High Efficiency Real-Time Sensor and Actuator Control and Data Processing

Marcello Ferro

April 13, 2006

Ph. D. Thesis - Department of Electric Systems and Automation

Interdepartmental Research Center “E. Piaggio”

Faculty of Engineering

University of Pisa, Italy

Doctorate title:	**Automatics, robotics and bioengineering**
Macroarea:	**Bioengineering**
Cycle:	**XVIII**
Scientific code:	**Area 09, ING-INF/06**
Doctorate starting date:	**January 1^{st} 2003**
Title:	**High efficiency real-time sensor and actuator control and data processing**

Tutor
Prof. Danilo De Rossi

Student
Marcello Ferro

Contents

1 Introduction **6**

1.1 Multi-Transducer Control 6

1.2 The End-User 8

1.3 A Framework Solution for High Complex Tasks 10

2 Facial Automaton for Conveying Emotion **13**

2.1 Introduction 13

2.2 F.A.C.E., the facial automaton 16

2.2.1 The anthropomorphic head 16

2.2.2 The anthropomorphic control 18

2.2.3 The artificial decision unit 19

2.2.4 Kinaesthetic proprioception: biomimetic facial skin as a redundant self-sensory apparel for the system 21

2.2.5 Human muscles and artificial actuators: borrowing from anthropomorphic mechanics 24

2.2.6 Biomimetic actuation using Feldmans muscle model . . . 25

2.2.7 Materials for actuation 26

2.3 Future directions 28

3 Framework Core: Control Architecture **32**

3.1 Modular, Reusable and Object-Oriented Framework 32

3.2 I/O Device Communication 34
3.3 Parallel Distributed Processing 34
3.4 Development of Control and Processing Modules 36
3.5 Framework Overview 37
3.5.1 Portability 39
3.5.2 Containers 40
3.5.3 The base structure: 3DObject 40
3.5.4 The object manager: 3DWorld 44
3.5.5 The base process structure: W 45
3.5.6 The connection between two processes: WConnection and WConnectionSpec 46
3.5.7 Grouping processes: WGroup and WProjectionSpec . . . 48
3.5.8 The interface with sensor and actuator systems: SensorDriver and ActuatorDriver 49
3.5.9 General Purpose Instruments 50
3.5.9.1 Data input from file: FileSensorDriver 51
3.5.9.2 Data input from camera: CameraSensorDriver and face-tracking 52
3.5.10 F.A.C.E. instruments 55
3.5.10.1 Facial Expression Estimator: FEECameraSensorDriver 55
3.5.10.2 Acquisition Boards 56
3.5.10.3 Proprioceptive System Overview: An Artificial Sensing Skin 57

4 Framework Extensions: ANN 64

4.1 The integrate and fire model implementation: IFNeuron, IFNeuronConnectionSpec and IFNeuronGroup 69
4.1.1 Multi layer perceptron (MLP) 71
4.1.1.1 MLP learning 72
4.1.1.2 MLP implementation into the framework: IFMLP 78

4.1.2 Self Organizing Map (SOM) 80
4.1.2.1 Kohonen Self Organizing Map (KSOM) 82
4.1.3 Case Study: Real-Time Facial Expression Recognition . . 82
4.1.3.1 Introduction 83
4.1.3.2 Facial expression estimator 85
4.1.3.3 Processing: The hierarchical neural network . . 87
4.1.3.4 Results and discussion 89
4.1.3.5 Conclusions 95
4.1.3.6 Real-time facial expression recognition implementation overview 96
4.1.4 Case Study: Facial Automaton for Conveying Emotions as a Tool for autism (F.A.C.E.T.) 99
4.1.4.1 Introduction 101
4.1.4.2 F.A.C.E.T. 104
4.1.4.3 The therapeutic protocol for autism 105
4.1.4.4 Preliminary Results 109
4.1.4.5 Conclusion 111
4.2 Leabra Neuron Model 112
4.2.1 Leabra Neuron Model Implementation 114
4.2.2 Associative Memory: Hippocampus Simulation 114
4.2.2.1 Pattern Separation and Completion 115
4.2.2.2 Paired-Associate Learning (AB-AC Test) 117
4.2.3 Case Study: An Electronic Nose for Olive Oil Classification117
4.3 An Artificial Neuron Model for Real-Time Data Processing . . . 121
4.3.1 The Artificial Neuron Model Proposed by Izhikevich . . . 123
4.3.1.1 Selection of Neuronal Groups 125
4.3.1.2 Polychronization 126
4.3.1.3 Spike-Timing-Dependent Plasticity (STDP) . . . 126
4.3.2 Implementation of the Izhikevich artificial neuron model in the framework 127
4.3.3 Simulation of a Minimal Group 133

5 Conclusions **137**

Chapter 1

Introduction

1.1 Multi-Transducer Control

The advances in sensor and actuator technology foster the use of large multi-transducer networks in many different fields. The increasing complexity of multi-transducer architectures requires high efficiency interconnection and co-operation of several control and processing modules. Recently a first step toward a standard design of multi-transducer communication protocols and interfaces have been defined in the IEEE 1451 [25, 26, 27]. Enhancing the reliability of high-level processing systems represents the next critical step. Multi-transducer network modules often include tasks such as control, data acquisition, data filtering interfaces, feature selection and pattern analysis [28, 29, 30]. Heterogeneous techniques derived from chemometrics, neural networks, fuzzy-rules used to implement such tasks may introduce module interconnection and cooperation issues. It may not be reliable to establish a multi-channel communication among common artificial neural networks tools, feature extraction and selection processes, and acquisition and control systems [31, 32, 33]. Moreover, high level interfaces often do not allow adapting of the architecture and/or the processes topology at run-time. As a result complex processing methods have to be designed.

Research focuses on developing multivariate data analysis techniques to elucidate information from the responses of sensors and analytical instruments. These methods include neural networks, genetic algorithms, signal processing, and multivariate statistical methods as applied to pattern recognition and multivariate calibration. During off-line data analysis different tools may be used in order to perform signal preprocessing, dimensionality reduction, classification and clustering. In a signal preprocessing stage various purpouses are served, including baseline manipulation, compression, normalization and drift compensation. After, data are sent to a dimensionality reduction module in order to perform a feature extraction. Selected features are ready for classification and clustering tasks. Each module may start when the previous module has completed its task and data have been stored.

On the other hand, a real-time approach for data analysis requires the realization of interconnected modules which are capable to establish an efficient communication channel. In this way the application should be able to control all modules of the elaboration chain, including analysis protocol management and sensory and actuating interfaces.

To help dealing with these problems an architecture for a dynamic and efficient management of multi-transducer data processing techniques is presented. The author proposes an homogeneous software framework able to manage at the same time transducer devices and data processing. The framework is realised as a software library in order to exploit the potentials of the computational algorithms and to enhance the performances of neural processing techniques. Synchronization among modules and data flow is managed by the framework offering remarkable advantages in simulation of heterogeneous complex dynamic processes. Specific control processes, pattern recognition algorithms, sensory and actuating interfaces may be created inheriting from the framework base structures. The architecture is library-oriented rather than application-oriented and starting from the base models available in the framework core dedicated models for processes, maps and connections may be derived. Such a strategy permits the realisation of a user-defined environment able to automate the elaboration of cooperating processes. Etherogenous processes will be able to communicate each other inside the framework as specified by the user.

The framework architecture has been designed as a hierarchical structure whose root is a manager module. It is realised as an high-level container of generic modules and it represents the environment in which process modules and I/O filtering interfaces are placed. Modules may be grouped recursively in order to share common properties and functionalities of entity modules belonging to the same type. Communication channels are realised as connections through specific projection types that specify the connection topology. Connections are delegated to dispatch synchronization information and user-defined data. The filtering interface modules are able to drive the transducer hardware and to dispatch information to process modules. All base modules manage dynamic structures and are designed to maintain data consistency while the environment state may change. High level processes such as control processes and pattern recognition algorithms are defined as application processes inside the framework. Such processes inherit properties and functionalities from the framework base structures, taking advantage of automation capabilities provided by the framework core. The framework allows to create a communication language between the framework core and the hardware architecture. This guarantees an increased flexibility thanks to the presence of interfaces performing the function of interpreters for the specific hardware and filters which specify the way the framework core senses and communicates the information.

1.2 The End-User

From an ideal point of view, the best and most efficient solution for a generic system consists in the realisation of a dedicated hardware which is specialised in the specified task. However, during the design of a new system, a simulation step is preferred to the immediate realisation of a dedicated hardware. This obvious step is necessary to limit the cost of the design for both expensiveness and time development. Usually the scientific researcher uses different tools to perform data acquisition, control and analysis. Such tools, as introduced above, have etherogeneous structures since they are developed as stand-alone applications. Even the most useful ones permit the data export and import to and from compatible data formats, the realisation of a complete and easy-to-use instrument is often prohibitive. This limit is the price the researcher has to pay in order to use a tool without necessarily being an expert in the informatics discipline. In the case of small tasks and expecially when off-line analysis is sufficient to perform each sub-task, general purpouse tools may be used to address each specific problem. However, as the complexity of the problem grows and several cooperating modules are needed, more complex tools have to be adopted.

Since the nature of both transducers and data processing modules cannot be known without having specified the task, a modular structure is needed. Such a structure must be able to permit the use of new user-defined modules. Actually many tools satisfy this request and, among others, the most known are MatLab, LabView, SimulLink and DataEngine. Each of these may include several modules which may be realised or modified using different programming languages and techniques. However all of these tools may not gain the maximum of the computational efficiency of the calculator where they run. This is related to their nature, since these tools do not allow a direct access to the low level system calls of the operative system. The approach used by all these tools underlyes the necessity to use the calculator as an instrument to develop data acquisition and analysis modules using predefined applications which are written by expert software programmers. A different choice constists in the use of the calculator as a platform for the development of home-maden applications. This approach complicates the life of the researcher since he is asked to write the source code of the entire application. Using modern calculators equipped with actual operative systems and high level tools, this effort is justifiable only if the previous described choice is not applicable to the specific problem.

The author here proposes a software library architecture as a compromise between the two choices. Despite this approach requires a computational experience of the end-user, it gives to the developer the opportunity to use a low level layer wich is able to perform all the basic tasks. Moreover, the design of an object-oriented framework illustrates to the user the right protocol that must be used in order to realize new modules.

1.3 A Framework Solution for High Complex Tasks

During the last three years, the author has been involved in an ambitious project for the development of an automaton as a man-machine interface for the non-verbal communication. The robot is called Facial Automaton for Conveying Emotions (FACE) and it will be discussed in chapter 2. Currently it is being developed at the Interdepartmental Research Center "E. Piaggio" of the Faculty of Engineering of the University of Pisa by the bioengineering group lead by the Prof. D. De Rossi [56, 57, 58]. From a general point of view and without loosing commonness, FACE can be thought as a complex system where different modules have to cooperate in order to perform data acquisition from multiple sensors, data analysis through several techniques and data redirection to the actuator systems. Inside this project, my personal contribute lies in the control architecture design and in the realisation of control systems for both transducers and data analysis modules. Until the project will be terminated, the simulation with the help of a calculator is the only way to permit the test of all of the FACE's modules. Because of the generality of the problem, FACE has been used as the test field for the framework control solution here proposed. Such an architecture addresses at least three main issues, as it will be discussed in chapter 3:

- acquisition from sensors: a protocol interface will be available to dispatch data coming from input systems; for each hardware sensory system the user will realize the software driver to filter the signals and to dispatch data to the framework core via the framework I/O interface.

- data processing: inside the framework core all the processes will be specified by the user; for each process the user will specify the algorithms, the connection topology between other processes and, optionally, the geometrical structure.

- actuator driving: a protocol interface will be available to dispatch data from the framework core; for each hardware actuating system the user will realize the software driver to filter the data and to dispatch the signals from the framework I/O interface to the actuating systems.

The framework is designed to be versatile and it does not include any specification about the domain of the data flowing from input systems and to the output systems. However many modules have been developed as framework extensions in order to perform standard filtering for input systems, such as reading data from files (*FileSensorDriver*), reading data from USB and FireWire cameras (*CameraSensorDriver*), reading audio data (*MicSensorDriver*), reading data from USB and National Instruments acquisition boards (*UsbSensorDriver* and *NISensorDriver*). Moreover, many data processign tools have been developed as extensions, such as Principal Component Analysis (*PCA*), Linear Discriminant Analysis (*LDA*), Fast Fourier Transform (*FFT*). To address the control of the specific input systems included in the FACE robot other modules have been developed for face-tracking, eye-tracking, data input from proprioceptive systems and data output to servo-motor control. To improve the application portability over different platforms, an acquisition and driving board based on the USB protocol has been developed as external hardware and an appropriate sensory interface has been included in the framework as an extension. This choice permitted to avoid the use of expensive general purpouse I/O boards.

A lot of attention has been paid to the developement on neural network tools, as it will be described in chapter 4. Several neuronal models have been developed and the computational efficiency has been optimised to speed-up real-time applications. Demonstration applications have been developed to test both the architecture and the right execution of the FACE's tasks. A Facial Expression Estimator (FEE) has been realised using the Integrate and Fire artificial neuronal model. The leabra neuronal model has been used to realise a classification module based on a model of the hippocampus working memory. Finally the Izhikevich neuronal model has been implemented to develop an efficient neural network which is able to run in real-time mode and to perform a time-space correlation on input signals. The Izhikevich model for the neuron entity, which is based on the Edelman theory of neuronal groups selection, is currently under study.

Chapter 2

Facial Automaton for Conveying Emotion

The human face is the main organ of expression, capable of transmitting emotions that are almost instantly recognised by fellow beings. The development of a lifelike facial display based on the principles of biomimetic engineering is described in this chapter. A number of paradigms that can be used for developing believable emotional displays, borrowing from elements of anthropomorphic mechanics and control, and materials science, are outlined. These are used to lay down the technological and philosophical premises necessary to construct a man-machine interface for expressing emotions through a biomimetic mechanical head. Applications in therapy to enhance social skills and understanding emotion in people with autism are discussed.

