory mechanism, as also demonstrated in the work of Ferreira et al. (2014). The integration of the STM model to train the associative learning rule permitted an online and fast method for training the connections encoded in the learning matrix. The visual input provided by the demonstrations made by the human tutor was replaced by the STM pattern during the Rehearsal mode, reducing the workload for the human co-worker. Furthermore, the compound chaining mechanism applied allows storing long-term precedencies between sub-goals, encoded in $\underset{pa \to pr}{A}$, which are needed when dealing with multi-sequential tasks. It was also verified that the variation of the learning threshold value could be used to control the number of sub-goals that are associated at each step of the task. In the work of Sousa et al. (2015), a similar approach was employed; however, the time window for learning was controlled by changing the resting level decay rate in the Past field. Also, the authors started by testing the learning of a sequence through simple chaining where the adaptable connections from Past to Present field only associated one sub-goal at each step. Following their work, in this experiment, a compound chaining approach was used, from the beginning, through all the trials, so the time window for learning always allowed, at least, two sub-goals in the Past field to become linked to the next one in the Present field. This way, more connections between sub-goals were established. It is also worth noting that, after the second demonstration, the robot was able to recall the task using a sequence that was not explicitly demonstrated by the tutor. This means that the model was able to extrapolate dependences between sub-goals that were not demonstrated explicitly. For example, in the second recall trial, after the Yellow piece, the robot recalled the Orange and then the Red, sequences that were not demonstrated by the tutor. This approach allowed the robot to extrapolate task information faster, with more flexibility and through a smaller number of demonstrations.

The next chapters will describe two other experiments performed using the developed models in different Human-Robot Collaboration scenarios.

Chapter 8

Extending the models to incorporate workspace information and action selection

In the previous Chapter, it was shown how it is possible to endow Sawyer with the ability to acquire task-related information, from tutor demonstration and verbal feedback. This Chapter presents a different HRC scenario where the learned information is applied in an architecture that incorporates workspace information and decision making regarding the selection of an appropriate action. The main goal was to incorporate the acquired knowledge in a construction scenario where Sawyer has to make a decision regarding, not only the next sub-goal but also how that sub-goal should be achieved. The key questions - described in section 6.2.2 - are 1) what action should be done over the selected part; and 2) who should do it. Next are described the architecture and the experiments made to validate the new incorporated models.

8.1 DNF - architecture incorporating workspace information and action selection

The models implemented in Chapter 7 endowed Sawyer with high-level knowledge about the sequential structure of the joint construction task. In the experiment presented in this Chapter, a new HRC scenario was set up, where the robot takes an active role and engages in the construction of the task together with the human partner. The main difference from the previous experiment is that the robot no longer receives feedback from the human tutor, since the task had been already learned, being stored in the Sequence Layer (SL) of the new architecture. Now, the robot not only verbalizes the next step (as before) but also, and important for the joint cooperation, it engages actively with his co-worker in the construction of the task.

Figure 8.1 is a diagram of the cognitive architecture implemented where three new layers are added, bringing new capabilities to the models. The architecture is inspired in the work of Bicho et al. (2011b) and Erlhagen and Bicho (2014), and combines information from several layers into a single bidimensional field that holds the decision making process for action selection. The several pieces that form the construction task are dispersed into three different workspaces - Human, Robot and Shared - encoded in the neural fields inside the Object Memory Layer (OML). Input from the Vision System triggers activity in each field, marking the location of each piece across the different workspaces. Moreover, information regarding what pieces the robot is (physically) able to insert is codified in the Executable Tasks Layer (ETL). Action Execution Layer (AEL) is a two-dimensional field that holds the decision-making process of the architecture and triggers the corresponding selected action to be executed, according to the input received from the OML, ETL, and SL. Task-information - i.e. sequence of sub-goals in the construction task - is stored in the SL and results from the previous learning models. For example, according to the Present field in the figure, the next sub-goal should be the insertion of the Green thruster. The Green piece is on the Robot Workspace and, according to the information provided by the ETL, the robot can insert it. Therefore, the decision that will emerge in the AEL is: Robot Inserts (RI) (the Green thruster).

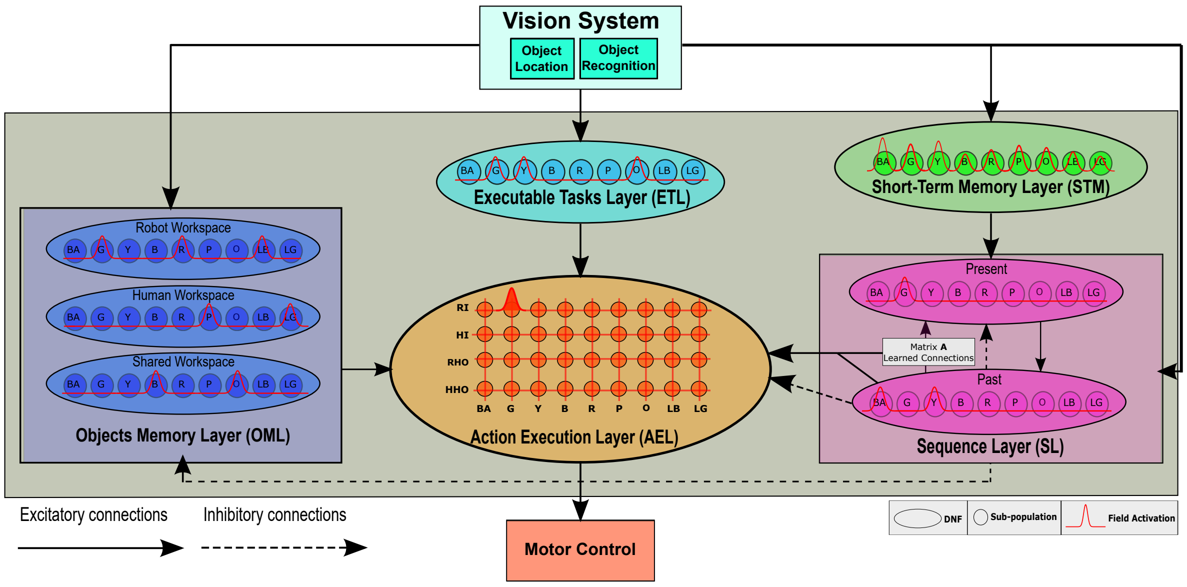


Figure 8.1: Schematic view of the cognitive architecture for joint action. The OML incorporates information regarding the state of each workspace, receiving input from the Vision System. ETL encodes the pieces that Sawyer can insert and AEL holds the decision generated from the input received from OML, ETL and SL. The decision process is constituted by two parts: 1) the color of the next piece and 2) how to perform the insertion: Robot Inserts (RI), Human Inserts (HI), Robot Hands Over (RHO) or Human Hands Over (HHO). The combination of both outcomes results in a bump of activity emerging in the two-dimensional field expressing the complete decision-output, which will trigger the corresponding motor control action.

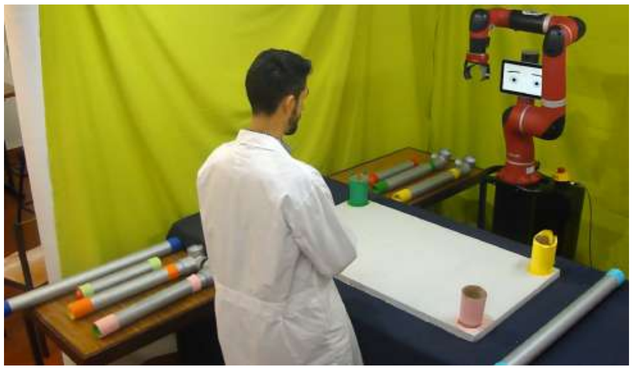
Summarizing, four different actions can be taken to insert each piece, which will depend on the location of the piece and on the ability of the robot to insert (or not) each thruster in the structure:

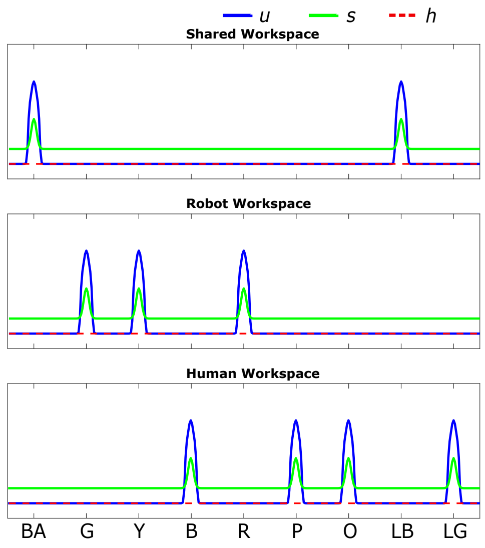
- **Robot Inserts (RI):** The thruster is placed in the Robot or Shared Workspaces and the robot can reach and insert it in the structure - robot grabs and inserts the piece;
- **Human Inserts (HI):** The thruster is placed in the Human or Shared workspaces and the robot is not able to insert it in the structure, so the action is fully performed by the human - human grabs and inserts the piece;
- **Robot Hands Over (RHO):** The thruster is placed in the Robot Workspace but the robot cannot reach its location in the structure - Robot grabs the piece and hands it over to the human co-worker that performs the insertion;

- **Human Hands Over (HHO):** The thruster is placed in the Human Workspace, but the robot can insert it in the structure - Human grabs the piece and hands it over to the robotic co-worker that performs the insertion, reducing the workload of the human partner.

8.1.1 Object Memory Layer

The Object Memory Layer encodes information regarding the three different workspaces. Each workspace is modeled as a single Dynamic Neural Field - fields colored in Blue in Figure 8.1 -, and represents the pieces that are placed in each one of them. Figure 8.2 shows one of the possible scenarios illustrating the three workspaces and the respective fields in the OML. The Human Workspace is formed by the pieces disposed on the table next


(a) Disposal of the pieces across the different workspaces

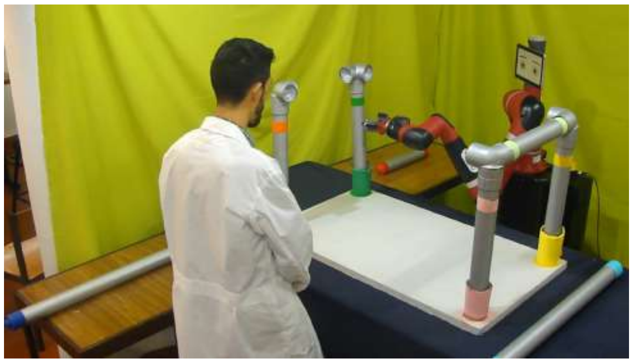


(b) Activation profile of the OML fields

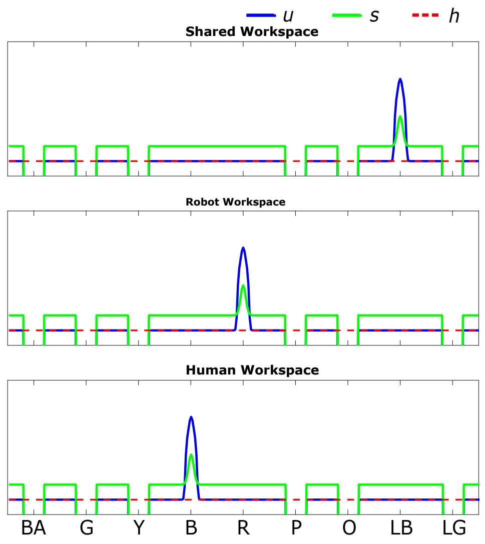
Figure 8.2: Human, Robot and Shared Workspaces - layout of the pieces and corresponding OML fields at the beginning of the construction. Orange, Pink, Light Green and Blue belong to the Human Workspace in (a) and are marked as bumps of activity in the corresponding field (b). Green, Red and Yellow are placed in the Robot Workspace. Lastly, the Shared Workspace has the Base and the Light Blue thruster.

to the human (Figure 8.2a). The pieces belonging to the Robot Workspace are placed on the other table next to the robot. Finally, the Shared Workspace is composed of the Base and the thrusters lying on both sides, on top of the working table (blue table). Figure 8.2b shows the activation of each one of the fields according to the disposal of the pieces in the scenario illustrated in Figure 8.2a. Input from the Vision System (Object Location) generates bumps of activity in the sub-populations encoding the colored thrusters placed in each workspace. The Base is always placed in the working table, belonging to the Shared Workspace.

The Object Memory Layer receives inhibitory input from the Past field in the SLL. When a sub-goal is marked as achieved, the inhibitory connection $\underset{pa\rightarrow oml}{I}(x^{pa}, x^{oml})$ - which is equal to $\underset{pa\rightarrow pr}{I}(x^{pa}, x^{pr})$ -, suppresses the sub-populations in the OML fields corresponding to the pieces already inserted. Figure 8.3 exemplifies the profile of each workspace and their corresponding fields at a middle step of the construction. The Base is the first sub-goal inhibited in the OML since it is already placed in the table when the construction starts. Then, with the insertion of the Pink, Yellow, Green, Orange and Light Green thrusters, all of the corresponding populations of neurons are inhibited (Figure 8.3b). Consequently, there are only peaks of activity in the Red, Blue and Light Blue thrusters that are placed in the Human, Robot and Shared workspaces, respectively.



(a) Disposal of the pieces across the different workspaces



(b) Activation profile of the OML fields

Figure 8.3: Human, Robot and Shared Workspaces - layout of the pieces and corresponding OML fields at a middle step of the construction. Sub-populations encoding the Yellow, Pink, Orange and Green are inhibited. Blue belongs to the Human Workspace, Red to the Robot Workspace and Light Blue is placed in the Shared Workspace.