2.1 Introduction

In recent years, the non-verbal expression of emotional states has rapidly become an area of interest in computer science and robotics. These developments parallel with important streams of thought in other fields, such as neurosciences, cognitive science, biology, developmental psychology and ethology. A great deal of research has been carried out in applications such as robotics and human-computer interaction (HCI), where the power of facial emotional expressions in human-human and human-machine communication has been recognised. Several systems capable of automatic analysis, interpretation and categorisation of some basic human facial expressions have been developed [61, 50, 48].

The other side of HCI involves the generation of emotion and expression in artificial systems. These can be divided into software-based systems and real three-dimensional (3D) artifacts, otherwise known as biologically inspired robots or sociable agents. Most emphasis has been given to the former; in fact many robotics researchers are investigating the emulation of human expression on computer-generated images. On the other hand, very little has been obtained toward the realisation of a 3D expressive face. Nevertheless, pioneering research work on the subject of sociable agents is being carried out at MIT by Breazeal and Scassellati [4]. They have designed a robot, known as KISMET, to interact socially with human parents. It integrates perception, attention, drives, basic emotions, behaviour selection and motor action. In this architecture the biomimetic aspects of emotional expression are not in the foreground and are at the moment reduced to meaningful but cartoon-like elements. A different but nonetheless interesting approach to the creation of agents emotionally and socially meaningful to humans is research aimed at endowing robots with more naturalistic mimic expressions, exploiting structures and actuators that can more closely replicate the subtleties of human movement. At the Science University of Tokyo, an android face with an elastomer skin overlying the mechanical skull produces a range of expressions using strategically positioned shape memory alloy actuators. The elastomer skin is shaped, tinted and made up to resemble the face of a woman, but a smile takes a rather long time to fade because of the long relaxation times of the alloys [16]. For the moment, the applications of these research-oriented systems are limited to exploring social behaviour and emotion in man-made systems, with the ultimate aim of

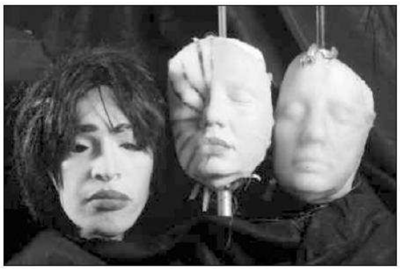


Figure 2.1: Three different phases of the development of F.A.C.E.

improving man-machine interfaces.

Facial androids abound in the entertainment industry [24] and have now achieved a high degree of believability, at least on celluloid. These systems have a remarkable aesthetic quality when they have been adequately retouched by lighting and graphic artists and are actuated by slow-moving motors and cables attached to an inert silicone skin. Seen in real life they are about as believable as a mannequin.

In summary, although some researchers and many artists are investigating the emulation of human expression and the replication of its aesthetic qualities, little attempt has been made toward the realisation of a 3D expressive face with truly lifelike or believable qualities.

For the past six years we at the University of Pisa have been involved in an ambitious project to develop a believable facial display system based on biomimetic engineering principles. The system is called FACE: facial automaton for conveying emotions (Fig. 2.1).

The underlying philosophy and design approach of the display is founded on the simulation of biological behaviour using materials, structures and control algorithms that can replicate some of the functions and responses of living systems. The long-term aim of the project is far reaching and culminates in achieving true believability, visually as well as in terms of texture and motion, in synthetic structures with human-like forms. At present, the immediate ob-

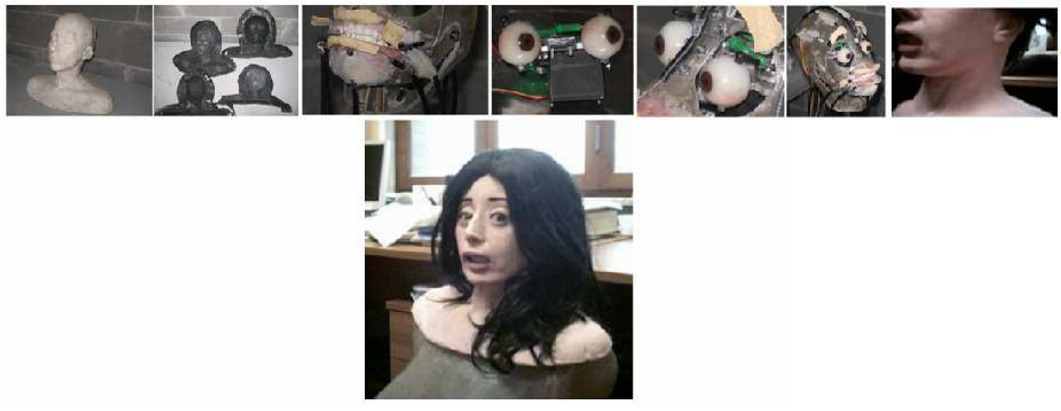


Figure 2.2: Facial Automaton For Conveying Emotions (FACE)

jective is focused upon simulating emotion in a 3D lifelike display (Fig. 2.2) and exploring its use in social skills and emotional therapy in individuals with autism.

2.2 F.A.C.E., the facial automaton

The aim of the FACE project is to realise a mechanical clone of a human head. The architecture of the facial automaton can be divided into three main sections (Figure 2.3): an anthropomorphic head (AH), an anthropomorphic control (AC) system and an artificial decision unit (ADU).

2.2.1 The anthropomorphic head

To obtain an artificial anthropomorphic head that embodies passive facial muscular and skeletal (skull) structures we start from a real subject and then attempt to reconstruct the hard and soft tissues of the selected human being. A copy of the skull is obtained starting from CAT (computerised axial tomography) data and by means of appropriate software of volumetric virtual reconstruction [18]. A segmentation process to isolate the skull is applied on the data, and the skull can be physically reconstructed in resin by means of CAD/CAM techniques. Reconstruction of soft tissues, on the other hand, is more difficult. In fact, were we to use a similar process, nuclear magnet resonance (NMR) is necessary. Starting from this data, each muscle has to be segmented and reconstructed; it is a long and complex process because of the difficulties of isolating muscles from images. For this reason, we have adopted a methodology known in the field of anthropology as facial reconstruction [84]. The technique involves the manual reconstruction of the facial muscular structure on the basis of tables indicating the thickness of soft tissues in different fixed points of the skull. The main facial muscles were separately reconstructed by using soft materials. This architecture was then coated by an artificial skin.

The artificial skin of FACE is a thin silicone-based mask equipped with a distributed kinaesthetic proprioception (KP) system, and coats the anthropomorphic head. It is fabricated by means of life-casting techniques and aesthetically represents a copy of the skin of the subject, both in shape and texture. Prosthetic grade alginate, plaster and plaster bandages are used to make casts of exceptional quality. Once the life cast is realised, it is covered with mouldable silicone rubber. This allows a silicone-based mould of a models face to be obtained. The silicone-based mould is supported by applying plaster around it. To mimic the supple flexibility of human skin, the mask of FACE is cast by filling the mould with a liquid silicone elastomer, which, once hardened, resembles the natural visco-elasticity of human tissue. Liquid silicone is mixed with pigments to emulate a medium level of melanin and to reproduce different shades on the face. An artificial muscle architecture is inserted into the passive muscular structure within the AH. In particular, the artificial muscle architecture, which we call the anthropomorphic mechanics (AM), is connected between the skin and the skull (Fig. 2.3).

Social nonverbal communication is largely dependent on the ability of the eyes to express emotions and track subjects. These actions are performed by the

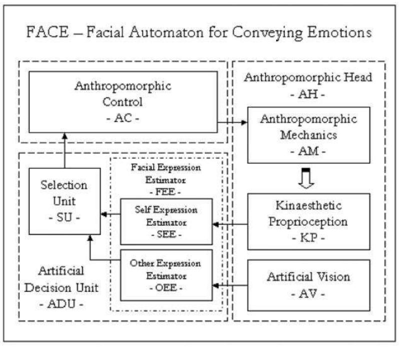

FACE – Facial Automaton for Conveying Emotions
Anthropomorphic Control - AC -
Anthropomorphic Head - AH -
Anthropomorphic Mechanics - AM -
Selection Unit - SU -
Facial Expression Estimator - FEE -
Self Expression Estimator - SEE -
Kinaesthetic Proprioception - KP -
Other Expression Estimator - OEE -
Artificial Decision Unit - ADU -
Artificial Vision - AV -

Figure 2.3: The architecture of F.A.C.E.

complex eyeball muscle system. An important feature of the AH is the artificial eyeballs and eyelids, which replicate the eye colours and shape of the selected subject. A system consisting of linear actuators made of dielectric elastomers designed to mimic the architecture of the human eye is under development. In particular, actuators are connected to its eyeballs and aimed at reproducing actions exerted by the eyelids and by the four main muscles of the human eyeball: the superior rectus, inferior rectus, lateral rectus and medial rectus.

2.2.2 The anthropomorphic control

Humans can easily express a mime and communicate their feelings and emotions. These tasks are extremely complex for a biomimetic android head, just as body movements are complex for a biomimetic robot. Due to the high dimensionality of the configuration space and redundancy of the system, such devices are very hard objects from the control point of view. One of the most evident is the fact that the process of rationalisation and simplification in the design of artificial moving parts leads to the choice of the smallest possible number of controls. For this reason, most research on humanoid control draws inspiration and models from biology, in particular from neuroscience of motor control and coordination

[5, 22, 15, 21]. Anyway, the biological paradigm is completely different in the sense that the number of muscles and tendons is much higher than the actual degrees of freedom of, for instance, a limb. Indeed, in human beings, routine tasks are carried out by an almost subconscious involuntary control, since we make use of sensory motor maps that have already been learned. Were we to reprogram these maps each time we perform a task, it would be a lot more difficult to learn complex motor actions. In accordance with this biological paradigm, a class of nonlinear controls along the lines of Feldman's [85, 86] model for muscle control have been developed [18]. Although this approach diverges from the framework of classic control theory, it may lead to motion that is more believable than that of a traditional robotic limb. This type of control scheme will be implemented in FACE.

2.2.3 The artificial decision unit

FACE is equipped with an artificial decision unit (ADU) which includes a facial expression estimator (FEE) system based on an artificial vision device and a selection unit (SU) based on imitation paradigms. The use of imitation is fundamental for a biomimetic and behaviour-based approach to humanoid control and learning. Imitation involves the classification of visual input for recognition of human behaviour and allows the actuating control system to perform general movements and imitation learning. The ADU communicates with the AC and manages the expressivity of FACE. The SU will be based on a neural network system and will make decisions based on previous learning experiences to generate the output desired to bestow FACE with its final expression. The FEE is divided into two sections: a self expression estimator (SEE) and an other expression estimator (OEE). The OEE receives its input by means of an artificial vision (AV) device already developed at our laboratory, which provides FACE with the ability to recognise a number of expressions. The device includes an attention system and a model for facial expression recognition. The SEE, on the other hand, is based on the processing of data obtained from the sensorised artificial skin, as described in the next section.

Recently, automatic face recognition by means of computer vision has become a popular application field, resulting in commercial products such as FaceIt, designed at the Computational Neuroscience Laboratory at Rockefeller University [14]. Nevertheless, the problem of automatic recognition of facial expressions still remains a challenging topic of application-oriented research work [17].

Ekmans studies to construct a system for the detailed description of emotional expressions, the facial action coding system (FACS), were important sources used by computer science and robotic researchers to explore the field of emotional expression and communication [47]. FACS measures the movements of forty muscles in the face, classifying them as action units (AUs). One AU represents a movement that a group of muscles make to obtain a particular expression. Thus, a facial expression can be captured by an artificial vision system, or simply a camera, and by means of a suitable algorithm, divided into its component AUs and subsequently identified by FACS coding.

Waters and Terzopoulos [61] have developed techniques for facial expression recognition; in particular, a model of the skin and facial muscles based on FACS. Images acquired by the artificial vision system are converted into 2D potential functions whose local minima correspond to salient facial features (snakes). The snakes are tracked through each image at each time step. With the aim of representing muscle actions that lead to a skin deformation that matches with the image analysed, a 3D model of the face is deformed to approximate the AUs involved. By using this procedure, a facial expression can be identified.

Our approach uses a different technique; it is based on the AV system, realised using a CCD camera and a fringe pattern analysis [91]. A curvature map and a 3D mesh of the head of a subject are calculated from images acquired with the camera. A dedicated process detects a number of points (markers), which are used to divide the human face into main zones (i.e. eyes, front, nose, mouth and chin). Data of each area are processed by a hierarchical neural-network architecture based on self-organising maps (SOM) and error backpropagation

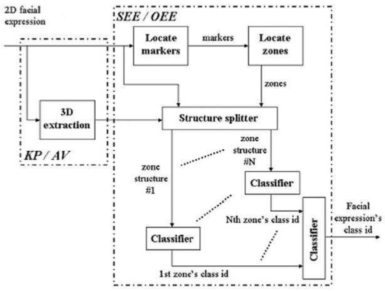


Figure 2.4: Architecture of the Facial Expression Estimator (FEE).

(EBP). An output classifier is used to classify the facial expression of the subject. Fig. 2.4 shows a block scheme of the system.

2.2.4 Kinaesthetic proprioception: biomimetic facial skin as a redundant self-sensory apparel for the system

The skin not only bestows expressivity and appearance but also sensing. Sensors in the skin are widely distributed and provide both tactile and kinaesthetic information. In an artificial sensing skin, the role of distributed sensors should not be to give an extremely accurate individual position of each element constituting FACE, but to enable a representation of the overall shape taken by the display. The skin of FACE is a complex 2D polymeric structure, and its response to simultaneous deformations in different directions is not easily reduced to a mathematical description. Thus, in this context, our biomimetic design is based on providing the artificial skin with a sort of proprioceptive mapping to approach the final required expression by a process of supervised learning. The supervised learning protocol leads FACE through a trial-and-error procedure until the system converges to a desired expression. Without sensing, the control of facial expression is open loop and hence unacceptable.