8.1.2 Executable Tasks Layer

The Human-Robot Collaboration scenario used in the next experiments requires Sawyer to collaborate with his co-worker in the construction of the task. Considering the physical limitations of the robot (such as reachable distance) and the characteristics of the pieces, Sawyer cannot perform the insertion of all of them. From the eight thrusters, the robot is only able to insert the Yellow, Green and Orange. Therefore, the field in the ETL - Light Blue layer in Figure 8.1 - will present the multi-bump pattern depicted in Figure 8.4.

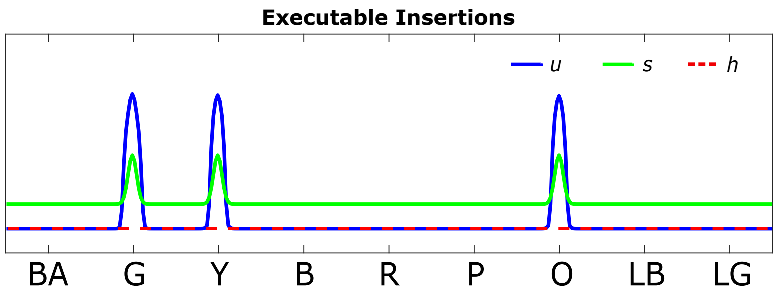


Figure 8.4: Executable Tasks Layer profile. Active sub-populations represent the color of the pieces that can be inserted by the robot. The robot cannot fully insert thrusters on the top of the structure (Red, Blue, Light Blue and Light Green) since two hands are needed to insert those. In that case, the robot can either instruct the human to insert or assist him by handing over the pieces. From the remaining thrusters, the only one out of the robot's reachability is the Pink. Therefore, the robot can only insert the Green, Yellow and Orange thrusters.

For the experiments, the inputs for the ETL were hand-coded. However, using information from the Vision System, for future work, it would be possible for the system to internally calculate if the motor action required to insert each one of the thrusters would be executable for the robot. That way, the ETL field would be generated autonomously.

8.1.3 Action Execution Layer

The Action Execution Layer is modeled as a two-dimensional field - colored in Orange in Figure 8.1 - and it computes the output-decision of the system. The use of a two-dimensional representation is inspired in the work of Ferreira (2014), where the author tested a bidimensional field in a robotic application scenario. Figure 8.5 shows the two-dimensional dynamic neural field, where the two dimensions represent "color" and "action". Input from the Sequence Layer contributes to the activation of the field corresponding to the color of the next piece. The dimension "action" receives input from the Object Memory Layer and the Executable Tasks Layer. A weighted sum of both inputs triggers activation in the region of the field corresponding to the action selected by the system. The activity that appears in both dimensions results in a two-dimensional bump, giving information regarding what is the piece that should be placed next in the structure and each one of the four motor actions suits the scenario the best. The area corresponding to the Base is already inhibited since that is the first sub-task achieved, triggering the next sub-goal, which in the case represented in the figure was the Yellow thruster.

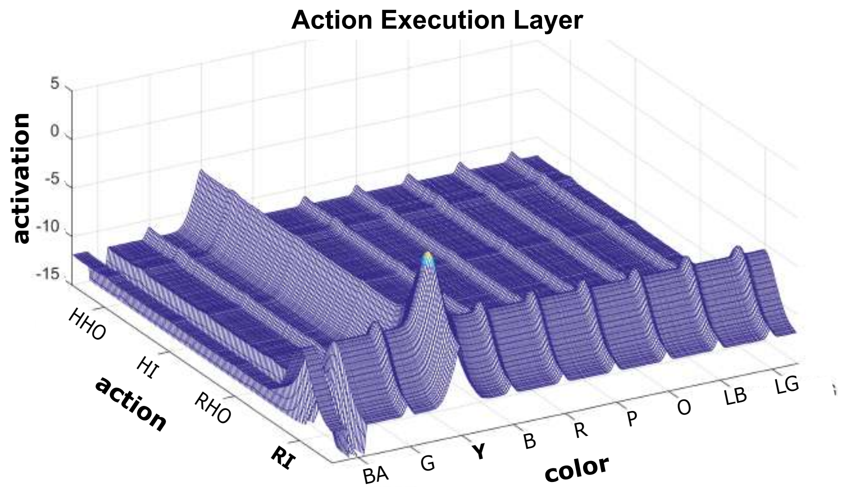


Figure 8.5: Action Execution Layer - output example. The active region of the field informs that the next step should be the insertion of the Yellow thruster (Y) and that the Robot should grab and insert the piece (RI).

8.1.4 Mathematical formulation

The mathematical base of each layer of the model is a Dynamic Neural Field, formalized by the equation (7.1).

Object Memory Layer

Each OML field, corresponding to the Human, Robot and Shared Workspaces, is designated as $u^{hw}(x^{hw},t)$, $u^{rw}(x^{rw},t)$ and $u^{sw}(x^{sw},t)$, respectively. The formulation of each field is identical, with the only difference being the visual input given to each field. To simplify, u^{oml} will designate each one of them, formalized by the equation below:

$$\begin{aligned}\tau^{oml}\frac{\partial u^{oml}(x^{oml},t)}{\partial t} &= -u^{oml}(x^{oml},t) + h^{oml}(x^{oml},t) + \varsigma^{oml}_{stoch}(x^{oml},t) \\ +f_\beta[u^{oml}(x^{oml},t)]&\left(s^{oml}(x^{oml},t) + \int w_{oml}(x^{oml}-y^{oml})f_0[u^{oml}(y^{oml},t)]dy^{oml}\right),\end{aligned} \tag{8.1}$$

where $oml = hw$, rw or sw for each field. The interaction kernel function w_{oml} used is an Oscillatory kernel (equation (4.7)) that permits the coexistence and stability of multiple bumps of activity. The implementation parameters for this field can be consulted in Appendix A.4.

- Input to the OML fields: s^{oml}

The input $s^{oml}(x^{oml}, t)$ is defined by the sum of the external visual input $v^{oml}(x^{oml}, t)$ to each field, with a hand-coded gain $C^{v \to oml}$ and the inhibitory interaction from the Past field u^{pa} to u^{oml} given by the integral term:

$$s^{oml}(x^{oml}, t) = C^{v \to oml} v^{oml}(x^{oml}, t) + \int f_0[u^{pa}(x^{pa}, t)] \underset{pa \to oml}{I}(x^{pa}, x^{oml}) dx^{pa}. \tag{8.2}$$

The integral term formalizes the propagation of the excitatory connections of u^{pa} to u^{oml}, $\underset{pa \to oml}{I}$, where only already achieved sub-goals will inhibit the corresponding sub-populations in the OML fields.

Executable Tasks Layer

The mathematical formulation of ETL is similar to the equation (8.1). $u^{etl}(x^{etl}, t)$ and $s^{etl}(x^{etl}, t)$ designate, respectively, the activation and input to the ETL field. The implementation parameters are described in the Appendix A.5.

- Input to the ETL field: s^{etl}

The Executable Tasks Layer holds information regarding what pieces can be inserted in the structure by the robot. The input given to the field depends on the location and the trajectory needed to execute each sub-task, and combined they infer the robot's capability to insert (or not) each thruster. In this implementation, the input was hand-simulated after heuristically testing what pieces could be inserted by the robotic system (see Figure 8.4).

Action Execution Layer

The AEL is implemented as a two-dimension field $u^{ael}(x^c, x^a, t)$, where x^c designates the dimension "color" and x^a the dimension "action" (see Figure 8.5). The formulation of the field can be generalized from the DNF-equation (4.1), as follows:

$$\begin{aligned}\tau^{ael}\frac{\partial u^{ael}(x^c,x^a,t)}{\partial t} =& - u^{ael}(x^c,x^a,t) + h^{ael} + s^{ael}(x^c,x^a,t) + \varsigma^{ael}_{stoch}(x^c,x^a,t)\\ &+ \int\int w_{ael}(x^c-q,x^a-s) f_0[u^{ael}(q,s,t)]dqds,\end{aligned} \tag{8.3}$$

where q and s represent the center distance within each dimension and w_{ael} is a two-dimensional kernel function with global inhibition (Ferreira, 2014), given by:

$$w_{ael}(x^c-q,x^a-s) = A\left[e^{\frac{-\frac{1}{2}(x^c-q)^2}{2\sigma^2}} * e^{\frac{-\frac{1}{2}(x^a-s)^2}{2\sigma^2}}\right] - w_{inh}. \tag{8.4}$$

The equation is a convolution of two gaussian kernel functions (4.5). Using this type of kernel, the competition between sub-neuronal populations is enabled, from where only one localized bump of activity will evolve. Implementation parameters are described in Appendix A.6.

- Input to the AEL field: s^{ael}

The input to the AEL field $s^{ael}(x^c,x^a,t)$ is given by two main sources, contributing to the dimensions x^c and x^a:

$$s^{ael}(x^c,x^a,t) = C^{c\rightarrow ael}s(x^c,t) + C^{a\rightarrow ael}s(x^a,t), \tag{8.5}$$

where $C^{c\rightarrow ael}$ and $C^{a\rightarrow ael}$ are the weight of the contributions from the components "color" and "action", respectively. $s(x^c,t)$ results from the propagation of the recall contribution from SL that encodes the sequence of sub-goals and the inhibitory connections from past sub-goals $\underset{pa\rightarrow ael}{I}(x^{pa},x^{ael})$:

$$\begin{aligned}s(x^c,t) =& C^{rec\rightarrow ael}\int f_0[u^{pa}(x^{pa},t)]\underset{pa\rightarrow pr}{A}(x^{pa},x^{pr},t)dx^{pa}\\ &+ \int f_0[u^{pa}(x^{pa},t)]\underset{pa\rightarrow ael}{I}(x^{pa},x^{ael})dx^{pa}.\end{aligned} \tag{8.6}$$

$s(x^a,t)$ is mathematically computed from the information received from the OML and ETL, according to the connections displayed in Figure 8.6. The connection were pre-defined to compute the expected action-output.

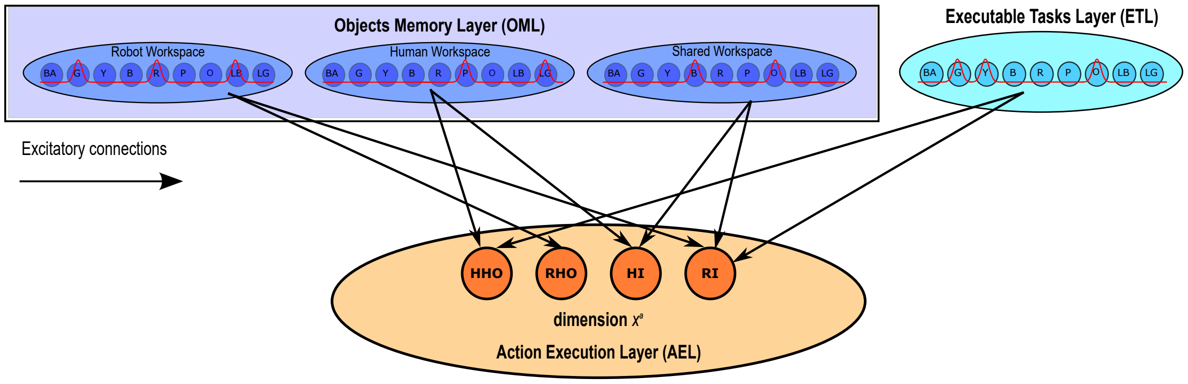


Figure 8.6: Excitatory connections from OML and ETL to AEL.

8.2 Results: What action should be taken to insert the next piece?

This section presents the results of the implementation of the architecture of Figure 8.1 in the HRC scenario presented in section 6.2.2. The goal was to test and validate the models in a scenario where Sawyer decides the most suitable action for each step of the construction and cooperates with the human co-worker. Moreover, the ability of the system to deal with missing items from the workspace was also tested.

The tests described next took place after conducting the learning experience presented in Chapter 7. Therefore, the synaptic weights of the learning matrix $\underset{pa\to pr}{A}$ were already defined, so it was expected that Sawyer would follow the last task-sequence that was completed with success in the third learning trial illustrated in section 7.2.4:

Yellow $\Rightarrow$ Pink $\Rightarrow$ Orange $\Rightarrow$ Light Green $\Rightarrow$ Green $\Rightarrow$ Red $\Rightarrow$ Blue $\Rightarrow$ Light Blue

Before each trial, the colored thrusters are distributed randomly across the three workspaces. At each step of the construction, the robot verbalizes to its co-worker the result of the selected decision - active region in the Action Execution Layer - and then generates the corresponding motor action.

8.2.1 Test number 1: Layout A

Figure 8.7 shows the activation profile of the OML fields at the beginning of the first test. Then, Figure 8.8 depicts the sequence of actions employed during the first test. A video of this trial can be found in https://youtu.be/XQwAz1PCuHw. In the beginning, the thrusters are displayed over the three workspaces according to the layout illustrated in Figure 8.8a.

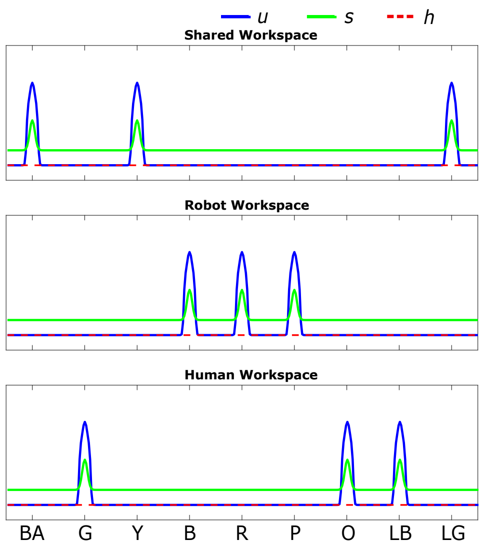


Figure 8.7: Test number 1 - Activation profile of the OML fields at the beginning of the construction. Green, Orange and Light Blue are placed in the Human Workspace; Blue, Red and Pink in the Robot Workspace; and finally Yellow and Light Green belong to the Shared Workspace.