The artificial silicone skin of FACE is deformable, so the sensors have to be deformable too. Problems relating to the monitoring of the kinematics of the human body have been widely studied [11], and devices able to detect body movements by means of wearable distributed sensing fabrics have been developed [12]. For the realisation of a poroelastic biomimetic skin with embedded sensing fibres, two types of piezoresistive weavable fibres are being developed. First, filled rubber-coated threads (carbon-filled rubbers, CFR), which are sensitive to slowly varying deformations: they can, for example, map the steady state expression or the mood [23]. In contrast, conducting polymer fibres form the second class and are capable of mapping rapidly changing movements corresponding to immediate expressions deriving, for instance, from temper. Polypyrrole (PPy), a p-electron conjugated conducting polymer, which combines good properties of elasticity with mechanical and thermal transduction, is particularly suitable for this application. PPy-coated Lycra fabrics have been prepared using the method reported by De Rossi et al [11]. Sensors based on CFR are realised either by directly printing the carbon/rubber mixture onto fabrics or by weaving CFR-coated fibres.

The sensors have been characterised in terms of their quasi-static and dynamic electromechanical transduction properties, and the thermal and ageing properties of the sensing fabrics have also been assessed [10].

From a technical viewpoint, a piezoresistive woven sensing fabric is a tissue whose local resistivity is a function of the local strain. In a discrete way, it can be thought of as a 2D resistive network where single resistors have a nonlinear characteristic that depends on the local strain. To know the exact deformation of the artificial skin, the resistance of every single resistor, i.e. of every single tissue element, could be measured. However, this would place too much importance on the behaviour of every single resistor and would require an exceedingly large number of electrical paths to be taken out of the system. It would also mean forcing the system into the common mental scheme of the Euclidean spatial representation of objects, which may not be the most convenient in this context.

For these reasons, we measure the impedance only between points located at the borders of the tissue. The integral impedance pattern, then, is a function

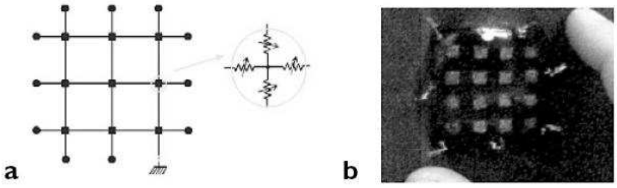


Figure 2.5: Scheme (a) and photo (b) of the sensorised matrix.

of the overall shape of the tissue and allows mapping between the electrical space and the expression space. This method is in line with current thinking on perception and action [3] and enables fast correction of the expression space according to the proprioceptive mapping pattern provided by the distributed sensing fabric. A similar procedure has been applied to the problem of recognising the position of an arm from a set of electrical signals originating from tissue sensors positioned on a shoulder joint with excellent results [10].

Structures that could be implemented by means of this technology are numerous. In the case of FACE, we have developed a sensorised matrix (Fig. 2.5), which has been integrated into the silicone-based mask. The matrix resistance can be read from its boundaries. In Figure 2.5a, the lines are the sensors, which are all connected with each other. Data can be generated by multiplexing the injection of a current in the different lines. This produces different voltages across the lines as a consequence of a deformation; the voltages are acquired and processed by a dedicated electronic device.

A liquid-filled cellular matrix, like human skin, is easily elongated, whereas elastomers require considerably greater force. This means that powerful and thus large actuators are required, or the skin-architecture needs to be modified. Several methods for such modifications are available. The first employs liquid- or air-filled pouches in the skin, but this is complicated and costly. The second method involves the strategic reduction of skin thickness. The latter technique

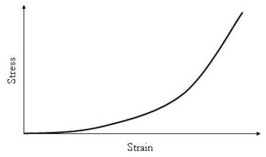


Figure 2.6: Typical stress-strain curve of the facial soft tissue.

has been adopted for FACE. Once poured, the silicone can be modelled so as to obtain different thicknesses of skin. The nose and the boundaries of the face can be made thicker so as to fix the skin to the skull in those regions. The resulting variable skin thickness enables more lifelike dynamics of the skin, more closely resembling natural facial expressions.

Should silicone not respond in an appropriately dynamic manner, we will examine the feasibility of using poroelastic hydrogels with embedded pre-stressed sensing fibres to realise the dermal layer of FACE.

2.2.5 Human muscles and artificial actuators: borrowing from anthropomorphic mechanics

In the human face, more than 55 000 different facial expressions can be generated by more than 200 voluntary muscles [61, 13]. These numbers give a rough idea of the extreme complexity, from an engineering point of view, of the mechanical system represented by the human face. A further complication, which defies any attempt at analysis, modelling and reproduction of such a system, is introduced by the intrinsic nonlinearities of the biological materials and controls underlying its mechanical functioning. In fact, muscles consist of bundles of contracting fibres, which pull nonlinear facial soft tissue (Fig. 2.6) to which they are bounded, and are driven by a nonlinear control, similar to that described by Feldman.

Furthermore, the number of facial muscles is much higher than the actual degrees of freedom, and while, generally, in the rest of the body the primary muscle associated with a particular action is easily identified, the complexity of facial expressiveness implies that more than one muscle generally performs a given action. Therefore, it is the concerted action of several muscle contractions that ultimately gives rise to a particular expression. The physical consistency of skin as well as the synergy between it and facial muscles provide the appearance that we are used to and trained to believe and that is so meaningful to us at both the conscious and unconscious levels. This aspect is of great importance in a static face and more so where dynamic behaviour is concerned.

2.2.6 Biomimetic actuation using Feldmans muscle model

The actuation of a believable and anthropomorphic facial automaton involves the solution of problems having a level of complexity comparable to that of the bio-mechanical architecture of the human face. In fact, considering the viscoelastic behaviour of the silicone skin of FACE, believable anthropomorphic control of the actuators employed to move the skin is a challenging task requiring significant technological and engineering breakthroughs.

In the context of actuation, believability is the ability of the system to produce recognisable expressions through movements similar to those exerted by biological muscles. This means that each point of the skin should not only move to a given position by the action of a definite force, but should do so with the fluidity and grace of biological systems. This level of performance could be obtained by means of artificial actuating devices with control of their geometrical status as well as of their stiffness, so as to mimic biological muscles. In fact, the variable slope of the mechanical force-length characteristic of a muscle allows humans to modulate their interactions with their surrounding environment more or less softly in relation to the intrinsically variable compliance of their muscular system.

Many of the features of muscular contraction can be described by Feldmans model. In this model, a muscles contracting force (F_m) depends on its intrinsic pseudo-stiffness (k), its effective length (x) and its rest length (λ) as follows:

$$F_m = k(x-\lambda)^2 u(x-\lambda)$$

where $u(x-\lambda)$ is the Heaviside function. Agonist and antagonist muscles can be grouped together, and position and stiffness of a pair or more of muscles can be controlled through modulation of muscle rest lengths.

A device force-length characteristic similar to that described by Feldman would allow the desired control of the system stiffness. In this regard, a methodology of control of such an actuating device aimed at reproducing the mechanical behaviour of human muscles was theoretically developed. This biomimetic control is based on a functional structure and a driving strategy of the device inspired by those adopted in biological systems. The Feldman muscle is made up of a set of active elements (motor units) with different rest lengths. These units are activated by a progressive recruitment, whereby the quadratic characteristic of the entire muscle is due to the superimposition of those of the elementary units. Following this concept, a bundle of artificial active fibres was considered as the actuating macro-device, and a suitable algorithm was developed to calculate values of a definite set of fibre parameters necessary to reproduce the characteristics of Feldman type muscle. The algorithm is valid for fibres assumed to be able to elongate or contract in response to an electrical stimulus regardless of their physical constitution [18]. This type of control algorithm could enable the expression of FACE to be much more human-like than those used in traditional robotic control.

2.2.7 Materials for actuation

One of the most challenging tasks in the development of FACE is the realisation of high performance actuators. They should enable actuation of an artificial skin and possess static and dynamic characteristics similar to those of human muscles to attain a satisfactory degree of believability. Many efforts are presently focused on the realisation of pseudo-muscular actuators showing performances similar to those of natural muscles, such as built-in tunable compliance, large strains, high active stresses (0.10.5 MPa), high efficiency, low volume/power ratio ($<$ 10-8 m3/W) and fast response times (e.g. the average shortening speed of an unloaded sarcomere is about 5 μm/s).

Traditional and consolidated actuating technologies consent the achievement of only few of these characteristics, and, for several years, attention has been focused on polymer devices. In particular, we believe dielectric elastomers, which have gained growing interest during the last few years, are promising superior actuating materials for use in FACE. When a thin film of an insulating rubber-like material is sandwiched between two compliant electrodes, and a voltage difference is applied between them, the polymer undergoes an electric field-sustained deformation, which is mainly due to the electrostatic forces exerted by the electrode-free charges [88, 89]. The resulting strain is proportional to the square of the applied electric field. Dielectric elastomers possess excellent figures of merit in several respects: linear actuation strains of up to 60%, fast response time (down to tens of milliseconds) and they generate stresses of the order of MPa. The price for achieving these high-level performances is the very high driving electric field needed (order of 100 V/μm) [90]. Many actuating configurations made of dielectric elastomers have been proposed, including planar, tube, roll, diaphragm and bender [19].

Linear actuators showing electrically activated contractions or, alternatively, elongations, can be advantageously used, for the actuation of the skin of FACE, within an agonist-antagonist couple. Such devices are useful for the emulation of the resulting functionalities of fusiform muscles and as well as sphincter muscles. For example, a linear compliant device can be easily looped to confer it sphincter-like capabilities owing to its internal state of stress.

A new configuration for a linear actuator, which is expected to be able to sustain contractions in response to an electrical stimulus (as for muscles), is currently under development. It consists of a dielectric elastomer having a structure twisted along its central axis, which has to be completed by the integration of two compliant electrodes having the same shape [6].

Finally, novel engineering approaches could be successfully used to simulate sheet muscles. Such muscles cannot be represented by a dielectric elastomer actuator having a simple planar configuration, which would be able only to actively elongate in the planar direction, instead of contract, as required. The micro-fabrication of morphologically bio-inspired actuating configurations may allow the realisation of actuators similar to facial sheet muscles (such as the frontalis muscle).

Once realised and tested, dielectric elastomer actuators, which are most suited for application in FACE, will be embedded in the passive silicone architecture of the AH with particular attention to their size and placement.

2.3 Future directions

The development of FACE is a long-term project that will evolve as new technological breakthroughs in materials engineering, control and other fields are made. The individual modules comprising FACE are under development, and their assembly will lead to new technological problems, which we are currently tackling; for example, the integration between artificial vision, the eyeball system and artificial skin, and the integration between artificial actuators and piezoresistive sensors. Interestingly, the piezoresistive sensors described in the section Kinaesthetic proprioception are much less sensitive to variations in electric fields than piezoelectric sensors. This is an advantage since they will require less rigorous shielding from the high electrical fields needed for actuating dielectric elastomers. In any case, should the processes of sensing and actuation interfere with each other, they can be effected at different instants since the frequencies of electronic acquisition and stimulation are far greater than those required for perceiving and generating expressions (the order of several milliseconds).

In the immediate future, attention will be focused on the use of the FACE system in therapy for people on the autistic spectrum; however, the concept of a believable humanoid display has far-reaching implications. Indeed, implications and applications of the whole system, or even of sub-aspects of this research, could potentially span a wide variety of fields. These can range from posing the basis to introduce new channels of interactivity in other intelligent artificial systems; exploring possible medical applications or exploiting more refined expressivity for the movie industries needs; to philosophical and psychosocial implications, or possible areas of inter-exchange with neurosciences. Moreover, it can be predicted that once the technological building blocks for constructing a believable facial display are developed and refined, the assembly of a whole mechanical android could become a real possibility.

To endow such a system with the elusive quality of believability, reducing expressive noise and disturbance in the interaction with the observer, in our opinion three essential elements should coerce in a synergistic fashion. It is the fusion of these three elements that can contribute to design a framework for lifelike artifacts. First, the choice of the real material to be used to build the system is crucial. As described here, we must borrow from the advances made in the past few decades in polymer science and new breakthroughs in smart biomimetic materials research. In biological systems, the basic building materials are soft gel-like macromolecules. From a physical point of view, they are lowly materials, floppy, imprecise, noisy and lossy. On the other hand, they are versatile and can be used for transduction, sensing, conduction and actuation [13]. An important feature of biological tissue is its multi-component and bi-phasic nature. This endows it with nonlinear properties and wide dynamic ranges. To be believable, then, the artificial entity must be constructed using materials with, among other biomimetic features, built in compliance, nonlinearity and softness. The second must is to confer the entity with the necessary human-like mechanical behaviour to obtain lifelike motion. Anthropomorphic mechanics and control attempt to replicate the characteristic features of human movement: many degrees of freedom, high dimensionality of the control space and redundancy. Achieving this requires not just the appropriate choice of actuating materials (slow-twitch and fast-twitch actuating fibres), but also the appropriate choice of variables and frame of reference with which to describe and predict human-like dynamics. The third element borrows from the neurosciences. In accordance with Western thinking from the time of Plato, logic and reasoning have been placed on a pedestal and emotions and passions considered as inferior animal processes. Two major shifts in thinking in the neurosciences have recently emerged. First, the brain is now considered as a predictive organ, in which consciousness emerges from different levels: the primordial protoself, the core consciousness and the extended consciousness [8]. Second, logic and reasoning are driven and assisted by emotions, so the two are now on an equal footing. Moreover, the study of the relationship between perception, sensing and action suggests that perception and cognition are inherently predictive, allowing us to anticipate the consequences of current or potential actions [2]. Thus, we can describe the brain as a simulator that is constantly inventing models to project onto the changing world, models that are corrected by steady, minute feedback patterns arising from kinaesthetic and proprioceptive sensors in interactions with the environment. This process of interrogation and updating position enables us to navigate the world around us with the minimum of neural processing [3]. On the contrary, current thinking in robotic control is dominated by a completely different paradigm, which separates the domains of sensing, action and control with obvious engineering consequences of excessive time and computing. We are still far from reproducing these features in man-made systems; however, it is essential to keep these paradigms in mind when designing biomimetic artifacts. Technically and theoretically, the culmination of the process might be a revisited form of the Turing test, with the human face and the artificial FACE indistinguishable, when silent, to a human observer. We could, for example, imagine a test that could be used to validate the work done by a facial display; in other words, a figure of merit for believability. Let us rephrase Turing's question in our context: is it true that a facial automaton built by adequate materials, with a suitably large database of expressions and equipped with appropriate controls, can be made to play the imitation game so satisfactorily as to fool a human judge? What is generally understood today when one talks about the Turing test could then assume the following form: lets imagine in a room, a human, we name him Judge, and two counterparts of which he can only see the faces. One is a human, A, and the other is the FACE mimicking As emotional responses. Judges task is to find out which of the two candidates is the machine and which is the human, only by looking at their facial expressions. If the machine can fool Judge, then the task of conferring believability to FACE can be said to be satisfactorily achieved. In the engineering and AI (artificial intelligence) communities, this can be considered as an ultimate goal and, at the same time, as a method to measure the advancements obtained. Given the complexity of human communication and emotional life, this goal, if at all attainable, is still a long way away as far as man-made artifacts of this kind are concerned. In our interdisciplinary team, discussion about this subject is still open, with some of us also posing ethical and philosophical questions about the point. One is, given the crucial role of emotions and their expressions in regulating human identity and social exchange, should perfect believability ever be desirable? This question should be reconsidered not only ethically but also on the basis of the specific knowledge human sciences have accumulated about our species. The second is that exploring lifelike systems in depth might also mean keeping our minds open to the possibility of renewing and evolving models of thought toward new forms and creative dimensions that may be previously unimaginable, and being aware of the risks the adventure may imply.