Following the Sequence Layer, the Base triggers the Yellow as the next sub-goal to be achieved. Since Yellow is placed in the Shared Workspace and marked in ETL as a piece that the robot can insert, the selected action verbalized by the robot is “I think I should grab and plug the Yellow thruster”. As a result, Sawyer grabs the Yellow piece from the shared working table and inserts it in the structure (Figure 8.8b). The sub-population encoding Yellow is then inhibited in OML, SL and AEL, marking its insertion.

The insertion of Yellow triggers the activation of the next sub-goal, which according to SL is placing the Pink thruster. The output-decision in AEL can be seen in Figure 8.9. Note that sub-populations in the dimension “color” have a small pre-shape of activity - with the exception of the ones encoding past sub-goals (Base and Yellow), already inhibited in AEL - due to the OML connection informing about available pieces in the workspaces. The sub-neuronal population encoding Pink wins the competition over the others. Pink is placed in the Robot Workspace, but its insertion is not an executable action for Sawyer. Therefore, in the dimension “action,” the active population corresponds to “Robot Hands Over”. Consequently, Sawyer decides to grab the Pink thruster and pass it to the human co-corker that inserts the piece in the structure (Figure 8.8c). The same happens in the steps illustrated in Figures 8.8g and 8.8h.

The next sub-task is the insertion of the Orange piece, placed in the Human Workspace, which can be done

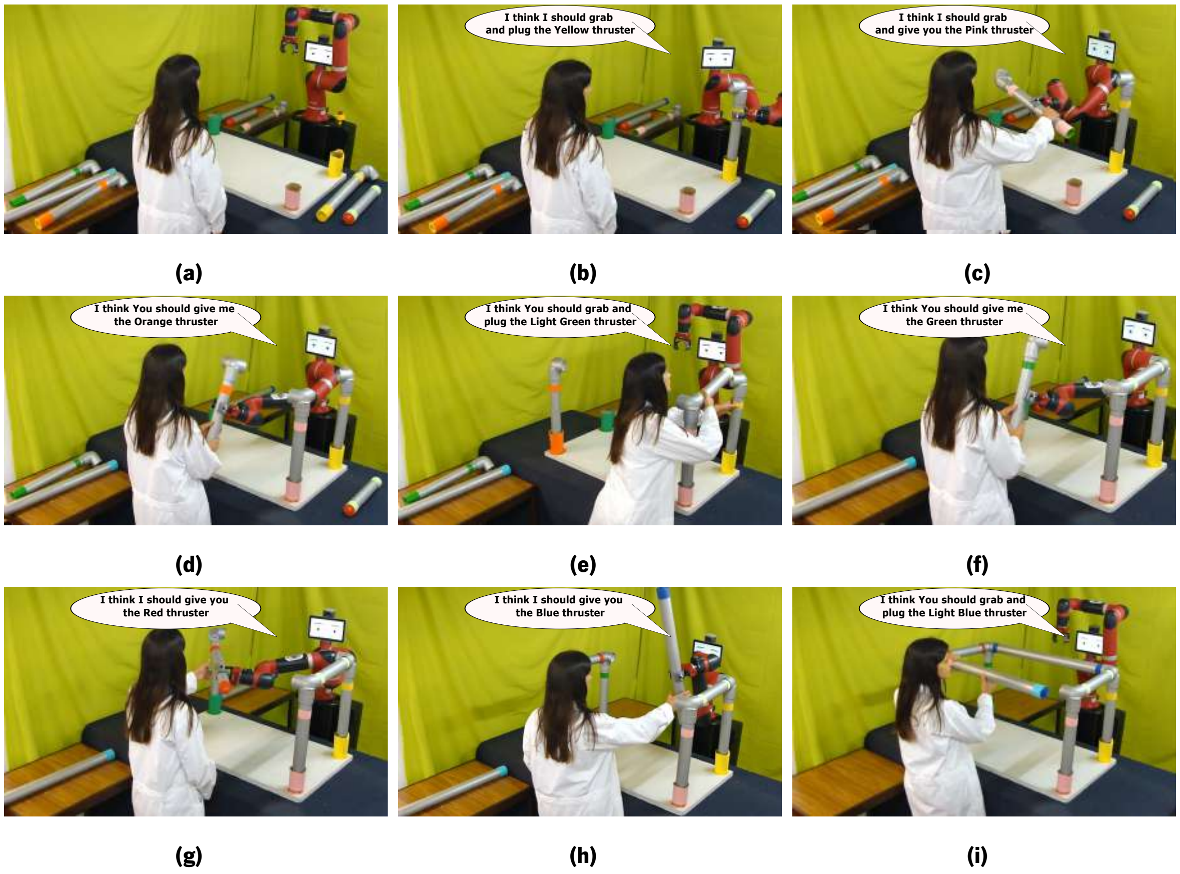

(a)
I think I should grab and plug the Yellow thruster
(b)
I think I should grab and give you the Pink thruster
(c)
I think You should give me the Orange thruster
(d)
I think You should grab and plug the Light Green thruster
(e)
I think You should give me the Green thruster
(f)
I think I should give you the Red thruster
(g)
I think I should give you the Blue thruster
(h)
I think You should grab and plug the Light Blue thruster
(i)

Figure 8.8: Test number 1 - snapshots illustrating the sequence of actions during the first trial.

by the robot. Therefore, Sawyer requests its co-worker to grab and pass the Orange thruster so the robot can perform the insertion (Figure 8.8d). The same action is taken in the step illustrated in Figure 8.8f to insert the Green. In the fourth step of the construction, the next sub-goal consists of inserting the Light Green thruster. Since it lays on the Shared Workspace and its insertion is not feasible for the robot, Sawyer requests its co-worker to grab the piece and insert it on his own: "I think You should grab and plug the Light Green thruster" (Figure 8.8e). The same action is selected in last step of the construction to insert the Light Blue (Figure 8.8i).

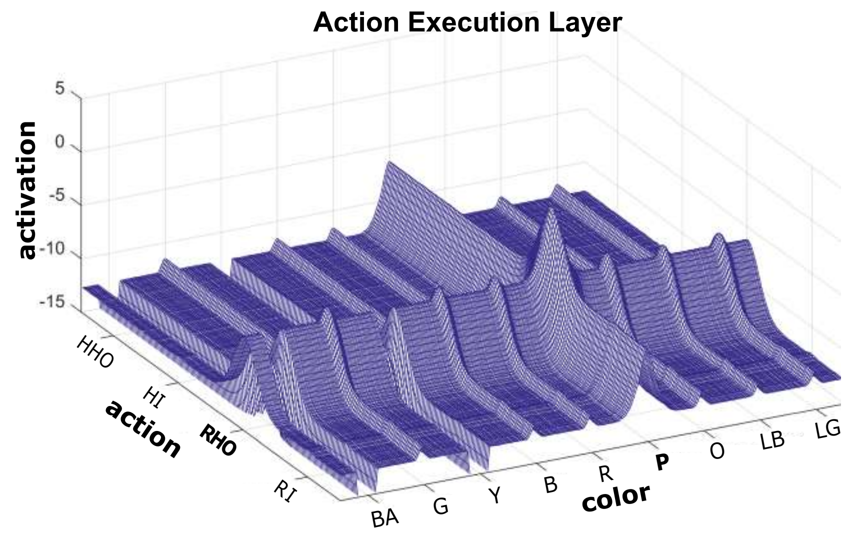


Figure 8.9: AEL activation profile in the second step of the construction. Sawyer's decision is to grab the Pink thruster and pass it to its human co-worker: Robot Hands Over (RHO).

8.2.2 Test number 2: Layout B

Secondly, a new test was conducted to validate the architecture with a different layout of the pieces across the workspaces. A video of this trial can be seen in https://youtu.be/nI6yCEVWJVI. Figure 8.10 shows the activation of the OML fields at the beginning of the trial. The thrusters were distributed by the three workspaces using a different layout, which implies that the actions selected by the robot to insert each thruster should be different.

Figure 8.11 shows a sequence of snapshots describing the actions selected at each step of the construction. The thrusters are distributed in the workplace according to the scenario of Figure 8.11a. The first sub-goal consists of inserting the Yellow thruster, which lays on the Robot Workspace. Consequently, Sawyer grabs and inserts the piece in the structure (Figure 8.11b). The insertion of Yellow triggers the next sub-goal - inserting Pink - and since the Pink thruster lays on the human's side and the robot cannot insert it, Sawyer instructs: "I think You should grab and plug the Pink thruster". The experiment continued until the construction was finalized. The decisions that emerged in the Action Execution Layer were correct and in accordance with the location of each thruster, following the expected sequence of sub-goals.

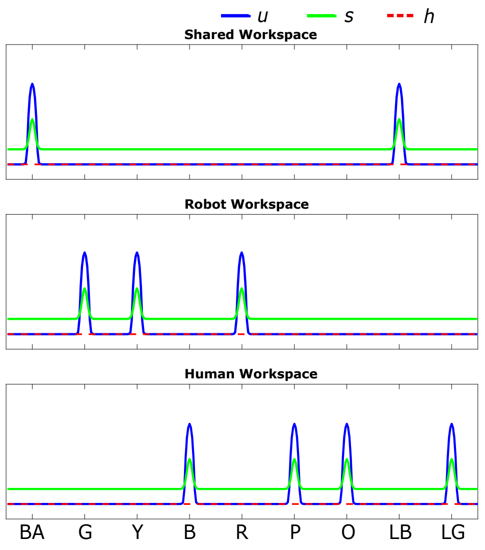
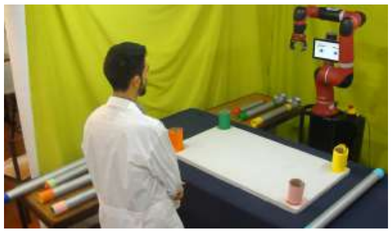
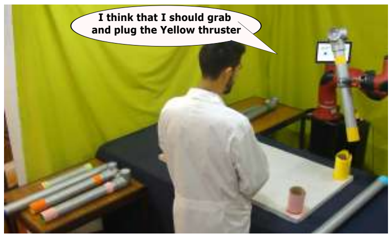


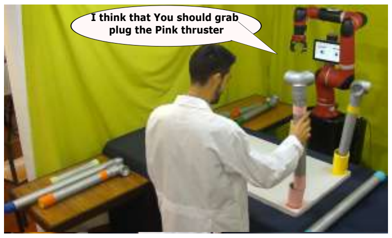
Figure 8.10: Test number 2 - Activation profile of the OML fields at the beginning of the construction. Blue, Orange, Pink and Light Green are placed in the Human Workspace; Green, Yellow and Red in the Robot Workspace; and finally Light Blue belongs to the Shared Workspace.



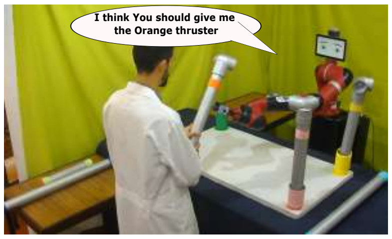
(a)



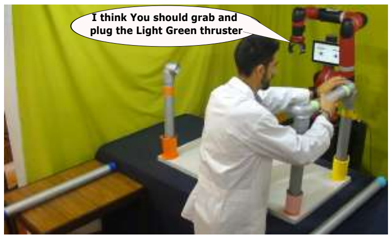
(b)



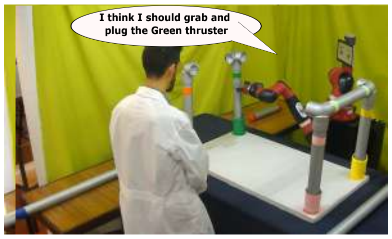
(c)



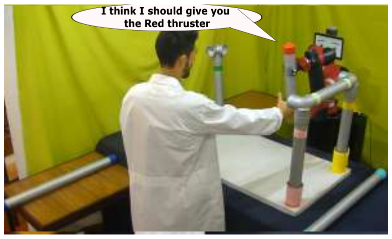
(d)



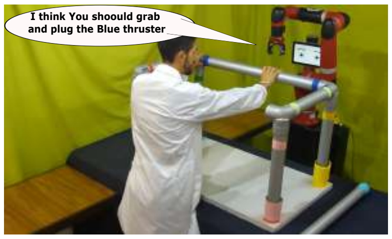
(e)



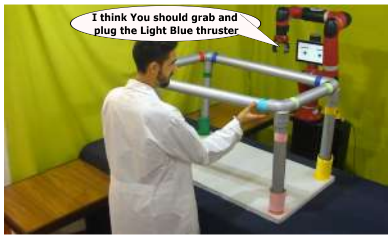
(f)



(g)



(h)



(i)

Figure 8.11: Test number 2 - snapshots illustrating the sequence of actions during the second test.

8.2.3 Test number 3: What if there is a missing piece?

The previous tests proved that it is possible to combine information from other layers to build an architecture for HRC in a real construction scenario. However, it is also important to evaluate the system's response when there is an item missing and what are the implications in the construction. In the third test, the Green thruster was removed from the workplace - layout A -, previously used in the first test (see Figure 8.8a). Therefore, the sub-population encoding the Green piece has no activity in any of the OML fields (Figure 8.12). A video of this trial is available at https://youtu.be/QO1ZqSpotfs.