Chapter 3

Framework Core: Control Architecture

3.1 Modular, Reusable and Object-Oriented Framework

The author presents an architecture for a dynamic and efficient management of multi-transducer data processing techniques. The design of a versatile instrument for data management and elaboration should be suitable for those systems which are equipped with distributed transducer devices, where a particular attention should be paid to inter-process communication. Applications of such instrument space from the simple elaboration of signals supplied by sensory and actuating networks, to pattern analysis and recognition techniques. Design specification included the ability to let the system to be able to operate in real-time. F.A.C.E. has been used as test field for the generic control architecture. F.A.C.E. is currently equipped with various sensory systems (artificial video and audio system, facial proprioception system) and actuating systems (artificial muscles, servo-motors) and includes several cooperating control modules.

Modules for acquisition, analysis and control have been realised for each of the transducer device of F.A.C.E. Modules include artificial vision, face tracking, eye tracking, facial expression recognition, attention system and motor control. The realization of a framework able to perform a parallel device management should give to all the modules the ability to cooperate and the possibility to share data coming from sensory systems and directed to the actuating systems. The possibility to operate in real time imposes critical efficiency requirements to each single module. The framework design pays attention to the management and the synchronization of data and processes. Control modules and pattern recognition algorithms are defined as application processes inside the framework. Such processes inherit properties and functionalities from the framework base structures, taking advantage of process automation provided by the framework core. The framework is realised as a software library in order to exploit the potentials of the computational algorithms and to enhance the performances of the processing techniques based on artificial neural networks. The architecture is able to manage at the same time transducer devices and data processing. Synchronization among modules and data flow is managed by the framework offering remarkable advantages in simulation of heterogeneous complex dynamic processes. Specific control processes, pattern recognition algorithms, sensory and actuating interfaces may be created inheriting from the framework base structures.

In order to exploit the potentials of the computational algorithms and to enhance the performances of control processing techniques, the framework has been realised as a C++ software library. In this way the library architecture is a re-programmable instrument available to the user to develop specific applications. It has been designed to be portable to any software platform in order to gain abstraction from the operative system. The framework however needs a low-level software layer to perform kernel re-building and low-level system calls. An Intel-based personal computer is actually being used and a commercial operative system grants the low-level communication.

3.2 I/O Device Communication

The framework core and application processes are interfaced with the outside world through the framework I/O interface. This layer has been developed in order to act as a buffer for the flow of information coming in from the sensors and out to the actuators. With this strategy sensory fusion is gained enabling an abstraction with respect to the specific technology of the transducers used. Signals coming from the sensors are gathered in parallel and are encoded according to a standard protocol. The encoded information is received by a specific filter for each sensor, which then sorts them to framework I/O interface. For each actuating system a mirror image architecture has been reproduced with respect to the one described for the sensors. The information available in the framework I/O interface is encoded by a filter using the same standard protocol. A specific interface for each actuator pilots its specific hardware system. This architecture allows setting up a communication language between the framework core and sensory and actuating devices. This guarantees an increased flexibility thanks to the presence of interfaces performing the function of interpreters for the specific hardware and filters which specify the way the framework core senses and communicates the information. Fig. 3.1 shows the flow of information to and from the framework core. Communication channels are established as connections between application processes so that framework is able to perform a low-level inter-process communication. The domain of data flowing through connections and the flow chart of the application processes are user-defined.

3.3 Parallel Distributed Processing

Synchronization among modules and data flow is managed by the framework offering remarkable advantages in simulation of heterogeneous complex dynamic processes. Specific control processes, pattern recognition algorithms, sensory and actuating interfaces may be created inheriting from the framework base structures, taking advantage of process automation provided by the framework

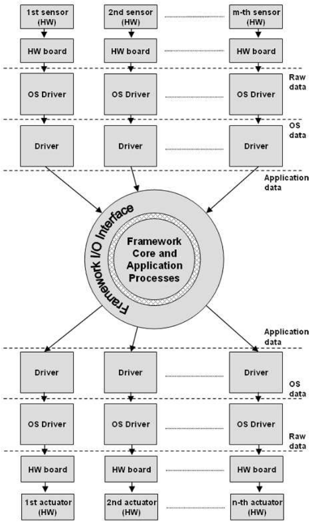


Figure 3.1: Architecture of the framework for the parallel management of multiple elaboration processes. The transducer devices are synchronized and controlled through an appropriate I/O interface

core. A spatial definition of the entities involved in the framework can be supplied, making this information available to the control system for subsequent processing. To guarantee the execution of real-time applications an inner synchronization signal is provided from the framework core to the processes and to the framework I/O interface, enabling to gain time-space correlation. A dynamic geometrical representation can be visualised by a high efficiency 3D graphic interface, giving a support during experimental setup debug. Processes and connections are managed at run time and they can be manipulated under request. The presence of dynamic structures implies a configurable resource management, so the framework offers an optimised interface for enumeration and direct access requests.

3.4 Development of Control and Processing Modules

The framework architecture has been designed as a hierarchical structure whose root is a manager module (*3DWorld*). It is realised as an high-level container of generic modules representing the environment in which process modules (*W*) and I/O filtering interfaces are placed (*Drivers*). All these modules inherit low-level properties and functionalities from a base module (*3DObject*) realised as an element able to populate the process environment. Virtual and pure-virtual functionality strategies have been applied to this base module in order to obtain an abstraction with respect to the generic application task. In this way the core is able to process user-defined functionalities without being reprogrammed. Moreover, modules may be grouped recursively (*WGroup*) in order to share common properties and functionalities of entity modules belonging to the same type. Communication channels are realised as connections (*WConnection*) through specific projection types that specify the connection topology. Connections are delegated to dispatch synchronization information and user-defined data (*WConnectionSpec*). The filtering interface modules are able to

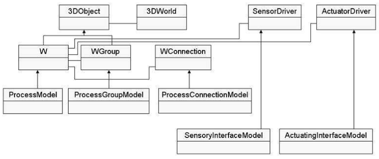


Figure 3.2: Inheritance and collaboration diagram of the main modules of the framework core: I/O interfaces, communication channels, processes

drive the transducer hardware and to dispatch information to process modules. All base modules manage dynamic structures and they are designed to maintain data consistency while the environment state may change. This behaviour permits the execution of dynamic and real-time parallel distributed processing while synchronization and data flowing are managed by the environment. All modules are realized as running processes while their control and synchronization is managed by the framework. The hierarchical and collaboration chart of the base structures is shown in Fig. 3.2.

3.5 Framework Overview

Architecture implementation details will be showed in this section making use of the Unified Modelling Language (UML) representation. In Fig. 3.3 the legend is showed.

The boxes in the above graph have the following meaning:

- A filled black box represents the struct or class for which the graph is generated.
- A box with a black border denotes a documented struct or class.
- A box with a grey border denotes an undocumented struct or class.

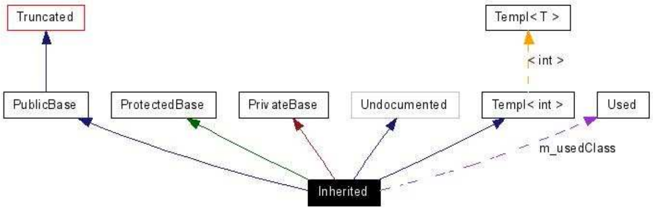


Figure 3.3: Graph relationship legend

- A box with a red border denotes a documented struct or class for which not all inheritance/containment relations are shown. A graph is truncated if it does not fit within the specified boundaries.

The arrows have the following meaning:

- A dark blue arrow is used to visualize a public inheritance relation between two classes.
- A dark green arrow is used for protected inheritance.
- A dark red arrow is used for private inheritance.
- A purple dashed arrow is used if a class is contained or used by another class. The arrow is labeled with the variable(s) through which the pointed class or struct is accessible.
- A yellow dashed arrow denotes a relation between a template instance and the template class it was instantiated from. The arrow is labeled with the template parameters of the instance.

The implementation will be described using technical object-oriented language terms. To avoid confusion main terms are resumed:

- an *object* is an synonymous word for a *class* or a *structure* and it defines a new data type.

- an *instance* of an object is a variable declared as object type.
- an object's property is called *member*.
- an object's function is called *method*.
- a *derived* object is a child of another object, and it inherits properties and functionalities from his father object, which is called *base* object.
- a *private* member or method is accessible only inside the object.
- a *protected* member or method is accessible both from the object and from a derived object.
- a *public* member or method is always accessible.
- while structures start implicitly with public definitions, classes start implicitly with private definitions.
- a *virtual* method defines a function that, if it is overridden in a derived object, cause the call to derived method even it is called on the base object.
- a *pure-virtual* method is undefined in the base object and, as a result, the base object can not be instantiated.
- an *abstract* object contains only pure-virtual methods.

3.5.1 Portability

Since the framework library has been written using the C++ programming language, the software portability is guaranteed by the standard ANSI-C/C++ definitions. However the low-level interfaces depend on the particular libraries of the operative system. For this reason a low-level layer has been defined to include all the dependencies for the specific platform. The low-level layer (*ARI_Macro*) has been realised defining a set of operations for run-time memory management (allocation and deallocation), file I/O interface, log reports and window assert dialogs. All these operations has been implemented as C++ macro which will be used by the framework for all the low-level operative system interfacing.

Graphic User Interface (GUI) has not been embedded in the framework in order to not slow down the application efficiency and to let the user to be able to choose his preferred tools. Since each of the framework objects provides functions to get information about the status and output data, the GUI tools have been developed as external modules that can be linked to the architecture. Main graphic output is guaranteed by OpenGL rendering [34], which libraries are available for many hardware platforms. If the user choose to use OpenGL support, then he must link OpenGL libraries to his application. Application GUI is actually supplied for Microsoft Windows operating system, including useful tools for layered graphs (*MGraph*) and OpenGL dialog windows (*glCDialog*). While data storage is already supplied by the framework, additional end-user tools are available for file tables (*TabData*) and database tables (*TabDataConn*) supporting MySQL, SQLServer and general ODBC drivers. Since all these tools are external they not compromise the architecture efficiency, making the user able to choose his appropriate strategy.

3.5.2 Containers

Container structures have been developed adopting a template-object strategy. A chunk-memory-allocation strategy has been applied to dynamic containers in order to obtain a configurable compromise between flexibility and direct memory access efficiency. Iterators have been defined for such dynamic structures in order to perform high-efficiency list browsing. Static arrays are able to perform real-time memory reallocation. Basic containers are showed in Fig. 3.4.

3.5.3 The base structure: 3DObject

The basic element of the framework is called *3DObject*. This is an element able to perform basic functions which are useful to many of the modules included in

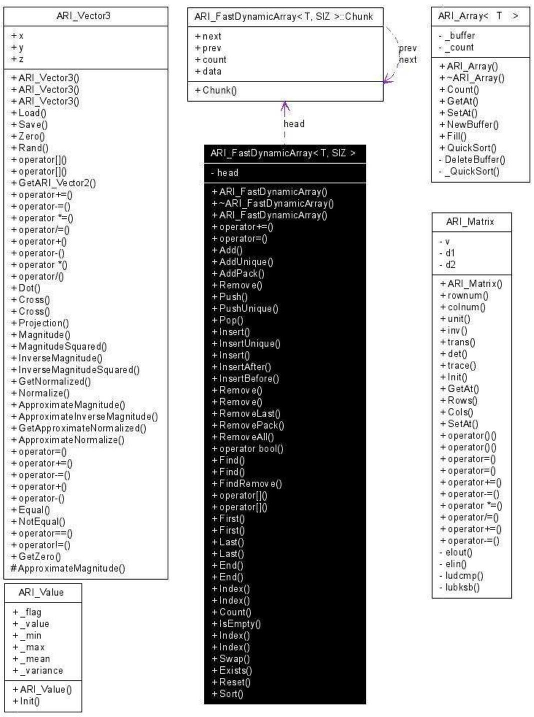


Figure 3.4: Basic containers implemented in the framework. A chunk-memory allocation stategy has been applied to dynamic containers

the framework.

3DObject properties include a label, a 3D position, a 3D radius and a set of user-defined flags. The label will be useful for log reports, application debug and for easy high-level object identification. The default label will be associated with the hexadecimal memory address of the object, which is unique during the execution of the application. Three-dimensional position and radius will be useful for graphic rendering under user request. Such values are initialised with null values, since the rendering functions are optional. Flags specify the way in which the object manager will process the object (see *3DWorld*). By default no flag is set for a *3DObject*.

Several functions has been supplied for this base object in order to guarantee I/O transfers, 3D management and flags maintenance. Since process objects will be derived from *3DObject*, a set of virtual functions has been defined to perform process synchronization and update. The use of virtual functions guarantees the ability of the object manager to call redefined functions on derived objects without knowing them. User-defined processes have to follow the base virtual protocol, redefining the way in which synchronization and update operations are performed. In Fig. 3.5 the *3DObject* architecture is showed.

Defined virtual functions are *Render*, *SetInput* and *Update*. Each of these functions will be called by the object manager only if the appropriate flag is set. Such strategy makes the user to be able to specify virtual overrides for his specific processes. *Render* function will specify the way in which the object will be graphically rendered. OpenGL is the default renderer, permitting an high efficient three-dimensional output. Graphic rendering is useful for object status monitoring during application execution and debug. The definition of a rendering function does not reduce object efficiency since *Render* virtual function will be called only if screen repainting is needed and if the rendering flag has been set.

SetInput and *Update* functions specify the core process algorithms. The first method is needed to workaround the serial processing of all the objects. In fact the parallel execution of all the process objects will be serialized by

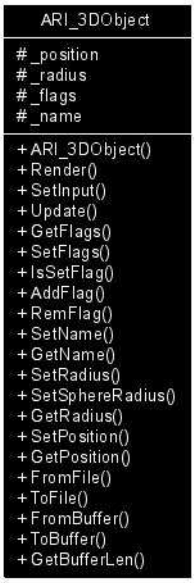


Figure 3.5: Architecture of the 3DObject structure. This is the base object used to populate the process environment managed by the 3DWorld structure

the low-level CPU scheduling. The order of the execution of each process may compromise the effective result of the process network in the case of multiple cooperating processes. For this reason the object manager will first call the *SetInput* method over all the processes with the intent of buffering the actual output data over communication channels (see *WConnection*). After this buffering step, the *Update* method will be called over all the objects making use of the buffered connection data instead of the object actual output data. Such a strategy guarantees the independence from the process execution order. Default base method just defines the virtual processing and rendering protocol.

The same virtual strategy has been adopted for buffers and files data storage. A set of virtual functions has been defined to perform an object independent way for file I/O (*ToFile*, *FromFile*) using platform independent file operations. Such methods use buffer virtual functions (*ToBuffer*, *FromBuffer*, *GetBufferLen*) which will be specified by the user for each process. Default implementation of buffer virtual methods just defines the buffering I/O protocol.

3.5.4 The object manager: 3DWorld

Synchronization and update of all the running processes is managed by *3DWorld*. This structure is a collection of *3DObject*. Elements are organised in a dynamic list where the access order is often sequential. *3DWorld* provides the execution of specific methods for all the objects added during an initialisation step. Internally *3DWorld* manages the reference of each object and not the object itself. This strategy speeds up the enumeration of the elements, giving to the user the opportunity to override the manager behaviour and to gain the direct control of each single element. Elements may be added and removed at run-time (*AddObject*, *RemoveObject*) while the execution of process virtual methods is managed accordingly to the active flags of each elements (*Render*, *SetInput*, *Update*).

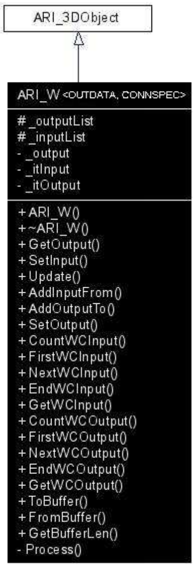


Figure 3.6: Architecture of the W structure. It represents the base process model

3.5.5 The base process structure: W

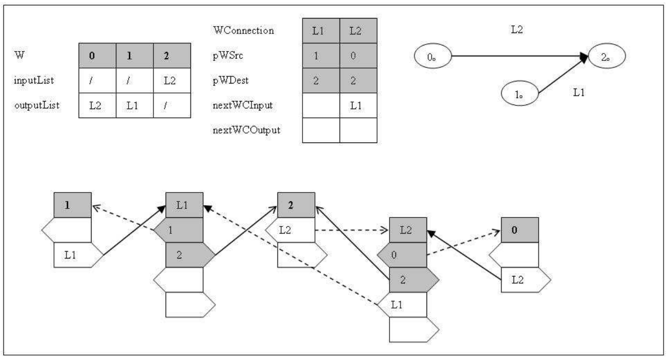
The process base structure inherits properties and functionalities from *3DObject*. *W* is a generic transferring function, which is able to communicate with structures of the same base type. Communication channels are established as connections and the domains of input and output data are defined by the user. For this reason a pure-virtual strategy has been applied to this structure to take into account a processing method (*Process*) which is still not defined at this level. For this reason a *W* structure can not be instantiated. It only defines the process standard protocol and provides topology connection methods. *Render*, *SetInput* and *Update* flags are automatically activated in the initialization step. The architecture of *W* structure is shown in Fig. 3.6.

W is a template structure where *OUTDATA* and *CONNSPEC* types are user-defined. While *OUTDATA* represents the output pattern, *CONNSPEC* contains the specification for each input connection. The *3DObject* virtual method *Update* is redefined to set the new output value accordingly to the result of the pure virtual method *Process*, which will be specified in derived process structures. Actual process output can be retrieved using *GetOutput* method. *W* provides methods to specify the connection topology of the single process (*AddInputFrom*, *AddOutputTo*) in respect to the other input and output processes. *SetInput* method is redefined to perform a scan over all the input connections and to transfer the actual output value of input process to the connection buffer (see *WConnection*). For sake of efficiency the starting point for both input and output connections is stored in each *W* structure . Since connections are usually browsed sequentially, enumeration methods are provided to perform high efficient navigation over input and output connections (*First*, *Next*, *End*). In derived structures the *Process* method has to be redefined to perform the core process algorithm, taking into account the specific data connection and the data output given by each input process, which are already buffered into the corresponding input connection. *Process* method is defined as private function, so it can not be directly called by the user since it is automatically managed by the framework during the update step. With this strategy the process, which core is still undefined, is able to manage the unknown information of the user-defined process.

Buffer I/O operation (*GetBufferLen*, *ToBuffer*, *FromBuffer*) are redefined to provide data storage for actual process output and for the actual specification values for all the input connections.

3.5.6 The connection between two processes: WConnection and WConnectionSpec

A connection between two process is realised by the *WConnection* template structure. Since input connections for a given process are embedded inside the process itself, the template types are the same used for *W* structure. *WCon-*



W	0	1	2
inputList	/	/	L2
outputList	L2	L1	/

WConnection	L1	L2
pWSrc	1	0
pWDest	2	2
nextWCInput		L1
nextWCOutput		

Figure 3.7: Connection strategy example. Each process contains the references of both the first input connection and the first output connection. Each connection contains the references of both the source and the destination process, plus the references to both the next input connection and the next ouput connection

nection stores references of both source and destination processes. A field of *OUTDATA* type is stored to perform data buffering. In fact during the *SetInput* step on the destination process, the *GetOutput* method is called over the input process and the resulting value is stored inside the connection. While the *W* structure contains the references of both the first input connection and the first output connection, the next links to connections are stored into two dynamic lists inside the *WConnection* structure. In Fig. 3.7 an example of this strategy is shown.

Inside the *WConnection* structure a field of *CONNSPEC* type is used to store the specific information needed by the destination process to evaluate the output data given by the source process. The base class for the *CONNSPEC* type is the *WConnectionSpec* structure. This structure is an abstract class where only a pure virtual method is defined (*Init*). This method will be automatically called during the creation of the connection to perform the initialisation of the *CONNSPEC* values on the basis of the user-defined parameters. This structure only defines the connection initialisation protocol and must be redefined in the derived structures.

3.5.7 Grouping processes: WGroup and WProjectionSpec

Processes may be grouped into a *WGroup* structure, which template arguments are the same of the *W* structure (*OUTDATA*, *CONNSPEC*). Since this structure only contains the references to several *W* structure, the user does not lose the control over each single process. Grouping processes results in a logical managing of several objects, which can be added and removed at run-time (*Add*, *Remove* methods). *3DObject* is the base class for the *WGroup* structure, permitting the redefinition of the virtual functions for flag, synchronization and I/O mangement over all the grouped entities. *WGroup* contains its own flags, with the possibility to propagate flags to each grouped entity (*AddFlags*, *RemFlags*, *SetFlag*, *IsSetFlag*). With the same strategy adopted for the *3DWorld* structure, *WGroup* is able to propagate rendering and synchronization signals over grouped entities according to their active flags. By default these flags are activated only for the group and they are not propagated to the sub-entities. Since the *WGroup* structure has its own geometrical position and volume, a *Dispose* function is supplied for geometrically disposing all the grouped entities according to their volumes. Such information will be automatically taken into account during the framework rendering processing.

WGroup contains useful methods to create connections both to other groups and to other single processes. Since these methods (*AddInputFrom*, *AddOutputTo*) may involve more than one entity, connections are realised through projection specifications (*WProjectionSpec*). During the projection initialisation step, the framework looks for projection flags in order to perform the user-specific connection strategy. Currently *one-to-one*, *N-to-one*, *uniform random-to-N*, *sub-group-to-N* and *polar-random-to-N* flags are supported (see *WGroup.h* header file).

Since the entities inside a group are often browsed sequentially, high efficient iterators are defined also for this structure. I/O buffering operations are

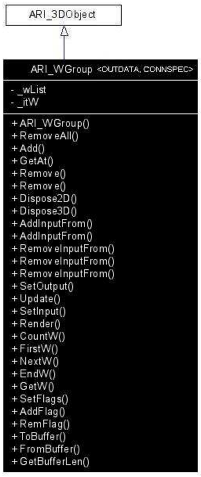


Figure 3.8: Architecture of the WGroup structure. Many entities may be grouped togheter in order to create an high-level entity able to populate the process environment

redefined to automatically join I/O buffering operations of each grouped entity. The *WGroup* architecture is shown in Fig. 3.8.

3.5.8 The interface with sensor and actuator systems: SensorDriver and ActuatorDriver

The *SensorDriver* structure is the starting point for filtering input data coming from the generic input system. This structure directly derives from *3DObject* and uses *IFNeuron* and *IFNeuronGroup* structures. These two structures are specifically designed for artificial neural networks using the integrate-and-fire model, as it will be discussed later. However at this level they are used as buffering structures where the process functions (*SetInput*, *Update*) are skipped by the framework. This choice, as it will be shown, does not cause lack of efficiency and of generality.

Since sensor information have to be available before then the framework update step, the *SensorDriver* objects have to be added into *3DWorld* by the user before than other process object. During the construction of the structure, only the *SetInput* flag is set. *SetInput* method will be redefined for the specific hardware in derived structures, where the data filtering algorithm will be specified.

During the initialisation step (*InitDriver*) the user will specify the length of the sensor buffer data needed during the hardware acquisition step. At this point the driver just allocates the memory space in a static array, and it waits for the registration of the *IFNeuron* entities. These entities represent the objects where sensor data will be mapped. Such entities will be specified using the *Register* method, where the *SetInput* and *Update* flags are automatically removed from each registered entity. *SensorDriver* structure will be operative only when all of the needed mapping entities will be registered by the user. In order to speed up the acquisition process, the references of the *IFNeuron* objects are indexed during the registering phase. Such a strategy permits a direct memory access over all the registered entities. The user may choose to switch on/off the driver using the *SetPowerOn* method. The *SensorDriver* architecture is shown in Fig. 3.9. The *ActuatorDriver* structure follows a similar architecture.

3.5.9 General Purpose Instruments

Since the framework permits module sharing and reuse, several tools have been developed as auxiliary instruments. Specific filtering interfaces have been realised for F.A.C.E. sensory systems (facial proprioception, vision system, audio system, chewing system). Process modules for F.A.C.E. basic control (Face-Tracking, Eye-Tracking and Attention System) and signal processing and data analysis (FFT, PCA, LDA) have been developed.

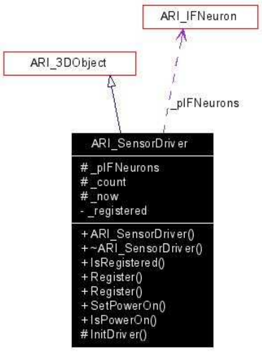


Figure 3.9: Architecture of the SensorDriver structure. This object represents the base model for the interface of a generic input system

3.5.9.1 Data input from file: FileSensorDriver

The framework is mainly designed for real-time applications. However off-line data analysis is still useful for applications running classification and prediction algorithms on previously stored data. For such tasks a *FileSensorDriver* structure has been developed according to *the SensorDriver* protocol. This structure supports the typical data protocol adopted in other data analysis tools such as MatLab's artificial neural network tool-box [35], DataEngine [36] and PDP++ architecture [37]. According to this protocol each data acquisition lies in a row of a ASCII text file, including features data and, optionally, desired output data. During the initialization step (*Init*) the structure calls the base class function (*InitDriver*) to allocate the requested amount of memory. Two input files are needed to obtain both training and test data sets. Data are the transfered from files to a local matrix structure in order to speed up the application update steps. If random presentation order is chosen, then an urn structure (*CUrn*) is initialised to perform run-time drawing-of-lots of the matrix rows. Following the base class protocol, the driver will wait the user registration of the entities for both features and desired output data. During run-time acquisition (*Set-*

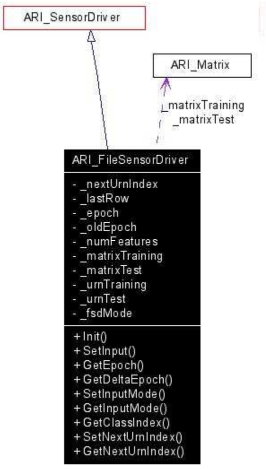


Figure 3.10: Architecture of the FileSensorDriver structure used to obtain data from a previous stored file

Input), the epoch time is taken into account instead of real-time flow. The user may switch training and test phases using the *SetInputMode* method. The architecture of *FileSensorDriver* is shown in Fig. 3.10.