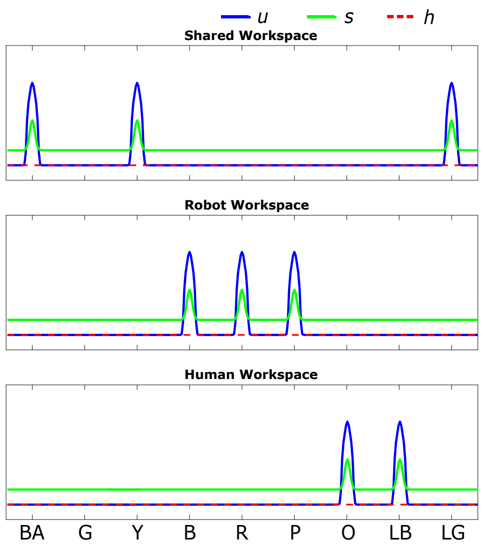
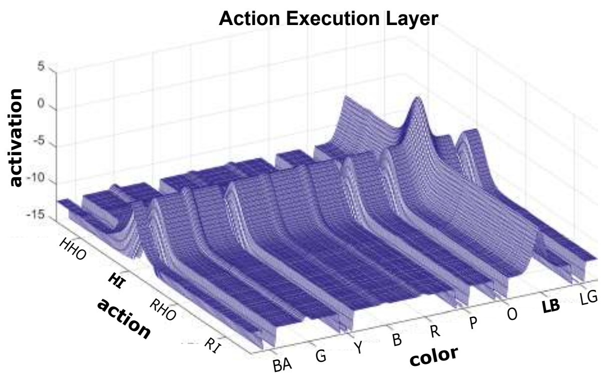
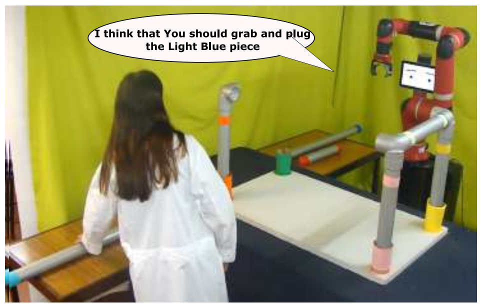


Figure 8.12: Test number 3 - Activation profile of the OML fields at the beginning of the construction. The Green thruster is missing from the workplace.

The first four steps of the construction were the same used in the first test. Sawyer and its human co-worker followed the same steps illustrated in Figures 8.8b-e to insert the Yellow, Pink, Orange and Light Green thrusters. At this step, the next sub-goal, according to the learned sequence, should be the insertion of the Green piece. However, Green is not available in any of the workspaces. The AEL activation profile after inserting the Light Green thruster is depicted in Figure 8.13a. Since the sub-neuronal population encoding the Green piece did not receive input from the Object Memory Layer, the pre-shaped activation is too low compared to other sub-populations. Therefore, the winning decision goes to the Light Blue that had already received excitatory input triggered by the Pink and Orange sub-populations (Figure 8.13b).



(a) AEL activation profile



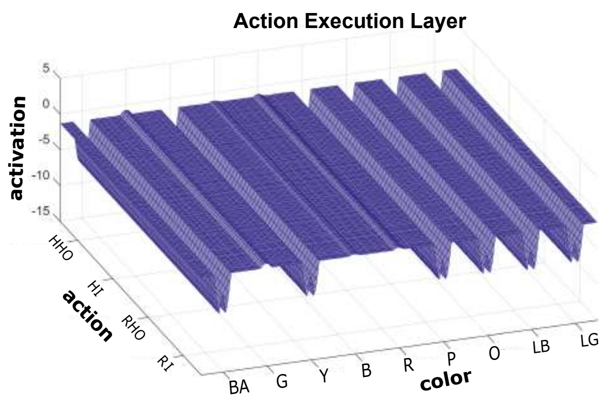
(b) Robot verbalizes to its co-worker that she should insert the Light Blue thruster

Figure 8.13: Sawyer's decision in the fifth step of the construction. Since the Green piece is not available, the next item according to the information stored in SL - Light Blue - is retrieved because it has enough activation to be recognized in the AEL as a valid sub-goal to continue the construction.

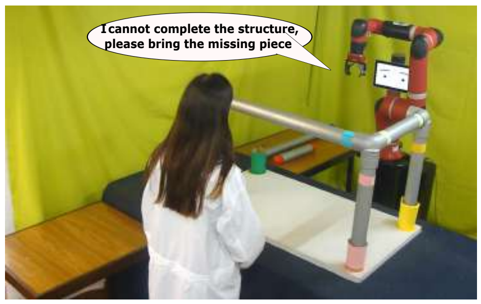
After placing the Light Blue, only two pieces remain in the workplace: Blue and Red. However, they cannot be placed in the structure without inserting the Green first. Figure 8.14 illustrates the result of the decision process at this step. None of the available items in the workspaces received enough input to evolve as a bump of activity in AEL (Figure 8.14a). From the input received from the OML and SL, the system recognizes that the structure is not yet completed. The absence of activity in AEL, combined with the previous information, leads the robot to realize that there is a missing item. Therefore, the construction cannot be completed (Figure 8.14b).

To continue the task, the human co-worker brings the Green piece and places it in her workspace, triggering a peak of activity in the sub-population encoding the Green. Subsequently, the input from OML causes activation in the corresponding sub-population of AEL, and the model retrieves the insertion of Green as the next sub-goal (Figure 8.15). The task continued with the insertion of the Red and Blue thrusters. The final sequence used in the task was the following:

$$\text{Yellow} \Rightarrow \text{Pink} \Rightarrow \text{Orange} \Rightarrow \text{Light Green} \Rightarrow \text{Light Blue} \Rightarrow \text{Green} \Rightarrow \text{Red} \Rightarrow \text{Blue}$$

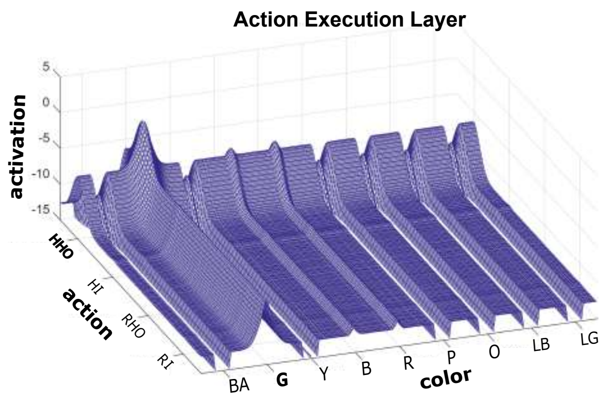


(a) AEL ativation profile

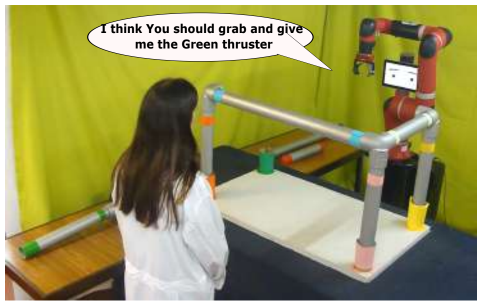

I cannot complete the structure, please bring the missing piece

(b) Robot verbalizes to its co-worker that it is not possible to complete the structure without the missing item

Figure 8.14: Sawyer's decision in the sixth step of the construction. None of the available items had enough activation to trigger a new decision. The model recognizes a long period without activation as either the end of the task or an impossibility to complete it caused by missing items. Since there are still items in the OML field, the robot realizes that a piece is missing from the workplace.



(a) AEL ativation profile



(b) Sawyer verbalizes to its co-worker that he should pass the Green piece to the robot

Figure 8.15: New decision in the sixth step of the construction. The input from the Green is now given by the OML. A bump of activity emerges in the AEL sub-population encoding the Green. The robot requests its co-worker to pass the Green so Sawyer can insert it.

8.3 Discussion

The experience presented in this Chapter had two main objectives: 1) use of knowledge about the construction sequence learned in Chapter 7 integrated into the cognitive architecture for Human-Robot joint action; and 2) validating the systems' architecture to represent workspace information and action selection (now modeled as a two-dimensional field) in the real HRC scenario.

The Action Execution Layer implemented was formalized as a two-dimensional Dynamic Neural Field representing the dimensions "color" and "action". The Sequence Layer provided task-relevant information to the dimension "color" during the trials to infer the color of the piece that should be inserted at each step. Two additional layers were incorporated to add information regarding the actions that should be taken by Sawyer and its co-worker in the construction of the structure. The Object Memory Layer encodes the item's location during the construction, marking the pieces placed in each workspace. The Executable Tasks Layer represented the actions that could be performed by Sawyer. By combining information from both layers, the system could select the most suitable action at each step of the task. The use of a two-dimensional field to aggregate information from different sources allowed to represent two different features in the same field and compute the possible decisions for the different combinations of "color" and "action". This type of representation was also employed in robotic applications by Faubel and Schöner (2008) to represent objects using a small number of features, showing how two-dimensional fields can be used in this type of application to represent information and/or make decisions.

The experiments conducted during the first and second tests validated the architecture for different layouts, confirming that the model can output correct decisions taking into consideration the information provided by the different layers. A third test was performed to study the architecture's behavior facing a situation where items were missing from the workspace environment. The results proved that the architecture developed can deal with this case. When facing a situation in which the Green piece was not present in any workspace, the system was able to retrieve a valid sub-goal - the insertion of Light Blue - due to the input received from the SL, where the task-constraints were previously acquired from the experiments performed in Chapter 7. Subsequently, when there were no more valid insertions to be made, the system was able to recognize that the structure was not complete and, combining information from the OML, realize that there was an item missing. After the human co-worker places the Green thruster back into the workspace, the system was able to continue the construction successfully. The OML and ETL connections to AEL (see Figure 8.6) were pre-defined. However, for future work, it would be important to explore if those could also be learned from demonstration, in the same way the robot learns the sequence of sub-goals to build the structure.

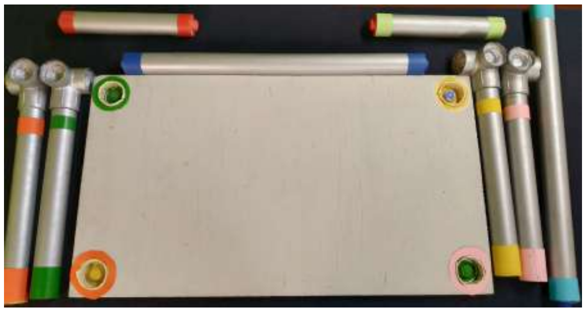
Chapter 9

Towards learning sequences with time constraints

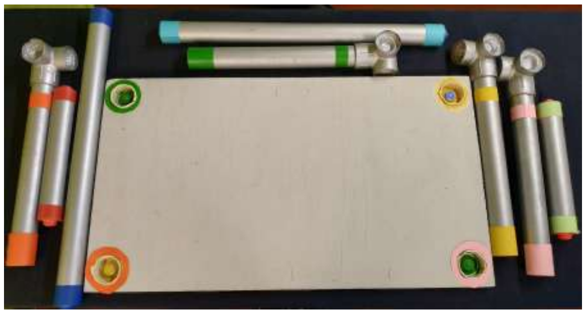
The Short-Term mechanism for memorizing sequential events presented in Chapter 7 allowed the robot to acquire the serial order of a sequential task from demonstration. When acquiring task-relevant information, another useful feature to be learned is the time-interval between assembling steps. Having in mind that collaborative robots are expected to work side-by-side with several operators, the ability to adjust to different time-behaviors will benefit the cooperation between Robot and Human. In this Chapter, a model for memorizing, from demonstration, sequential tasks with time constraints is presented. The model was validated in a real experiment where Sawyer memorizes, in one-shot, information regarding both ordinal and temporal aspects of the task, demonstrated by different tutors with different behavioral time-scales (younger and older).

9.1 Experimental setup

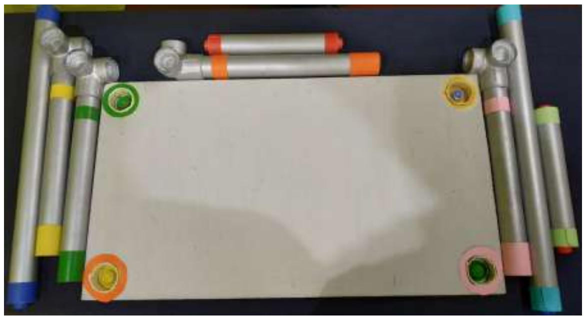
As a test scenario, the construction of the structure of the previous experiments was used, according to the scenario described in section 6.2.3. For the demonstration, three different layouts were considered - that implied three different assembly sequences - and two tutors with different behavioral time scales. The placement of the thrusters in each scenario can be seen in Figure 9.1. A tutor demonstrates how to assemble the thrusters while the robotic co-worker observes and memorizes the serial order and timing of each assembly step. Later, the robot takes the role of a tutor itself and recalls the memorized sequence to a different operator, maintaining the order and time interval previously observed. Depending on how the pieces are placed on the workplace table, different sequences can be used to assemble all the parts, which will require the robotic platform to memorize multiple possible sequences to build the structure. Moreover, different tutors will build the sequence using different intervals of time: as a concrete example, an older tutor may take longer to reach and insert all the thrusters than a younger one.



(a) Layout A



(b) Layout B



(c) Layout C

Figure 9.1: Layout scenarios used during the experiments.

9.2 DNF-model for memorizing sequences with time constraints

The model presented in this Chapter is also based on Dynamic Neural Fields, and it is inspired by previous research on memory mechanisms for order and time interval of sequential processes (Ferreira et al., 2011; Ferreira et al., 2014; Wojtak et al., 2018). Figure 9.2 presents a model for memorizing and storing several sequences from demonstration. Each memorized sequence is stored in a dynamic neural field, designated by Sequence Memory field - $u^{SM}(x,t)$ - that holds a sequence of stimulus events as a multi-bump activation pattern, similar to the fields generated in the Short-Term Memory Layer of the model described in Chapter 7. According to the context behind each demonstration trial - that is, the position of the objects in the workplace and the characteristics of the tutor (younger or older) -, several sequences with different order and timing are stored in the model.