3.5.9.2 Data input from camera: CameraSensorDriver and face-tracking

A *CameraSensorDriver* structure has been developed to perform digital camera image acquisition and filtering. Since data transfer protocol may differ depending on the communication bus used, the framework provides a hardware abstraction layer for cameras. The *HW_ Camera* is an abstract class defining two pure virtual methods for the camera initialisation (*CameraInit*) and the camera image acquisition (*CameraFrame*). Low level camera support is provided through the high efficient OpenCV interface [38]. If the user choose to use camera support, then he must link OpenCV libraries to his application. Actu-

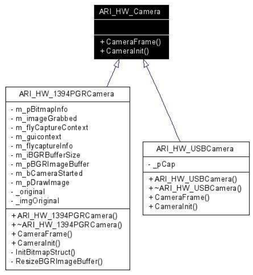


Figure 3.11: Architecture of the camera hardware abstraction. USB and FireWire protocols are currently supported

ally the framework provides the support for general USB cameras (through the *HW_ USBCamera* structure) and PGR FireWire DragonFly camera (through the *HW_ 1394PGRCamera* structure). Base class virtual methods are appropriately implemented for these structures. The architecture is show in Fig. 3.11.

The *CameraSensorDriver* structure provides camera acquisition and face tracking algorithms according to *SensorDriver* base structure. The user must specify the specific camera hardware support, providing the final stretch size for output data. If the face tracking is set (*FaceTracking*), only the sub-image containing the tracked face is mapped. The architecture is shown in Fig. 3.12.

The face tracking algorithm actually uses an object cascade detection technique. The object detector described below has been initially proposed by Paul Viola [39] and improved by Rainer Lienhart [40]. First, a classifier (namely a cascade of boosted classifiers working with haar-like features) is trained with a few hundreds of sample views of a particular object (i.e., a face or a car), called positive examples, that are scaled to the same size (say, 20x20), and negative examples which are arbitrary images of the same size.

After a classifier is trained, it can be applied to a region of interest (of the same size as used during the training) in an input image. The classifier outputs a "1" if the region is likely to show the object (i.e., face/car), and "0" otherwise. To search for the object in the whole image one can move the search window

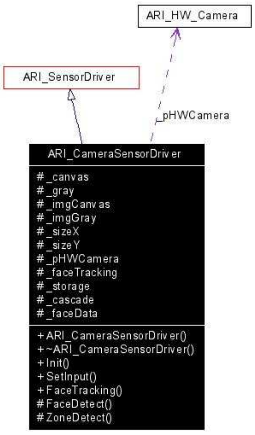
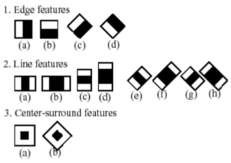


Figure 3.12: Architecture of the CameraSensorDriver structure. The hardware abstraction is gained through the HW_Camera structure

across the image and check every location using the classifier. The classifier is designed so that it can be easily "resized" in order to be able to find the objects of interest at different sizes, which is more efficient than resizing the image itself. So, to find an object of an unknown size in the image the scan procedure should be done several times at different scales.

The word "cascade" in the classifier name means that the resultant classifier consists of several simpler classifiers (stages) that are applied subsequently to a region of interest until at some stage the candidate is rejected or all the stages are passed. The word "boosted" means that the classifiers at every stage of the cascade are complex themselves and they are built out of basic classifiers using one of four different boosting techniques (weighted voting). Currently *Discrete Adaboost*, *Real Adaboost*, *Gentle Adaboost* and *Logitboost* are supported. The basic classifiers are decision-tree classifiers with at least 2 leaves. Haar-like features are the input to the basic classifiers, and are calculated as described below. The current algorithm uses the following *Haar-like* features:



The feature used in a particular classifier is specified by its shape (1a, 2b etc.), position within the region of interest and the scale (this scale is not the same as the scale used at the detection stage, though these two scales are multiplied). For example, in case of the third line feature (2c) the response is calculated as the difference between the sum of image pixels under the rectangle covering the whole feature (including the two white stripes and the black stripe in the middle) and the sum of the image pixels under the black stripe multiplied by 3 in order to compensate for the differences in the size of areas. The sums of pixel values over a rectangular regions are calculated rapidly using integral images. A separate application can train a cascade of boosted classifiers from a set of samples.

3.5.10 F.A.C.E. instruments

3.5.10.1 Facial Expression Estimator: FEECameraSensorDriver

A *FEECameraSensorDriver* structure has been developed to perform the tracking of facial zones. Zones include left and right eyes, nose and mouth areas to be mapped into output driver space. This structures uses the *CameraSensorDriver* as base structure, processing the image in order to appropriately select the four face zone. During the initialisation step the user must specify the stretch size of each zones. The user must also specify the four entity groups which will be able to receive each zone of the facial image. The architecture of the driver is shown in Fig. 3.13.

A simple application has been developed to test the face tracking algorithm. Currently the system is able to easily detect a single subject and to track the face even in noisy environment. Fig. 3.14 shows the face tracking algorithm

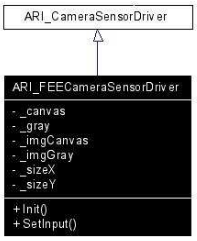


Figure 3.13: Architecture of FEECameraSensorDriver. This object is used to perform the facial expression recognition for human subjects in real time applications

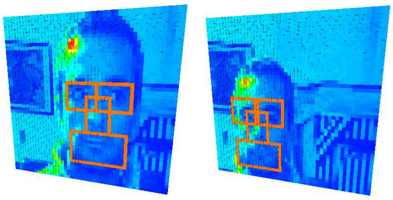


Figure 3.14: Face tracking and facial zone detection using the FEECameraSensorDriver structure

running at 25 frames per second on a standard PC using a digital camera. This structure will be used for real-time automatic facial expression recognition, as it will be discussed in 4.1.3.

3.5.10.2 Acquisition Boards

An hardware board for the generic I/O control has been designed and realised. The I/O board has been developed as a versatile instrument in substitution of the more expensive general purpouse acquisition boards (i.e. National Instruments Acquisition Boards). The goal was to gain the control over the hardware sensory and actuating systems realised for the FACE robot. The board may be connected to a standard PC via the USB connection protocol. The firmware

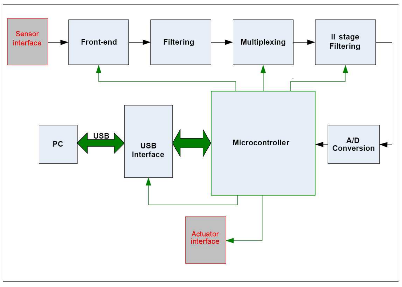


Figure 3.15: Block schema of the hardware board for I/O control

of the micro-controller has been programmed in order to perform the operations needed for calibration, multiplexing, data acquisition and control. The hardware provides the control of 16 transducers at the same time. If a more complex control is needed, the front-end module has to be realised by the user. The communication protocol is designed to dispatch sub-messages to the micro-controller of the front-end device in order to address user-specified requests. In Fig. 3.15 the block schema of the hardware board is showed below.

Such hardware board has been used to perform data acquisition and control of specific systems of the FACE robot, such as the chewing system, the facial proprioceptive system and the artificial muscular system. A specific *SensorDriver* for each of these systems has been realised in order to connect the electronic devices to the framework. A base class has been supplied to grant a simple low-level communication using the USB protocol (*HidDevice*). The derived structures are shown in Fig. 3.16.

3.5.10.3 Proprioceptive System Overview: An Artificial Sensing Skin

The artificial sensing skin realised for FACE is a 3D latex foam, under which lies a sensing layer. The sensing layer responds to simultaneous deformations in different directions by means of a piezoresistive network which consists of a Conductive Elastomers (CEs) composites rubber screen printed onto a cotton

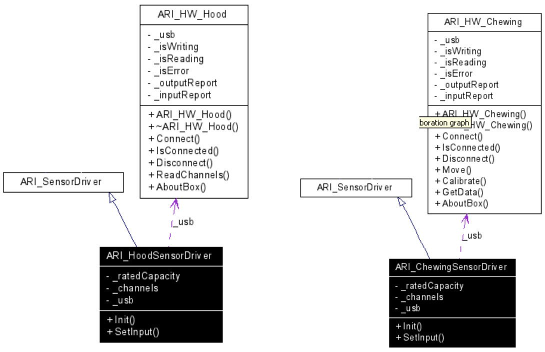


Figure 3.16: Architecture of the HoodSensorDriver and the ChewingSensorDriver structures

lycra fabric. CE composites show piezoresistive properties when a deformation is applied and can be easily integrated into fabric or other flexible substrate to be employed as strain sensors (Fig. 3.17). They are elastic and do not modify the mechanical behaviour of the fabric. CEs consists in a mixture containing graphite and silicon rubber. Resistance, Gauge Factor, Temperature Coefficient Ratio and Reactive Properties have been classified [12]. In the production process of sensing fabrics, a solution of CE and trichloroethylene is smeared on a lycra substrate previously covered by an adhesive mask. The mask is designed according to the desired topology of the sensor network and cut by a laser milling machine. After the deposition, the cross-linking process of the mixture is obtained at a temperature of 130°C. Furthermore, by using this technology, both sensors and interconnection wires can be smeared by using the same material in a single printing and manufacturing process.

From the technical viewpoint, a piezoresistive woven sensing fabric is a system whose local resistivity is a function of the local strain. In a discrete way, it can be thought of as a two dimensional resistive network where single resistors have a non-linear characteristic that depends on the local strain. The integral impedance pattern is a function of the overall shape of the sensorised

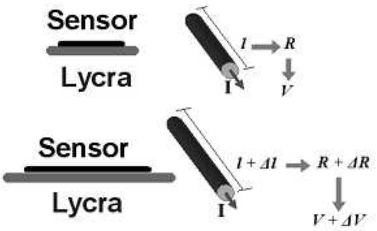


Figure 3.17: Transduction principle of the strain sensor

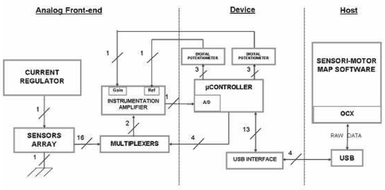


Figure 3.18: Block schema of the acquisition hardware

fabric and allows mapping between the electrical space and the shape space. For the characterisation of the sensors in terms of their quasi-static and dynamic electromechanical transduction properties sensors were serially connected. In this case, a current is superimposed in the circuit and high impedance differential voltages are acquired from each sensor. A block scheme of the acquisition hardware is presented in Fig. 3.18. Two multiplexers allow a sensor to be selected and the relative signal is acquired by a differential amplifier. A microprocessor drives the whole system, performs the analogous/digital conversion and exchanges data via usb interface. The device is provided with an automatic calibration subsystem which allows gain and offset to be tailored to each sensor.

A pushing punch driven by a stepper motor was used to apply alternate mechanical deformations (by indentation) to each fabric based sensor. Several tests were carried out, by using rectangular-wave mechanical stimulations (series

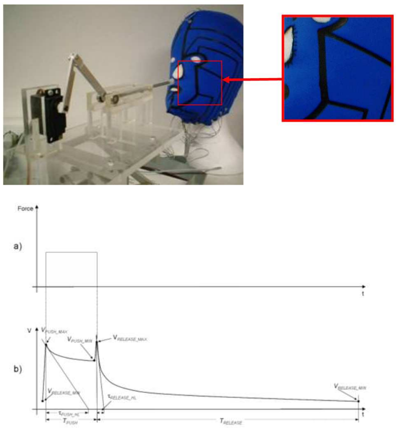


Figure 3.19: Top: facial skin prototype and stepper motor. Bottom: a) A pressure impulse; b) sensor response a pressure impulse and selected features

of pressure impulses). In order to model the electromechanical response of each sensor, an equivalent circuit based on the equivalence between the electrical response (current variation) of the circuit and the response (resistance variation) of the sensor was proposed. Each sensor was tested applying a series of pressure impulses (Fig. 3.19a) and acquiring the differential voltage across the sensor as sensor response (Fig. 3.19b).

Pressure impulses result in a typical differential voltage behaviour showed in Fig. 3.19b. Sensor response shows a peak in correspondence to every mechanical transition. Data acquired were filtered, peaks were detected and relative maximum and minimum, and time constants were selected as features (Fig. 3.19b).

Sensor responses during constant pressure time intervals were approximated by decreasing exponentials selecting the local minimum as steady state value. This approximation results as true as high is the pressure time interval. In order to remove the contribution of high order exponentials, first order time constants were calculated discarding the first 5% of each curve. This choice

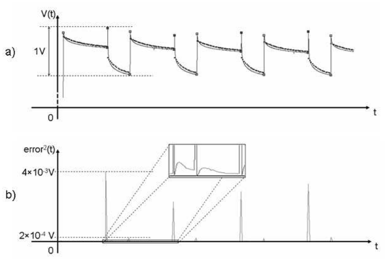


Figure 3.20: a) acquired signal during pressure deformations (continuous line) and extracted exponential law (dashed line); b) quadratic error

allowed quantization errors introduced by the acquisition device in response to rapid transitions to be avoided and sensor steady state deformation, related to slower frequency components, to be maintained.

During a series of pressure impulses, the time constants of the mechanical deformation phases resulted in a mean of 9.32 seconds, while the time constants during the lack of mechanical deformation resulted in a mean of 4.72 seconds. Fig. 3.20a shows the acquired signal during pressure deformations (continuous line) and the extracted exponential discharging law (dashed line); Fig. 3.20b shows the quadratic error. Close to the mechanical transitions the differences between the signal and the exponential law is high; during constant pressure differences are very low ($< 3x10^{-5}$).

In order to model the first-order components of the sensor response (resistance variation) to a rectangular stimulation (applied deformation) the equivalent circuit represented in Fig. 3.21 was considered.

The power supply V is the electrical analogous of the imposed deformation. The switch T1 (initially open) is closed and open in correspondence of, respectively, the beginning and the end of the imposed deformation. The switch T2 (initially open) is closed when T1 is open again. Following a simple analysis of this circuit, it is easy to recognise that the variation of the charging and discharging currents of the capacitance in consecutive phases of stimulation are

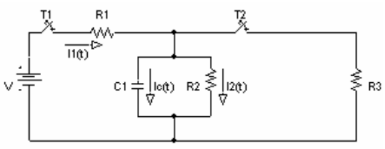


Figure 3.21: Proposed equivalent electric model of each sensor

Feature of the variation of the sensor resistance	Feature of the variation of the charging/discharging currents of the circuit	Symbol
Initial peak [$k\Omega$]	Initial peak [A]	$I_1(0)$
Steady-state value for the deformation phase [$k\Omega$]	Steady-state value for the charging phase [A]	$I_1(\infty)$
Time constant of the first-order exponential components for the deformation phase [s]	Time constant for the charging phase [s]	τ_1
Time constant of the first-order exponential components for the release phase [s]	Time constant for the discharging phase [s]	τ_2

Table 3.1: Considered analogous features

analogous to the variation of the resistance of the sensor during, respectively, its deformation and the following release. The circuit parameters R1, R2, R3 and C can be derived by using the features, extracted from reference experimental signals, listed in Table 3.1.

A circuit voltage of 1 V was assumed as the analogous of a deformation of 1 mm, while a circuit current of 1 A was assumed to correspond to a variation of the sensor resistance of 1 $k\Omega$. Values of the features listed above were extracted from ten cycles of a reference experimental signal and were used to derive the circuit parameters by means of the following system of equations:

$$\begin{cases} \tau_1 = C(R_1||R_2) \\ I_1(0) = \frac{V}{R_1} \\ I_1(\infty) = \frac{V}{R_1+R_2} \\ \tau_2 = C(R_2||R_3) \end{cases}$$

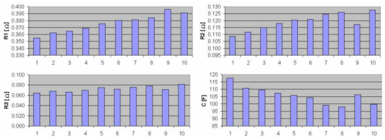


Figure 3.22: Values of the parameters of the equivalent electric model extracted from ten cycles of a reference experimental signal

The solution of this system provided, for the considered ten cycles of stimulation, the results reported in Fig. 3.22.

In consideration of the limited number of tests considered so far, definitive assessments and interpretations of the trends reported in Fig. 3.22 are premature at the moment. Accordingly, we are approaching a second phase of tests, in order to validate such an electrical equivalent model by subjecting each sensor to an extensive campaign of measurements, by applying deformations consisting of rectangular-wave signals with variable amplitudes, frequencies and duty-cycles.

Chapter 4

Framework Extensions: ANN

The concept of artificial neural networks is to imitate the structure and workings of the human brain by means of mathematical models. Three basic qualities of the human brain form the foundations of most neural network models:

- knowledge is distributed over many neurons within the brain;
- neurons can communicate (locally) with one another;
- and the brain is adaptable.

The terminology with which neural networks are described is derived from these three qualities of the human brain, and is as follows:

- structure of the neuron;
- network topology;
- and the adaptation or learning rule.

The neurons, or processing units, which make up the neural network are single elements and consist principally of four components:

- a connection function;
- an input function;
- an activation or transfer function;
- and an output function.

A neuron receives signals via several input connections. These are weighted at the input to a neuron by the connection function. The weights employed here define the coupling strength (synapses) of the respective connections and are established via a learning process, in the course of which they are modified according to given patterns and a learning rule. The input function compresses these weighted inputs into a scalar value, the so-called network activity at this neuron. Simple summation is generally employed here. In such cases, the network activity, which results from the connection function and the input function, is the weighted sum of the input values. The activation function determines a new activation status on the basis of the current network activity, if appropriate taking the previous status of the neuron into account. This new activation status is transmitted to the connecting structure of the network via the output function of the neuron, which is generally a linear function. By way of reference to biological neurons, the activation status at the output of a neuron is also known as the excitation of the neuron.

There are various types of activation functions, the type used often depends on the learning rule. The backpropagation learning rule, which will be described later, requires a differentiable function such as the sigmoid function.

Apart from a small number of exceptions, most of the process units encountered in practical applications can be represented with this general model, whereby minor alterations to the functions may lead to substantial changes in the network's behavior.

A process unit is of interest only as a unit of a network consisting primarily of homogeneous elements. In artificial neural nets, these elements are generally interconnected to form a rigid network structure, as a result of which the learning algorithm only rarely includes provision for the formation of new connections and the removal of old connections, such as occurs in biological systems.

A layered connecting structure is generally employed, whereby the layer on which the input signals act is referred to as the input layer; the layer at which the results are collected is known as the output layer; and the layers located between these are known as hidden layers. The neurons are generally fully connected on a layer-by-layer basis. The number of layers often determines the performance of a network.

A distinction can be made between feedforward, lateral and feedback connections for the method of linking the different layers. Both feedforward and feedback connections over several layers are conceivable.

The connecting structure and the choice of processing units determine the structure of a network. In order to carry out classification, the network must be taught a task by presenting it with examples in a training phase.

The knowledge relating to this task is not stored in a knowledge basis, but is distributed throughout the network in the connection weights. The weights are adjusted in this phase to fit the problem. This is done by means of learning rules or algorithms. In biological systems, a process of learning via structural adaptation also takes place, i.e. connections between neurons may become "extinct," and totally new connections may be established. Artificial neural nets do not normally make any provision for such a process.

The training phase normally proceeds as follows: random values are initially assigned for the weights of the neurons. Patterns from a training data record are then presented to the network and the weightings are adapted on the basis of the learning rule and training pattern until a convergence criterion, e.g. a defined error threshold, is attained. A test phase is then carried out, in which unknown test patterns are presented to the network to establish the extent to which the network has learnt the task in hand.

Selection of the patterns for the training phase is a particularly important aspect. These patterns must describe the task as completely as possible, as in later use the network will only be able to provide good results for problems which it has learnt. This means that patterns must be selected which cover all classes and, where possible, describe the boundary ranges between the classes.

The learning processes can be divided into:

- supervised
- unsupervised (self-organizing) and
- stochastic strategies.

In the case of supervised learning, in addition to the input patterns, the desired corresponding output patterns are also presented to the network in the training phase. The network calculates a current output from the input pattern, and this current output is compared with the desired output. An error signal is obtained from the difference between the generated and the required output. This signal is then employed to modify the weights in accordance with the current learning rule, as a result of which the error signal is reduced. The best-known and most commonly employed network model here is the multilayer perceptron with backpropagation learning rule [Rumelhart, McClelland 1986].

In the case of unsupervised learning, the network is required to find classification criteria for the input patterns independently. The network attempts to discover common features among the presented input patterns via a "similarity comparison", and to adapt its weight structure accordingly. The neurons thus form independent pattern classes and become pattern detectors. This method is similar in effect to clustering algorithms or vector quantification methods. An example of this process is provided by Kohonen's self-organizing feature maps [Kohonen 1989], which organize themselves with the aim of converting signal similarity into proximity between excited neurons.

Stochastic learning methods employ random processes and probability distributions to minimize a suitably defined energy function of the network. This process corresponds to that of crystal growth: in order to obtain a crystal with the minimum errors in its lattice structure, it must be cooled very slowly, to ensure that the molecules have sufficient time to find those positions at which the total energy of the structure becomes minimal. As long as the temperature remains sufficiently high, individual molecules can change positions, resulting in an increase in the total energy. This may result in departures from local minima. As the temperature falls, the probability of this occurring diminishes. Examples of this process, which by analogy with thermodynamics is also referred to as simulated annealing, are Boltzmann machines [Köhle 1990].

A large number of neural models now exist, and each of these models is available in various forms. A rough classification is shown in the figure below. By way of example, the network types which are most commonly employed in practice are explained below: the multilayer perceptron with backpropagation learning rule as a network model employing supervised learning, and Kohonen's self-organizing feature maps as a network with unsupervised learning.

The framework is designed the development of generic cooperative running processes. Artificial Neural Networks (ANNs) are implementable inside the framework defining the algorithms needed to perform artificial neurons and maps processes. The versatility of the framework gives the possibility to implement different and cooperating models. Multi Layer Perceptrons (MLPs), Self Organizing Maps (SOMs) and Kohonen SOMs (KSOMs) will be discussed using the integrate and fire neuron model. A short term associative memory architecture will be shown using the leabra neuron model. An architecture able to gain time-space correlation on input signals will be showed using the Izhikevich neuron model obtained as an optimization of the Hodgkin and Huxley model. The framework implementation details for each of these architecture will be discussed and example applications will be showed.

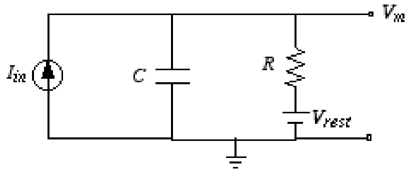


Figure 4.1: The integrate-and-fire artificial neuron model: equivalent electric schema

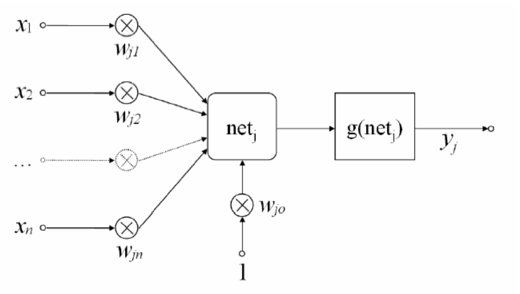


Figure 4.2: The integrate-and-fire artificial neuron model: computational schema

4.1 The integrate and fire model implementation: IFNeuron, IFNeuronConnectionSpec and IFNeuronGroup

The integrate-and-fire model is the simplest model of a spiking neuron that takes into account the dynamics of the input. The basis of the integrate-and-fire model is the simple compartmental model of a neuron [ ]. The equivalent electric schema is showed in Fig. 4.1.

The computational implementation of the integrate and fire model follows the schema showed in Fig. 4.2.

An *IFNeuron* structure has been implement in the framework as a running

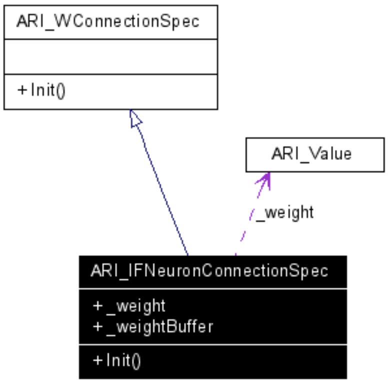


Figure 4.3: The IFNeuronConnectionSpec structure

process directly deriving from the *W* structure. Template arguments have been specialised to obtain an *OUTDATA* as a real number (double precision floating point value) and a *CONNSPEC* as a *IFNeuronConnectionSpec* structure, which is shown in Fig. 4.3.

The connection structure for such a process uses a real number to manage the synaptic weight. The value may be initialised by the user or randomly chosen by the framework according to the value initialisation parameters. A weight buffer value is needed for internal operations during supervised learning using the multi layer perceptron process, which will be discussed later. The *IFNeuron* structure defines the private virtual method *Process* in order to perform the weighted sum of signal coming from input connections. The result value is then filtered using the sigmoid function according to the integrate and fire model. The structure internally saves a value to speed up the delta-rule algorithm adopted during supervised learning. The I/O buffering operations simply manage internal members and recall the base class methods. The rendering function provides the graphic visualisation of the soma and of the input connections. The architecture is shown in Fig. 4.4.

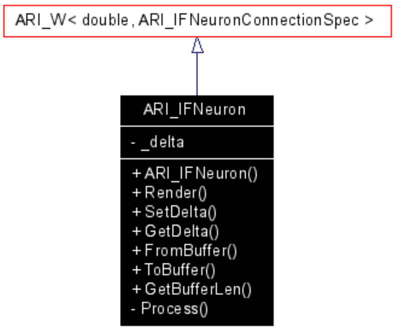


Figure 4.4: Architecture of IFNeuron structure

The *IFNeuronGroup* structure, which represents a group of *IFNeurons*, has been derived from the *WGroup* base structure. The *IFNeuronGroup* structure will be used by high-level processes in order to perform supervised and unsupervised learning tasks based on the integrate and fire neuron model.

4.1.1 Multi layer perceptron (MLP)

A multi level network is realised by several units which are organised in a layered structure. The information flow through the layers connections, without any connection inside the same layer. The layers other than input and output ones are known as hidden layers. A general scheme is showed in Fig. 4.5, where output layer represents the neural network's output. Each layer is fully connected to the next one and such a structure is known as Multi Layer Perceptron (MLP).

According to the integrate and fire neuron model, the connections are characterised only by the synaptic weights, while the neuron process model consists in the non linear filtering of the weighted sum of input signals. In 1989 Hornik, Stinchcombe and White showed the ability of such a structure to approximate a math function with a given precision a value, depending on the number of hidden units and layers. In particular a MLP is able to separate convex regions

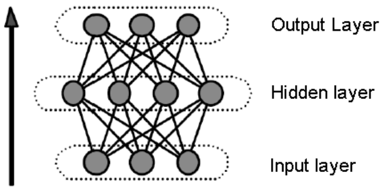


Figure 4.5: Multilayer perceptron schema

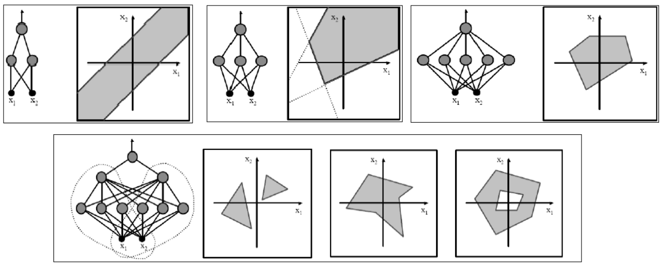


Figure 4.6: MLP structures and corresponding region separation on input space

with a number of sides less or equal to the number of hidden units, as it is shown in Fig. 4.6.

In a MLP it is important to note that, using linear activation functions, an N-layers network is always referable to a 2-layers network, as it is show in Fig. 4.7.