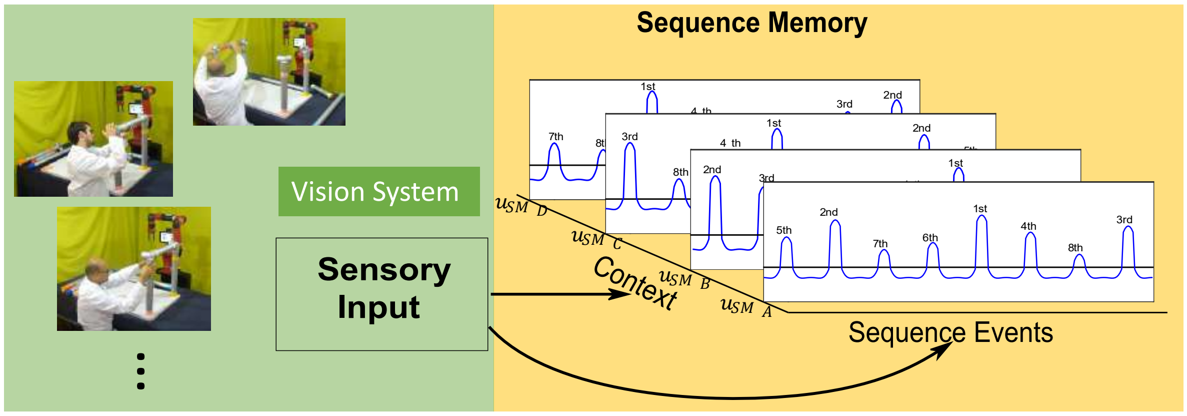


Figure 9.2: Sketch of the sequence memory process. A bump represents an event triggered through excitatory input from the Vision System. The strength of each memory representation reflects the time elapsed since stimulus presentation, resulting in an activation gradient from the first to the last event.

Figure 9.3 depicts an overview of the Sequence Recall procedure. Taking into consideration a specific context, one of the patterns in the Sequence Memory fields is selected from the group of stored fields. Then, the Sequence Recall field - $u^{SR}(x,t)$ - receives the multi-bump pattern of u^{SM} as an input lowered below the threshold level 0 (black line in u^{SR}). During the recall process, the continuous growth of the baseline activity in u^{SR} takes all sub-neuronal populations, one by one, closer to the threshold for the evolution of self-stabilized peaks of activity. When the currently most active population reaches the threshold level, the corresponding output-action is triggered, which consists of verbalizing what is the color of the piece that should be inserted at that moment. Simultaneously, the excitatory-inhibitory connections between associated populations in u^{SR} and the Past Events field - $u^{PE}(x,t)$ - ensures that the suprathreshold activity marking the latest sub-goal achieved becomes first stored in u^{PE} and subsequently suppressed in the Sequence Recall field.

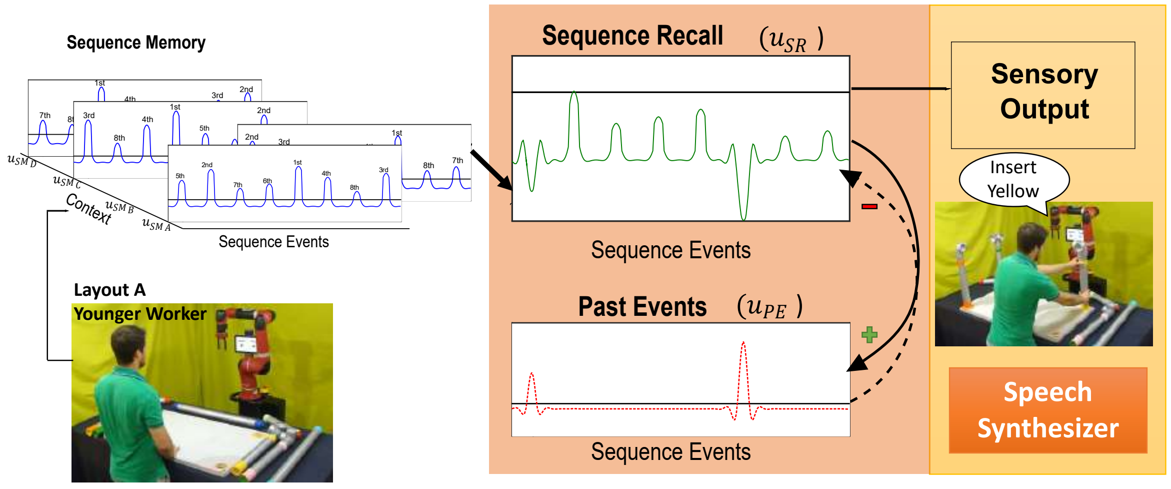


Figure 9.3: Sketch of sequence recall process. Dashed lines indicate inhibitory connections, solid lines excitatory connections.

9.2.1 Mathematical formulation

The dynamics of each Sequence Memory field u^{SM}, the Sequence Recall field u^{SR} and the Past Events field u^{PE} are formulized by the following equations, respectively (Amari, 1977; Ferreira et al., 2011; Cunha et al., 2020):

$$\begin{aligned}\tau^{SM}\frac{\partial u^{SM}(x,t)}{\partial t} &= -u^{SM}(x,t) + s^{SM}(x,t) + h^{SM}(x,t) \\ &+ \int w(x-y) f\left(u^{SM}(y,t)\right) \mathrm{d}y,\end{aligned} \tag{9.1}$$

$$\begin{aligned}\tau^{SR}\frac{\partial u^{SR}(x,t)}{\partial t} &= -u^{SR}(x,t) + \int w(x-y) f\left(u^{SR}(y,t)\right) \mathrm{d}y \\ &- \int w(x-y) f\left(u^{PE}(y,t)\right) \mathrm{d}y + u^{SM}(x) + h^{SR}(t),\end{aligned} \tag{9.2}$$

$$\begin{aligned}\tau^{PE}\frac{\partial u^{PE}(x,t)}{\partial t} &= -u^{PE}(x,t) + \int w(x-y) f\left(u^{SM}(y,t)\right) \mathrm{d}y \\ &+ u^{SR}(x,t) f(u^{SR}(x,t)) + h^{PE},\end{aligned} \tag{9.3}$$

where $u^{SM}(x,t)$, $u^{SR}(x,t)$ and $u^{PE}(x,t)$ represent the activity at time t of a neuron tuned to the feature value x. The parameters $\tau^{SM}, \tau^{SR}, \tau^{PE} > 0$ are the time scale of each field. The interaction kernel function $w(x)$ implements the coupling between neuronal populations. An oscillatory kernel was used (equation (4.7)) to enable multi-peak solutions. Implementation parameters can be found in Appendix B.

The function $s^{SM}(x,t)$ represents the time dependent localized input at populations in location x that consists on the input prevenient from the Vision System. The strength of individual memory representations in u^{SM} is controlled by the baseline resting level $h^{SM}(x,t)$:

$$\frac{\partial h^{SM}(x,t)}{\partial t} = \Big(1 - f(u^{SM}(x,t))\Big)\Big(-h^{SM}(x,t) + h_0^{SM}\Big) + \frac{1}{\tau^{h^{SM}}} f(u^{SM}(x,t)), \tag{9.4}$$

where $h_0^{SM} < 0$ is the resting value to which h^{SM} decays in the absence of suprathreshold activity at position x and $\tau^{h^{SM}}$ measures the growth rate when activity is present. The baseline activity $h^{SR}(t)$ grows continuously over time as described by the following equation:

$$\frac{\partial h^{SR}(t)}{\partial t} = \frac{1}{\tau^{h^{SR}}}, h^{SR}(t_0) = h_0^{SR} < 0\ , \tag{9.5}$$

where $\tau^{h^{SR}}$ defines the growth rate of h^{SR}. The baseline activity $h^{PE} < 0$ is constant.

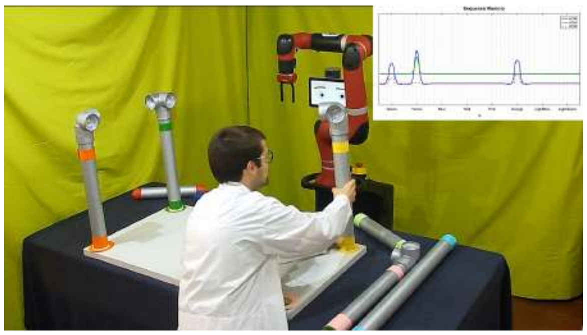
9.3 Experimental Results

This section describes the experimental results of the learning of the sequential order and timing of a construction task using the implemented model described in the previous section. Two tutors with different behavioral time scales, who already knew how to build the structure, were asked to demonstrate it to the robot, starting from three different Layouts (A, B and C, Figure 9.1). During the demonstration, Sawyer pays attention to the tutor and stores the sequential order and time interval of each step for all demonstrations. Afterward, the robot acts as a tutor and instructs two other inexperienced co-workers with no knowledge of the construction task. From the stored sequences, the system selects the most suitable one according to the context of the trial, that is, the characteristics of the new worker and the layout of the pieces in the table. Sawyer recalls the selected sequence, verbalizing the color of the piece that should be inserted, according to the order and time stored in the Sequence Memory field during the demonstration trials.

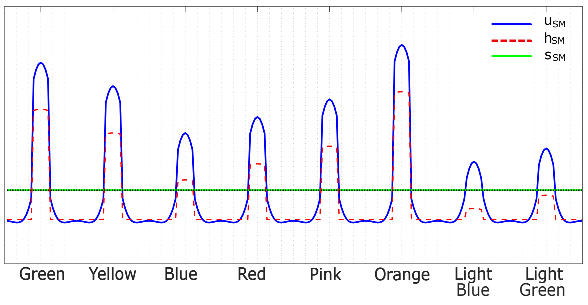
9.3.1 Learning the sequence order and timing

During the demonstration process, when the system detects that one of the thrusters is being inserted, an input stimulus is generated in the location of the population of neurons encoding the respective colored thruster, which leads to a bump of activity in the u^{SM} field, forming a peak that grows gradually as a function of time. Figure 9.4 pictures the demonstration experiment performed, where both tutors show to the robotic co-worker how to assemble the structure. In the scenario displayed in Figures 9.4a and 9.4b, the pieces are distributed in the work table according to Layout A. A video of this trial can be seen in the following link: https://youtu.be/YTZTDJzzGYw. Next, Figures 9.4c and 9.4d show the Sequence Memory fields stored during the demonstration, where the amplitude of the bumps encodes the serial order of the inserted thrusters, with the highest peak (orange) being the color of the first inserted piece and the shortest one (light blue) being the last. Although the sequence used by both tutors is the same, the intervals of time between each assembly step in both trials are considerably different, which will be demonstrated in section 9.3.2.

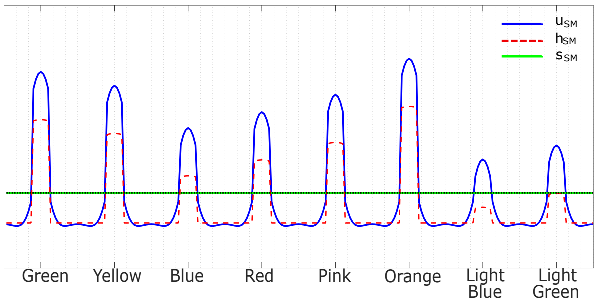
(a) Older tutor inserting the Orange thruster (1st piece)



(b) Younger tutor inserting the Yellow thruster (3rd piece)



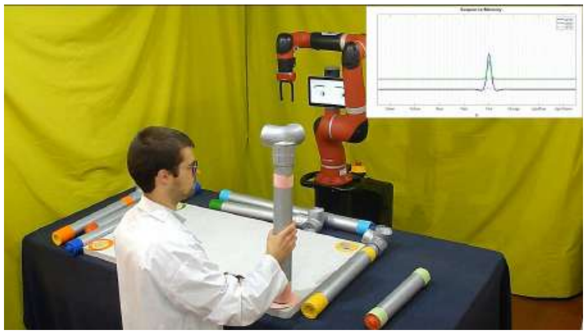
(c) Sequence Memory Field A demonstrated by the **Older tutor**



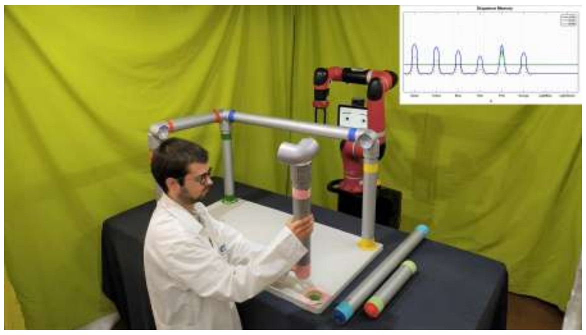
(d) Sequence Memory Field A demonstrated by the **Younger tutor**

Figure 9.4: Two different tutors assembling Sequence A: Orange → Green → Yellow → Pink → Red → Blue → Light Green → Light Blue.

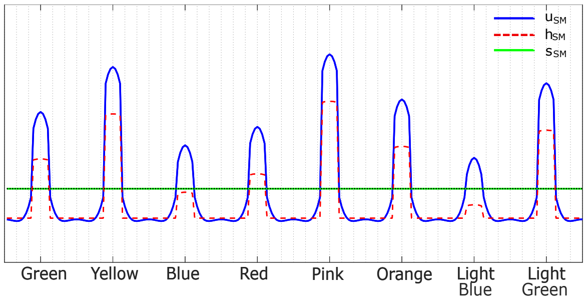
Subsequently, Figure 9.5 portraits the demonstration of two other sequences, this time by the younger tutor. In Figure 9.5a, the thrusters are distributed in the table according to Layout B, so the tutor starts the construction by inserting the pink thruster, opting for a different assembly sequence. A complete video of this trial can be found at https://youtu.be/0F8T_d_y2xs. Likewise, in the scenario displayed in Figure 9.5b, the tutor was asked to build the structure starting from Layout C, which resulted in a new sequence. After the demonstration of each trial, the information stored in both fields - Sequence Memory B (Figure 9.5c) and Sequence Memory C (Figure 9.5d) - is stored in the model, so afterward Sawyer can use the memorized sequences to teach inexperienced workers, taking into account the Layout of the pieces that constitute the sequence.



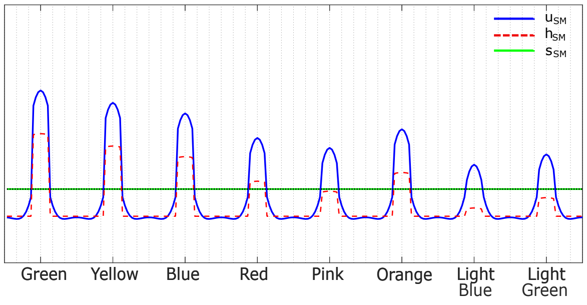
(a) Younger tutor assembling **Sequence B**, starting from the Pink thruster (1st piece).



(b) Younger tutor assembling **Sequence C**, inserting the Pink thruster (6th piece)



(c) Sequence Memory Field B: Pink → Yellow → Light Green → Orange → Green → Red → Blue → Light Blue



(d) Sequence Memory Field C: Green → Yellow → Blue → Orange → Red → Pink → Light Green → Light Blue

Figure 9.5: Younger tutor assembling Sequence B and C.

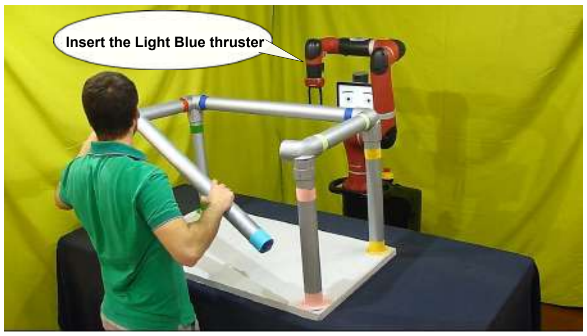
9.3.2 Recalling the memorized sequences

In order to verify if the system was able to store the time interval between each construction step, two workers with no previous knowledge of the task were asked to collaborate with Sawyer and follow its instructions to learn the assembly steps of the sequence A, previously demonstrated by the two previous tutors, with different time intervals. The robot was programmed to verbalize each step, taking into consideration the timing between each insertion.

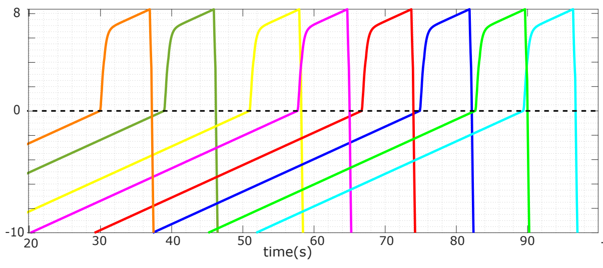
Figures 9.6a and 9.6b illustrate Sawyer as a tutor, guiding two different workers through the construction steps, while Figures 9.6c and 9.6d illustrate the time course of the maximal activation of each sub-population of neurons when the task was assembled by each worker, according to the instructions given by Sawyer. A video example can be found at https://youtu.be/Vn0_1raKq4I.

Each instruction is verbalized when each sub-population of neurons encoding the respective thruster in the Sequence Recall field reaches the threshold level, as stated in section 9.2. As shown in both figures, all sub-neuronal populations seem to have a pre-activation strength that respects the temporal order of the task steps, learned during the demonstration process. The first thrusters (orange) are inserted at t=30s (older tutor trial) and t=27s (younger tutor trial), while the last ones (light blue) are placed at t=97s (older tutor trial) and t=85s (younger tutor trial). By comparing both trials (Figure 9.6e), one can see that the older worker was slower than the younger in the majority of the steps and the younger worker took less time to perform the full task, as it should be expected since the co-workers are following the order and time memorized in the previous demonstrations (Figure 9.4).

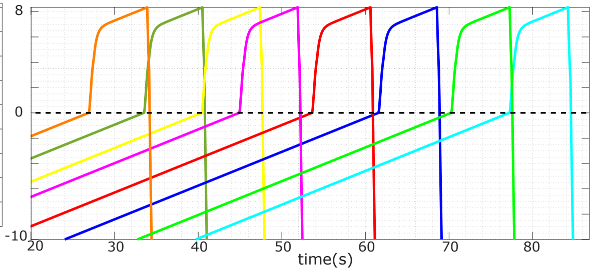
(a) Older worker following Sawyer's instruction to assemble Sequence A



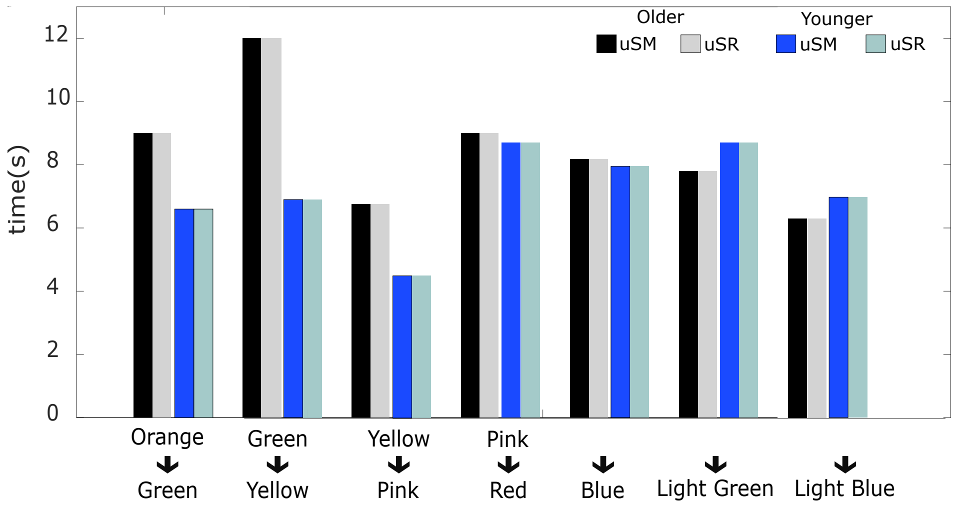
(b) Younger worker following Sawyer's instruction to assemble Sequence A



(c) Older worker: Total time of 67s **(slower)**



(d) Younger worker: Total time of 58s **(faster)**



(e) Contrast between the time intervals of consecutive assembly steps during the construction of the sequence A, performed by an older (slower) and a younger (faster) worker

Figure 9.6: Co-worker Sawyer recalls the memorized sequence, respecting the time interval of each demonstration.

9.4 Discussion

In this Chapter, a rapid learning system that allows the robot to memorize knowledge about ordinal and temporal aspects of sequential tasks and store it in a single Dynamic Neural Field was proposed and tested. This model allows codifying two different features of the task - order and time - in a unique representation that consists of the level of activation of each sub-neuronal population encoding each sub-goal of the task. After learning, the recall of the memorized information can be used by the robot to instruct inexperienced human operators, in the same context. Similar to the Short-Term Memory mechanism presented in Chapter 7, a single demonstration is sufficient for the robot to memorize the ordinal and temporal aspects of a sequence, thus minimizing the efforts of the human tutor to train the robot. Moreover, as seen in the same Chapter, the recall of the stored information can be used as input to a long-term learning mechanism that allows the robot also to learn the connections between the several sub-tasks. As future work, it should be explored if this new model could replace the previous STM process to allow the system to also learn the time constraints between the several sub-goals of the task.

Endowing a collaborative robot with the ability to predict, not only the sequence structure but also the time interval between sequential events is vital for efficient coordination of actions and decisions in time, in Human-Robot joint action tasks. It allows the robot to anticipate what the human operator will need, or will do, and when it should start an adequate complementary behavior in the service of the joint task.

Part V

General discussion and future work

Chapter 10

General discussion and Future Work

10.1 Conclusions

The ability to acquire information and learn from the environment is proved to be a fundamental capacity for the emerging generations of collaborative robots that are expected to work side-by-side with humans in several tasks. Inspired by neuro-cognitive findings on how humans learn from each other, some roboticists have started to use the Dynamic Neural Fields framework to model cognitive capacities in robots. Learning from demonstration and acquiring sequential information are the key paradigms explored in this dissertation.

The experiments presented in Part IV, applied DNF-based models to endow the collaborative robot Sawyer with the capacity to extract task-relevant information from demonstration in real scenarios where Robot and Human work together to achieve mutual goals. A model for acquiring the serial order of a task is proposed in Chapter 7, inspired by previous work conducted in the Mobile and Anthropomorphic Robotics Laboratory. The model was validated in a scenario where Sawyer acquires, from tutor demonstration and verbal feedback, the sequential structure and the precedence relations between sub-goals of an assembly task. By adding a Short-Term Memory mechanism for memorizing sequential information, it was possible to reduce the number of demonstration trials necessary to train the network for learning and thus reducing the workload for the human tutor. The results were satisfactory and revealed that Sawyer could acquire the constraints of the task successfully.

Furthermore, the learned information was employed in an architecture that incorporated workplace knowledge in a scenario where Human and Robot had to coordinate their actions in space to insert each one of the pieces that compose the structure. The additional layers provided information regarding object location to help the robotic system in selecting the most suitable action in each step. For that, a two-dimensional DNF was implemented to represent in a single field the two distinct features of the new process: the color of the next piece and the action that need to be performed to insert it. The output of the system satisfied the expected results, also validating the architecture in a situation where one of the pieces was missing. The benefit of this new model is that, instead of stopping when facing an unpredictable situation, the system takes advantage of the knowledge of the task acquired before to continue and compute a different decision considering the state of the workplace, which brings greater flexibility for the task.

For better coordination of actions in HRC scenarios, another critical capability is to know the time interval between events. Chapter 9 describes the first steps towards learning sequential information with time constraints. It follows the work of Ferreira et al. (2014) and Wojtak et al. (2018) that started to employ the DNF-framework to apply temporal information in robotic tasks. The tests performed pictured a scenario where the robot memorizes, in a single demonstration, information regarding the temporal and sequential order of the task codified in the same DNF representation. The model is able to store several multi-bump patterns in different fields according to the context associated with each demonstration, which in the case implemented corresponded to the different workspace layouts and time-behaviors of the different tutors. The results of this work allowed the robot to evolve from being in a learning position to be in a teaching position and instruct inexperienced co-workers facing similar contexts. That means that the robot can rapidly acquire information and transpose it to other co-workers without having to be pre-programmed for any task, having learned the sequence and time intervals entirely from the demonstrations.

10.2 Future Work

Although the work performed has achieved good results and the objectives for this dissertation have been achieved, there is room for improvement and exploration of the models developed.

While the models presented in Chapter 7 allowed to successfully acquire the connections between the several sub-goals from the demonstration, the architecture of Chapter 8 had mostly pre-defined connections between its several layers. It would be important to explore how the connections between item's representations in the Object Memory Layer and Executable Tasks Layer to the decision process in the Action Execution Layer (Figure 8.6) can be obtained without the need to hand-code them. The possibility of acquiring these connections also from demonstration remains to be explored. Moreover, while tests were made to study how the architecture of Figure 8.1 behaves when one piece is missing from the workplace, further experiments should be made to test what would be the output of the models in the absence of multiple pieces.

Additionally, Chapter 9 provided insights on a different matter: acquiring temporal order of a sequential task, also from demonstration. The models presented had the purpose of providing the base mechanism for coordinating actions and decisions in time in joint tasks between robots and humans. Although proving to be successful in memorizing the time interval between successive events in a sequence, it remains unexplored the possibility of acquiring the temporal constraints of a multiple-sequential task, the same way the system acquires the precedence relations between sub-goals in Chapter 7. That would imply the estimation/extrapolation of the interval of times between events that were not sequentially observed, a feature that could be incorporated in the architecture of Chapter 8 to allow the coordination of decisions and actions not only in space but also in time. With the integration of temporal information, it could be possible for the Robot to anticipate its human co-worker actions and starting handing over the pieces. That would allow Robot and Human to construct the structure using simultaneous motor-actions, increasing the autonomy of the Robot, and reducing the construction duration.

Part VI

Appendices

Appendix A

Implementation parameters - Chapters 7 and 8

A.1 Sequence Learning Layer (SLL)

Past Field u^{pa}

τ^{pa}	Time constant of u^{pa}	2.00
A^{pa}	Amplitude of the kernel	4.00
α^{pa}	Spatial phase of the kernel	0.50
k^{pa}	Decay rate of the kernel	0.30
ς^{pa}_{stoch}	Stochastic noise	0.30
$C^{v \to pa}$	Gain parameter controlling the strength of contribution v^{pa} to s^{pa}	1.05
$\underset{pr \to pa}{E}$	Value of the excitatory connection from u^{pr} to u^{pa}	5.00
H^{pa}_{dec}	Resting level	-4.50
H^{pa}_{high}	Higher resting level for the recall mode	-0.50
H^{pa}_{low}	Lower resting level for the learning mode	-1.00
τ^{pa}_{h}	Time constant of $h^{pa}(x^{pa}, t)$	10.00

Present Field u^{pr}

τ^{pr}	Time constant of u^{pr}	2.00
A^{pr}	Amplitude of the kernel	9.40
σ^{pr}	Standard deviation of the gaussian curve	7.00
λ^{pr}	Learning threshold of the Present field	6.00
w^{pr}_{inh}	Global inhibition of the kernel	8.80
ς^{pr}_{stoch}	Stochastic noise	3.50
$C^{v\rightarrow pr}$	Gain parameter controlling the strength of contribution v^{pr} to s^{pr}	17.00
$C^{rec\rightarrow pr}$	Gain parameter controlling the strength of the Recall contribution to s^{pr}	2.20
$C^{stm\rightarrow pr}$	Gain parameter controlling the strength of contribution of the STM Layer to s^{pr}	0.90
$\underset{pa\rightarrow pr}{I}$	Value of the inhibitory connection from u^{pa} to u^{pr}	8.00
H^{pr}	Resting level	-1.60
τ_a	Time scale of the differential equation defining a synaptic weight variation	10.00

A.2 Short-Term Memory Layer (STM)

τ^{stm}	Time constant of u^{stm}	1.00
A^{stm}	Amplitude of the kernel	1.50
α^{stm}	Spatial phase of the kernel	0.31
k^{stm}	Decay rate of the kernel	0.30
ς^{stm}_{stoch}	Stochastic noise	0.00
$C^{v\rightarrow stm}$	Gain parameter controlling the strength of contribution v^{stm} to s^{stm}	7.00
H^{stm}_{dec}	Baseline value of $h^{stm}(x^{stm}, t)$	-4.00
τ^{stm}_{h}	Time constant of $h^{stm}(x^{stm}, t)$	50.00

A.3 Rehearsal Mode

Rehearsal Epochs	10
Steps per Epoch	400

A.4 Object Memory Layer (OML)

τ^{oml}	Time constant of u^{oml}	2.00
A^{oml}	Amplitude of the kernel	2.00
α^{oml}	Spatial phase of the kernel	0.50
k^{oml}	Decay rate of the kernel	0.30
ς^{oml}_{stoch}	Stochastic noise	0.00
$C^{v\to oml}$	Gain parameter controlling the strength of contribution v^{oml} to s^{oml}	2.00
$\underset{pa\to oml}{I}$	Value of the inhibitory connection from u^{pa} to u^{oml}	8.00
H^{oml}	Resting level	-1.00

A.5 Executable Tasks Layer (ETL)

τ^{etl}	Time constant of u^{etl}	2.00
A^{etl}	Amplitude of the kernel	2.00
α^{etl}	Spatial phase of the kernel	0.50
k^{etl}	Decay rate of the kernel	0.30
ς^{etl}_{stoch}	Stochastic noise	0.00
H^{etl}	Resting level	-1.00

A.6 Action Execution Layer (AEL)

τ^{ael}	Time constant of u^{ael}	30.00
A^{ael}	Amplitude of the kernel	1.00
σ^{ael}	Standard deviation of the gaussian curve	7.00

w_{inh}^{ael}	Global inhibition of the kernel	4.00
ς_{stoch}^{ael}	Stochastic noise	0.50
$C^{c\rightarrow ael}$	Gain parameter controlling the strength of contribution “action” to s^{ael}	5.00
$C^{a\rightarrow ael}$	Gain parameter controlling the strength of contribution “color” to s^{ael}	5.00
$C^{rec\rightarrow ael}$	Gain parameter controlling the strength of the Recall contribution to AEL	2.20
$\underset{pa\rightarrow ael}{I}$	Value of the inhibitory connection from u^{pa} to u^{ael}	8.00
H^{ael}	Resting level	-10.00

Appendix B

Implementation parameters - Chapter 9

B.1 Sequence Memory Field

τ^{SM}	Time constant of u^{SM}	1.00
A^{SM}	Amplitude of the kernel	1.50
α^{SM}	Spatial phase of the kernel	0.31
k^{SM}	Decay rate of the kernel	0.30
$C^{v\rightarrow SM}$	Gain parameter controlling the strength of the contribution from the Vision System to u^{SM}	7.00
h_0^{SM}	Neuronal resting level of $u^{SM}(x,t)$	-4.00
$\tau^{h^{SM}}$	Time constant of h^{SM}	50.00

B.2 Sequence Recall Field

τ^{SR}	Time constant of u^{SR}	1.00
A^{SR}	Amplitude of the kernel	1.50
α^{SR}	Spatial phase of the kernel	0.31
k^{SR}	Decay rate of the kernel	0.30
h^{SR}	Neuronal resting level of $u^{SR}(x,t)$	-20.00

$\tau^{h^{SR}}$	Time constant of h^{SR}	50.00

B.3 Past Events Field

τ^{PE}	Time constant of u^{PE}	40.00
A^{PE}	Amplitude of the kernel	10.00
α^{PE}	Spatial phase of the kernel	0.50
k^{PE}	Decay rate of the kernel	0.30
h^{PE}	Neuronal resting level of $u^{PE}(x,t)$	-2.00

Bibliography

Ackerman, E. (2019). A Robot That Explains Its Actions Is a First Step Towards AI We Can (Maybe) Trust - IEEE Spectrum. Retrieved January 10, 2020, from https://spectrum.ieee.org/automaton/robotics/artificial-intelligence/a-robot-that-explains-its-actions

Amari, S.-i. (1977). Dynamic of pattern formation in lateral-inhibition type neural fields. *Biological Cybernetics*, *27*, 77–87. https://doi.org/10.1007/BF00337259

Amit, R., & Mataric, M. (2002). Learning movement sequences from demonstration, In *Proceedings 2nd International Conference on Development and Learning. ICDL 2002*. https://doi.org/10.1109/DEVLRN.2002.1011867

Argall, B. D., Chernova, S., Veloso, M., & Browning, B. (2009). A survey of robot learning from demonstration. *Robotics and Autonomous Systems*, *57*(5), 469–483. https://doi.org/10.1016/j.robot.2008.10.024

Bandura, A. (1977). *Social Learning Theory*. Englewood Cliffs, NJ: Prentice Hall.

Bicho, E., Erlhagen, W., Sousa, E., Louro, L., Hipolito, N., Silva, E. C., Silva, R., Ferreira, F., MacHado, T., Hulstijn, M., Maas, Y., De Bruijn, E., Cuijpers, R. H., Newman-Norlund, R., Van Schie, H., Meulenbroek, R. G., & Bekkering, H. (2012). The power of prediction: Robots that read intentions, In *IEEE International Conference on Intelligent Robots and Systems*. https://doi.org/10.1109/IROS.2012.6386297

Bicho, E. (1999). *Dynamic Approach to Behavior-Based Robotics: Design, Specification, Analysis, Simulation and Implementation* (Doctoral dissertation). University of Minho. http://repositorium.sdum.uminho.pt/handle/1822/210

Bicho, E., Erlhagen, W., Louro, L., & Costa e Silva, E. (2011a). Neuro-cognitive mechanisms of decision making in joint action: A human-robot interaction study. *Human Movement Science*, *30*(5), 846–868. https://doi.org/10.1016/j.humov.2010.08.012

Bicho, E., Erlhagen, W., Louro, L., Costa e Silva, E., Silva, R., & Hipólito, N. (2011b). A dynamic field approach to goal inference, error detection and anticipatory action selection in human-robot collaboration. In *New Frontiers in Human-Robot Interaction (Advances in Interaction Studies)* (pp. 135–164). https://doi.org/10.1075/ais.2.10bic

Bicho, E., Louro, L., & Erlhagen, W. (2010). Integrating verbal and nonverbal communication in a dynamic neural field architecture for human-robot interaction. *Frontiers in Neurorobotics*, *4*. https://doi.org/10.3389/fnbot.2010.00005

Bicho, E., Louro, L., Hipólito, N., & Erlhagen, W. (2009). A dynamic field approach to goal inference and error monitoring for human-robot interaction, In *Proceedings of the International Symposium on New Frontiers in Human-Robot Interaction AISB (Adaptive & Emergent Behaviour & Complex Systems) Convention*. http://hdl.handle.net/1822/10306

Bicho, E., Mallet, P., & Schöner, G. (2000). Target Representation on an Autonomous Vehicle with Low-Level Sensors. *The International Journal of Robotics Research*, *19*(5), 424–447. https://doi.org/10.1177/02783640022066950

Billard, A., Calinon, S., Dillmann, R., & Schaal, S. (2008). Robot programming by demonstration. In *Springer Handbook of Robotics* (pp. 1371–1394). https://doi.org/10.1007/978-3-540-30301-5_60

Breazeal, C., Gray, J., & Berlin, M. (2009). An embodied cognition approach to mindreading skills for socially intelligent robots. *International Journal of Robotics Research*, *28*(5), 656–680. https://doi.org/10.1177/0278364909102796

Brown, G. D., Hulme, C., & Preece, T. (2000). Oscillator-Based Memory for Serial Order. *Psychological Review*, *107*(1), 127–181. https://doi.org/10.1037/0033-295X.107.1.127

Butterfield, J., Osentoski, S., Jay, G., & Jenkins, O. C. (2010). Learning from demonstration using a multi-valued function regressor for time-series data. *10th IEEE-RAS International Conference on Humanoid Robots*, 328–333. https://doi.org/10.1109/ICHR.2010.5686284

Cakmak, M., & Thomaz, A. L. (2012). Designing robot learners that ask good questions, In *Proceedings of the Seventh Annual ACM/IEEE International Conference on Human-Robot Interaction*. https://doi.org/10.1145/2157689.2157693

Calinon, S., Guenter, F., & Billard, A. (2007). On learning, representing, and generalizing a task in a humanoid robot. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, *37*(2), 286–298. https://doi.org/10.1109/TSMCB.2006.886952

Chang, G. (2013). *Robot Motion and Task Learning with Error Recovery*. University of Waterloo. https://uwspace.uwaterloo.ca/handle/10012/7627

Chen, K., Kvasnicka, V., Kanen, P. C., & Haykin, S. (2001). Neural and adaptive systems: Fundamentals through simulations. *IEEE Transactions on Neural Networks*, *12*(3), 648–649. https://doi.org/10.1109/TNN.2001.925574

Chernova, S., & Thomaz, A. L. (2014). Robot Learning from Human Teachers. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, *8*(3), 1–121. https://doi.org/10.2200/S00568ED1V01Y201402AIM028

Cunha, A., Ferreira, F., Erlhagen, W., Sousa, E., Louro, L., Vicente, P., Monteiro, S., & Bicho, E. (2020). Towards endowing collaborative robots with fast learning for minimizing tutors' demonstrations: What and when to do?, In *Robot 2019: Fourth Iberian Robotics Conference*. https://doi.org/10.1007/978-3-030-35990-4_30

Curtis, C. E., & Lee, D. (2010). Beyond working memory: The role of persistent activity in decision making. *Trends in Cognitive Sciences*, *14*(5), 216–222. https://doi.org/10.1016/j.tics.2010.03.006

Dautenhahn, K., Nehaniv, C. L., Walters, M. L., Robins, B., Kose-Bagci, H., Mirza, N. A., & Blow, M. (2009). KASPAR - a minimally expressive humanoid robot for human-robot interaction research. *Applied Bionics and Biomechanics*. https://doi.org/10.1080/11762320903123567

Diedrichsen, J., & Kornysheva, K. (2015). Motor skill learning between selection and execution. *Trends in Cognitive Sciences*, *19*(4), 227–233. https://doi.org/10.1016/j.tics.2015.02.003

Dillmann, R. (2004). Teaching and learning of robot tasks via observation of human performance. *Robotics and Autonomous Systems*, *47*(2-3), 109–116. https://doi.org/10.1016/j.robot.2004.03.005

Ekvall, S., & Kragic, D. (2006). Learning task models from multiple human demonstrations. *Proceedings - IEEE International Workshop on Robot and Human Interactive Communication*, 358–363. https://doi.org/10.1109/ROMAN.2006.314460

Engels, C., & Schöner, G. (1995). Dynamic fields endow behavior-based robots with representations. *Robotics and Autonomous Systems*, *14*(1), 55–77. https://doi.org/10.1016/0921-8890(94)00020-3

Erlhagen, W., Mukovskiy, A., Bicho, E., Panin, G., Kiss, C., Knoll, A., van Schie, H., & Bekkering, H. (2006). Goal-directed imitation for robots: A bio-inspired approach to action understanding and skill learning. *Robotics and Autonomous Systems*, *54*(5), 353–360. https://doi.org/10.1016/j.robot.2006.01.004

Erlhagen, W., & Bicho, E. (2006). The dynamic neural field approach to cognitive robotics. *Journal of Neural Engineering*, *3*(3), R36–R54. https://doi.org/10.1088/1741-2560/3/3/R02

Erlhagen, W., & Bicho, E. (2014). A dynamic neural field approach to natural and efficient human-robot collaboration. *Neural Fields: Theory and Applications*, 1–487. https://doi.org/10.1007/978-3-642-54593-1

Erlhagen, W., Mukovskiy, A., Bicho, E., Panin, G., Kiss, C., Knoll, A., Van Schie, H., & Bekkering, H. (2005). Action understanding and imitation learning in a robot-human task, In *Artificial Neural Networks: Biological Inspirations – ICANN*. https://doi.org/10.1007/11550822_42

Faubel, C., & Schöner, G. (2008). Learning to recognize objects on the fly: A neurally based dynamic field approach. *Neural Networks*, *21*(4), 562–576. https://doi.org/10.1016/j.neunet.2008.03.007

Ferreira, F. (2014). *Multi-bump solutions in dynamic neural fields : Analysis and Applications* (Doctoral dissertation). University of Minho.

Ferreira, F., Erlhagen, W., & Bicho, E. (2011). A dynamic field model of ordinal and timing properties of sequential events. https://doi.org/10.1007/978-3-642-21738-8_42

Ferreira, F., Erlhagen, W., & Bicho, E. (2016). Multi-bump solutions in a neural field model with external inputs. *Physica D: Nonlinear Phenomena*, *326*, 32–51. https://doi.org/10.1016/j.physd.2016.01.009

Ferreira, F., Erlhagen, W., Sousa, E., Louro, L., & Bicho, E. (2014). Learning a musical sequence by observation: A robotics implementation of a dynamic neural field model. *IEEE ICDL-EPIROB 2014 - 4th Joint IEEE International Conference on Development and Learning and on Epigenetic Robotics*, 157–162. https://doi.org/10.1109/DEVLRN.2014.6982973

Fong, T., Nourbakhsh, I., & Dautenhahn, K. (2003a). A Survey of Socially Interactive Robots: Concepts, Design, and Applications. *Robotics and Autonomous Systems*, *42*(3), 143–166. https://doi.org/10.1016/S0921-8890(02)00372-X

Fong, T., Thorpe, C., & Baur, C. (2003b). Collaboration, Dialogue, Human-Robot Interaction, In *Robotics Research*, Springer Berlin Heidelberg. https://doi.org/10.1007/3-540-36460-9_17

Garland, A., & Lesh, N. (2003). *Learning hierarchical task models by demonstration*. Technical Report TR2003-01, Mitsubishi Electric Research Laboratories. http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.69.1211

Hebb, D. O. (1949). *The organization of behavior: A neuropsychological theory*. New York, Wiley.

Henson, R. N. (1998). Short-Term Memory for Serial Order: The Start-End Model. *Cognitive Psychology*, *36*(2), 73–137. https://doi.org/10.1006/cogp.1998.0685

Houghton, G. (1990). The problem of serial order: A neural network model of sequence learning and recall. In *Current research in natural language generation* (pp. 287–319). USA, Academic Press Professional, Inc. https://dl.acm.org/doi/10.5555/103363.103622

International Federation of Robotics. (2018). *Robots and the Workplace of the Future*. https://ifr.org/papers/robots-and-the-workplace-of-the-future

Khamassi, M., Girard, B., Clodic, A., Sandra, D., Renaudo, E., Pacherie, E., Alami, R., & Chatila, R. (2016). Integration of action, joint action and learning in robotics cognitive architectures. *Intellectica. Revue de l'Association pour la Recherche Cognitive. Nouvelles approches en Robotique Cognitive.*, *65*. https://doi.org/10.3406/intel.2016.1794

Kim, M.-G., & Suzuki, K. (2010). A card playing humanoid for understanding socio-emotional interaction, In *Entertainment Computing - ICEC 2010*, Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-15399-0_2

Koenemann, J., Burget, F., & Bennewitz, M. (2014). Real-time imitation of human whole-body motions by humanoids, In *2014 IEEE International Conference on Robotics and Automation (ICRA)*. https://doi.org/10.1109/ICRA.2014.6907261

Konidaris, G., Kuindersma, S., Grupen, R., & Barto, A. (2012). Robot learning from demonstration by constructing skill trees. *International Journal of Robotics Research*, *31*(3), 360–375. https://doi.org/10.1177/0278364911428653

Krüger, J., Bernhardt, R., & Surdilovic, D. (2006). Intelligent assist systems for flexible assembly. *CIRP Annals - Manufacturing Technology*, *55*(1), 29–32. https://doi.org/10.1016/S0007-8506(07)60359-X

Kuniyoshi, Y., Inaba, M., & Inoue, H. (1994). Learning by watching: Extracting reusable task knowledge from visual observation of human performance. *IEEE Transactions on Robotics and Automation*, *10*(6), 799–822. https://doi.org/10.1109/70.338535

Kyrarini, M., Haseeb, M. A., Ristić-Durrant, D., & Gräser, A. (2018). Robot learning of industrial assembly task via human demonstrations. *Autonomous Robots*, *43*(1), 239–257. https://doi.org/10.1007/s10514-018-9725-6

Laing, C. R., Troy, W. C., Gutkin, B., & Ermentrout, G. B. (2002). Multiple Bumps in a Neuronal Model of Working Memory. *SIAM Journal on Applied Mathematics*. https://doi.org/10.1137/S0036139901389495

Lashley, K. (1951). The problem of serial order in behavior. *Cerebral mechanisms in behavior*, (7), 112–147.

Lee, J. (2017). A survey of robot learning from demonstrations for human-robot collaboration. https://arxiv.org/abs/1710.08789

Lek, S., & Park, Y. (2008). Artificial neural networks. In S. E. Jørgensen & B. D. Fath (Eds.), *Encyclopedia of ecology* (pp. 237–245). Oxford, Academic Press. https://doi.org/https://doi.org/10.1016/B978-008045405-4.00173-7

Lek, S., & Guégan, J. F. (1999). Artificial neural networks as a tool in ecological modelling, an introduction. *Ecological Modelling*, *120*(2-3), 65–73. https://doi.org/10.1016/S0304-3800(99)00092-7

Levas, A., & Selfridge, M. (1984). A user-friendly high-level robot teaching system, In *Proceedings. 1984 IEEE International Conference on Robotics and Automation*. https://doi.org/10.1109/ROBOT.1984.1087193

Lewandowsky, S., & Murdock, B. B. (1989). Memory for Serial Order. *Psychological Review*, *96*(1), 25–57. https://doi.org/10.1037/0033-295X.96.1.25

Lind, J., Ghirlanda, S., & Enquist, M. (2019). Social learning through associative processes: A computational theory. *Royal Society Open Science*, *6*(3). https://doi.org/10.1098/rsos.181777

Malheiro, T., Bicho, E., Machado, T., Louro, L., Monteiro, S., Vicente, P., & Erlhagen, W. (2017). A software framework for the implementation of dynamic neural field control architectures for human-robot interaction, In *2017 IEEE International Conference on Autonomous Robot Systems and Competitions, ICARSC 2017*. https://doi.org/10.1109/ICARSC.2017.7964067

Mcclelland, J., Mcnaughton, B., & O'Reilly, R. (1995). 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*, *102*, 419–57. https://doi.org/10.1037/0033-295X.102.3.419

McCulloch, W. S., & Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. *The Bulletin of Mathematical Biophysics*, *5*(4), 115–133. https://doi.org/10.1007/BF02478259

Misra, D. K., Sung, J., Lee, K., & Saxena, A. (2016). Tell me Dave: Context-sensitive grounding of natural language to manipulation instructions. *International Journal of Robotics Research*, *35*(1-3), 281–300. https://doi.org/10.1177/0278364915602060

Miura, J., Iwase, K., & Shirai, Y. (2005). Interactive teaching of a mobile robot. *Proceedings of the IEEE International Conference on Robotics and Automation*, 3378–3383. https://doi.org/10.1109/ROBOT.2005.1570632

Monteiro, S., Vaz, M., & Bicho, E. (2004). Attractor dynamics generates robot formations: From theory to implementation. *IEEE International Conference on Robotics and Automation*, *3*, 2582–2586. https://doi.org/10.1109/ROBOT.2004.1307450

Murdock, B. B. (1995). Developing TODAM: Three models for serial-order information. *Memory & Cognition*, *23*(5), 631–645. https://doi.org/10.3758/BF03197264

Nicolescu, M. N., & Mataric, M. J. (2003). Natural Methods for Robot Task Learning: Instructive Demonstrations, Generalization and Practice. *Proceedings of the International Conference on Autonomous Agents*, *2*(May 2014), 241–248. https://doi.org/10.1145/860575.860614

Niekum, S., Osentoski, S., Konidaris, G., & Barto, A. G. (2012). Learning and generalization of complex tasks from unstructured demonstrations, In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, IEEE. https://doi.org/10.1109/IROS.2012.6386006

Ollis, M., Huang, W. H., & Happold, M. (2007). A bayesian approach to imitation learning for robot navigation, In *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*. https://doi.org/10.1109/IROS.2007.4399220

Page, M. P. A., & Norris, D. (1998). The Primacy Model: A New Model of Immediate Serial Recall. *Psychological Review*, *105*(4), 761–781. https://doi.org/10.1037/0033-295X.105.4.761-781

Pardowitz, M., Knoop, S., Dillmann, R., & Zollner, R. D. (2007). Incremental learning of tasks from user demonstrations, past experiences, and vocal comments. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, *37*(2), 322–332. https://doi.org/10.1109/TSMCB.2006.886951

Rethink Robotics. (2018). Sawyer collaborative robot. Retrieved July 2, 2019, from http://www.rethinkrobotics.com/sawyer/

Salem, M., Eyssel, F., Rohlfing, K., Kopp, S., & Joublin, F. (2013). To err is human(-like): Effects of robot gesture on perceived anthropomorphism and likability. *International Journal of Social Robotics*, *5*(3), 313–323. https://doi.org/10.1007/s12369-013-0196-9

Sammut, C., & Webb, G. I. (Eds.). (2017). Unsupervised learning. In *Encyclopedia of Machine Learning and Data Mining* (pp. 1304–1304). Boston, MA, Springer US. https://doi.org/10.1007/978-1-4899-7687-1_976

Sandamirskaya, Y. (2014). Dynamic neural fields as a step toward cognitive neuromorphic architectures. *Frontiers in Neuroscience*, *7*(8 JAN), 1–13. https://doi.org/10.3389/fnins.2013.00276

Sandamirskaya, Y., & Schöner, G. (2010). An embodied account of serial order: How instabilities drive sequence generation. *Neural Networks*, *23*(10), 1164–1179. https://doi.org/10.1016/j.neunet.2010.07.012

Sandry, E. (2015). Re-evaluating the form and communication of social robots. *International Journal of Social Robotics*, *7*(3), 335–346. https://doi.org/10.1007/s12369-014-0278-3

Sato, Y., Bernardin, K., Kimura, H., & Ikeuchi, K. (2002). Task analysis based on observing hands and objects by vision, In *IEEE/RSJ International Conference on Intelligent Robots and Systems*. https://doi.org/10.1109/IRDS.2002.1043898

Scholtz, J. (2003). Theory and evaluation of human robot interactions, In *Proceedings of the 36th Annual Hawaii International Conference on System Sciences*. https://doi.org/10.1109/HICSS.2003.1174284

Schöner, G., Dose, M., & Engels, C. (1995). Dynamics of behavior: Theory and applications for autonomous robot architectures. *Robotics and Autonomous Systems*, *16*(2), 213–245. https://doi.org/10.1016/0921-8890(95)00049-6

Seitz, A. R., & Dinse, H. R. (2007). A common framework for perceptual learning. *Current Opinion in Neurobiology*, *17*(2), 148–153. https://doi.org/10.1016/j.conb.2007.02.004

Sheridan, T. B. (1997). Eight ultimate challenges of human-robot communication, In *Proceedings 6th IEEE International Workshop on Robot and Human Communication*. https://doi.org/10.1109/ROMAN.1997.646944

Sheridan, T. B. (2016). Human–robot interaction: Status and challenges. *Human Factors*, *58*(4), 525–532. https://doi.org/10.1177/0018720816644364

Silva, R. (2008). Design e construção de um robot antropomórfico. http://hdl.handle.net/1822/9102

Sousa, E. (2014). *On Learning and Generalizing Representations in a Dynamic Field Based Architecture for Human-Robot Interaction* (Doctoral dissertation). University of Minho. http://repositorium.sdum.uminho.pt/handle/1822/36642

Sousa, E., Bicho, E., & Erlhagen, W. (2009). On the learning of inter-field connections in a dynamic field architecture for human-robot collaboration. *9th Conference on Autonomous Robot Systems and Competitions*.

Sousa, E., Erlhagen, W., & Bicho, E. (2014). On observational learning of hierarchies in sequential tasks: A dynamic neural field model. In *Computational Models of Cognitive Processes* (pp. 196–210). https://doi.org/10.1142/9789814458849_0015

Sousa, E., Erlhagen, W., Ferreira, F., & Bicho, E. (2015). Off-line simulation inspires insight: A neurodynamics approach to efficient robot task learning. *Neural Networks*, *72*, 123–139. https://doi.org/10.1016/j.neunet.2015.09.002

Sutton, R. S., & Barto, A. G. (1998). *Introduction to reinforcement learning* (1st). Cambridge, MA, USA, MIT Press.

Veeraraghavan, H., & Veloso, M. (2008). Teaching sequential tasks with repetition through demonstration. *Proceedings of the 7th international joint conference on Autonomous agents and multiagent systemsVolume 3*, *3*(Aamas), 1357–1360. http://portal.acm.org/citation.cfm?id=1402821.1402871

Widrow, B., & Hoff, M. E. (1988). Adaptive switching circuits. In *Neurocomputing: Foundations of research* (pp. 123–134). Cambridge, MA, USA, MIT Press. https://dl.acm.org/doi/10.5555/103363.103622

Wiese, E., Metta, G., & Wykowska, A. (2017). Robots as intentional agents: Using neuroscientific methods to make robots appear more social. *Frontiers in Psychology*, *8*, 1663. https://doi.org/10.3389/fpsyg.2017.01663

Wise, M. (2018). The 5 stages of acceptance as robots enter the workforce - World Economic Forum. Retrieved January 14, 2020, from https://www.weforum.org/agenda/2018/10/robots-are-coming-to-your-workplace-here-s-how-to-get-along-with-them/

Wojtak, W., Ferreira, F., Louro, L., Bicho, E., & Erlhagen, W. (2018). Towards temporal cognition for robots: A neurodynamics approach. *7th Joint IEEE International Conference on Development and Learning and on Epigenetic Robotics, ICDL-EpiRob 2017*, 407–412. https://doi.org/10.1109/DEVLRN.2017.8329836

Zador, A. M. (2019). A critique of pure learning and what artificial neural networks can learn from animal brains. *Nature Communications*, *10*(1). https://doi.org/10.1038/s41467-019-11786-6

Zhao, L., Lawhorn, R., Patil, S., Susanibar, S., Lu, L., Wang, C., & Ouyang, B. (2017). *Multiform adaptive robot skill learning from humans*, Vol. 1.*August*. https://doi.org/10.1115/DSCC2017-5114