4.1.1.1 MLP learning

The rule of a supervised learning shows how the connection weights should be changed in order to reduce the error between the activation levels of the network and the target activation levels. In a MLP target levels are backpropagated from output layers to hidden layers using different techniques. A typical strategy consists in changing synaptic weights following the gradient decay. For the

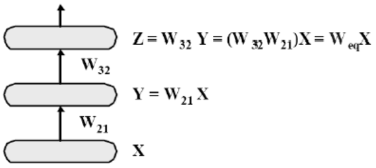


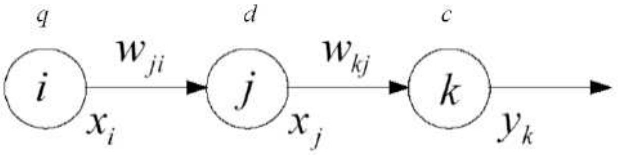
Figure 4.7: A N-layers MLP using linear activation function is equivalent to a 2-layers MLP

generic input connection j of unit i, the weight change in response to an error E between output and target, we have:

$$\triangle w_{ij} \propto \frac{\partial E}{\partial w_{ij}}$$

This supervised learning strategy is known as backpropagation of error learning algorithm (generalised delta rule) and it was thought up by Rumelhart, Hinton and Williams on 1985. In this context the learning process consists in a set of desired associations $(x_k; t_k)$ on a training set (TS), where $x_k$is the network output and $t_k$is the network target regarding to the k-th example. The algorithm convergence consists in reducing the global error E while weights are changing, in order to obtain $E < \varepsilon$. The network generalisation property consist in the ability to obtain $E < \varepsilon$even on examples not shown in the TS. The MLP is able to train itself by propagating the resulting error backward following the back-propagation algorithm [59]. The learning process is obtained through propagation and backpropagation steps, as it is shown in Fig. 4.8.

Let us consider a MLP with three layers with the convention showed below:



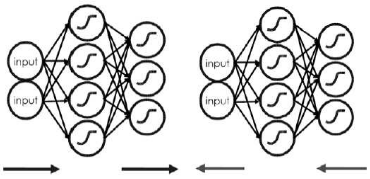


Figure 4.8: Propagation and backpropagation steps in a MLP

The number of neurons of input, hidden and output layers is respectively q, d and c. The generic neuron i of input layer has an output x_i; the generic neuron j of hidden layer has an output x_j; the generic neuron k of output layer has an output y_k, while its target is t_k. We define the average squared error on k-th input pattern:

$$E_k = \frac{1}{2}\sum_{i=1}^{c}(y_{ki} - t_{ki})^2$$

The global error in respect of all patterns in the TS is:

$$E = \sum_{k=1}^{M} E_k$$

For output and hidden layers we have:

$$\text{output layer} \begin{cases} y_k = g(net_k) \\ net_k = w_{ko} + \sum_{j=1}^{d} w_{ki}x_j \end{cases} \qquad \text{hidden layer} \begin{cases} x_j = g(net_j) \\ net_j = w_{jo} + \sum_{i=1}^{q} w_{ji}x_i \end{cases}$$

Delta values for output layer are:

$$\delta_k = \frac{\partial E_k}{\partial \mathrm{net}_k} = \frac{\partial}{\partial \mathrm{net}_k}\left(\frac{1}{2}\sum_{i=1}^{c}\left(y_{ki} - t_{ki}\right)^2\right) = \frac{1}{2}\frac{\partial\left(y_k - t_k\right)^2}{\partial \mathrm{net}_k} =$$

$$= \left(y_k - t_k\right)\frac{\partial\left(y_k - t_k\right)}{\partial \mathrm{net}_k} = \left(y_k - t_k\right)\frac{\partial y_k}{\partial \mathrm{net}_k} =$$

$$= \left(y_k - t_k\right)\frac{\partial g\left(net_k\right)}{\partial \mathrm{net}_k} = \left(y_k - t_k\right)g'\left(net_k\right)$$

Delta values for hidden layers are:

$$\delta_j = \frac{\partial E_k}{\partial \mathrm{net}_j} = \frac{\partial E_k}{\partial \mathrm{x}_j}\frac{\partial x_j}{\partial \mathrm{net}_j} = \frac{\partial E_k}{\partial \mathrm{x}_j}g'(net_j) = g'(net_j)\left[\sum_{k=1}^{c}\frac{\partial E_k}{\partial \mathrm{net}_k}\frac{\partial \mathrm{net}_k}{\partial x_j}\right] =$$

$$= g'(net_j)\left[\sum_{k=1}^{c}\delta_k\frac{\partial}{\partial x_j}\left(w_{ko} + \sum_{j=1}^{d}w_{kj}x_j\right)\right] = g'(net_j)\sum_{k=1}^{c}\delta_k w_{kj}$$

$$\begin{cases} f(x) = f\left(g_1(x), g_2(x), \ldots, g_n(x)\right) \\ \dfrac{\partial f}{\partial x} = \displaystyle\sum_{i=1}^{n}\frac{\partial f}{\partial g_i}\frac{\partial g_i}{\partial x} \end{cases} \implies \delta_j = g'(net_j)\sum_{k=1}^{c}\delta_k w_{kj}$$

The error gradient in respect to the generic weight w_{kj} is:

$$\frac{\partial E_k}{\partial w_{kj}} = \frac{\partial E_k}{\partial \mathrm{net}_k}\frac{\partial \mathrm{net}_k}{\partial w_{kj}} = \delta_k\frac{\partial \mathrm{net}_k}{\partial w_{kj}} =$$

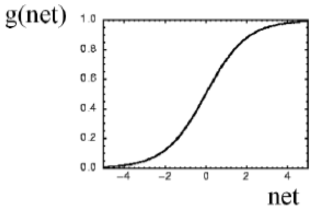
$$= \delta_k\frac{\partial}{\partial w_{kj}}\left(w_{ko} + \sum_{j=1}^{d}w_{kj}x_j\right) = \delta_k\, x_j$$

Weight variations result in:

$$\begin{aligned} \Delta w_{kj} &= \eta \frac{\partial E_k}{\partial w_{kj}} &= \eta \delta_k x_j &= \eta (y_k - t_k) g'(net_k) x_j \\ \Delta w_{ji} &= \eta \frac{\partial E_k}{\partial w_{ji}} &= \eta \delta_j x_i &= \eta x_i g'(net_j) \sum_{k=1}^{c} \delta_k w_{kj} \end{aligned}$$

where $\eta \subset [0;1]$ represents the learning rate value. Final delta values are obtained taking into account the derivative of the actuation function:

$$g(net_j) = \frac{1}{1 + e^{-\gamma net_j}}$$



$$g'(net_j) = \frac{\partial g(net_j)}{\partial net_j} = g(net_j)\left[1 - g(net_j)\right]$$

$$\begin{cases} \delta_k = (y_k - t_k) y_k (1 - y_k) \\ \delta_j = x_j (1 - x_j) \sum_{k=1}^{c} \delta_k w_{kj} \end{cases}$$

It is to be noticed that the activation function must be a non linear and derivable function. Weight variations have to be subtracted to original weights in order to perform an error minimization, as it is shown in Fig. 4.9.

Since $\Delta w_{ji} \propto g'(net_j)$ and considering that the sigmoid derivative function has maximal values for $y_j = 0.5$ and minimal values for $y_j = 0$ and $y_j = 1$, weights variation is greater for those neuron having a medium activation value and it is negligible for those neurons having a high or low activation value. This implies the stability of the learning process, and it indicates how weights initialisation has to be performed with small values. A learning rate decay is often useful in order to speed up the learning process. Moreover a low-pass filter

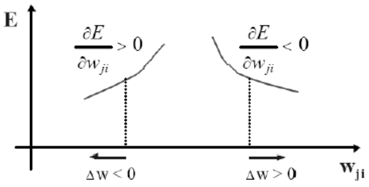

E
$\frac{\partial E}{\partial w_{ji}} > 0$
$\frac{\partial E}{\partial w_{ji}} < 0$
Δw < 0
Δw > 0
w_{ji}

Figure 4.9: Error minimization causes weights variation to have opposite sign in respect to error derivative function

is applied using a term called momentum ($\gamma \subset [0;1]$) which is useful to minimize the process oscillations. The value δ_i for each neuron is finally calculated as follows:

$$\delta_i = \begin{cases} f'(I_i) \bullet (z_i - o_i) \mid i \in neurons_{output} \\ f'(I_i) \bullet \sum_j (\delta_j \bullet w_{ij}) \mid i \notin neurons_{output} \end{cases}$$

Taking into account both the learning rate value and the momentum value, the adaptation of the weights results in:

$$\Delta w_{ij}(t) = \alpha(t)\delta_i x_j + \gamma(t)\Delta w_{ij}(t-1)$$

where:

- z_i the components are the target output values
- o_i is the output value of the neuron i
- I is the weighted sum of all signals which are active at the input connections
- f' is the derivative of the activation sigmoid function f used to compute the output
- w_{ij} is the weight of the connection between neurons i and j.

- $\alpha(t)$ is the learning rate with a decay factor function of epoch t.
- $\gamma(t)$ is the momentum with a decay factor function of epoch t.
- the x_j components represent the signals at the connection concerned.

4.1.1.2 MLP implementation into the framework: IFMLP

The IFMLP structure implements the supervised learning task based on the delta-rule algorithm with learning rate decay and momentum decay. The structure is initialised by the user specifying the input pattern map, the target pattern map and the maps related to the hidden layers, the bias unit and the output layers. The input and the target data may be provided using the *SensorDriver* structures or using other maps already registered into the framework by the user. The structure, which is directly derived from *3DObject* structure, provides the learning parameters initialisation and automatically sets the *Update* flags. Since the *SetInput* and the *Update* operations are overridden, the corresponding flags are removed from the hidden layers, from the output layer and finally from the bias unit, whose output is forced to a value equal to 1.0. Layers are the fully connected and the bias unit is connected to all the hidden layers and to the output layer and synaptic weight are initialised in the range [0.0 ; 0.1]. During the Update process, the signals are propagated according to the integrate and fire model for each unit. The backpropagation algorithm is applied and the delta values are calculated and stored internally. Finally, the synaptic weights are updated according to the actual values of learning rate and momentum. The *IFMLP* structure provides functions to perform learning parameters decay, making the user able to apply the appropriate decay strategy for such parameters. The I/O buffering operations are defined in order to manage the whole state of the multi layer perceptron. The architecture is shown in Fig. 4.10.

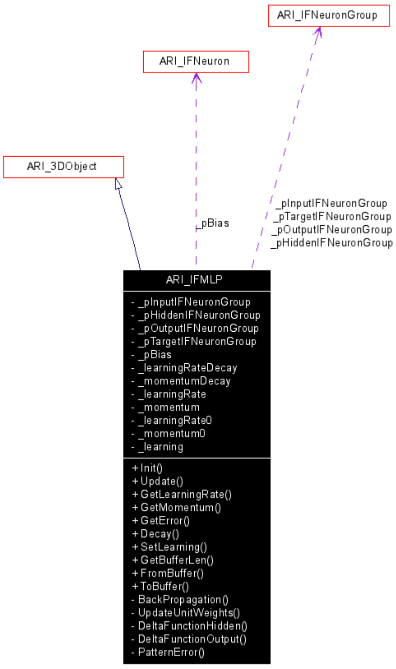


Figure 4.10: The IFMLP structure

4.1.2 Self Organizing Map (SOM)

A SOM structure has been implemented in the framework using the *IFNeuronGroup* structure as the base class. The derived *IFSOM* class contains several extended methods in order to perform the initialization, learning and test tasks. During the initialization step the two-dimensional plane of the map is specified and an auxliary neuron group is needed to store the output classification values.

In the SOM units, all weight elements $w_{ij}$of a generic neuron i at time t, for the input vector e, are modified as follows [52]:

$$w_{ij}(t) = w_{ij}(t-1) + \alpha r_{iz}(t)[e_j - w_{ij}(t-1)]$$

where:

- the neuron that has the minimum distance $d = min_i \parallel e - w_i \parallel$ from the input vector e is the winning unit z.
- $\alpha(t) = f_\alpha \alpha(t-1)$ is the learning rate of epoch t and f_α is the learning rate factor.
- $r_{iz}(t) = e^{-\frac{d^2}{\sigma^2}}$ is the feedback function of neuron i to the winning neuron z of epoch t.
- $\sigma(t) = f_\sigma \sigma(t-1)$ is the learning radius of epoch t and f_σ is the learning radius factor.

The I/O buffering operations are overriden to manage all the needed parameters. The virtual *SetInput* and the *Update* methods contains the implementation of the learning and test tasks. At each step of the framework process, the winning neuron is found according to the minimum distance between the input vector and the weights vector. Learning values decay is performed at each step and, while the learning status is active, the synaptic weights are modified as it is showed above. The structure is showed in Fig. 4.11.

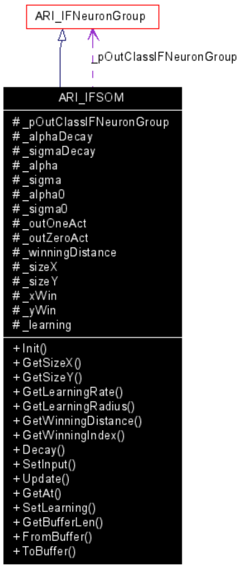


Figure 4.11: The IFSOM structure

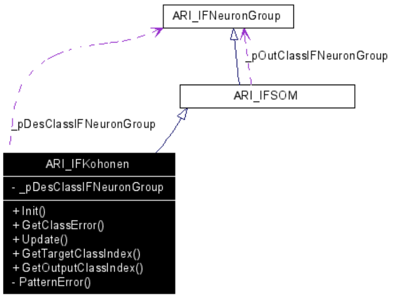


Figure 4.12: The IFKohonen structure

4.1.2.1 Kohonen Self Organizing Map (KSOM)

The KSOM structure is implemented in the framwork as a child of the SOM structure. The IFKohonen structure is quite similar to the IFSOM structure, since it has the same learning strategy. During the training step it can perform a labeling task thanks to the information about the desired output classification. The few differences of the IFKohonen structure are specified in the *Init* and in the *Update* methods, while some extended methods are provided to manage the error in respect of the desired classification. During the initialization step a neuron group is needed to store the information about the desired classification. The labeling process is performed during the *Update* step. The structure is shown in Fig. 4.12.

4.1.3 Case Study: Real-Time Facial Expression Recognition

Automatic classification of human facial expressions is a topical challenge undergoing constant improvement. Fields of interest range human behavioural modelling, social robot development to man-machine interfaces and risk assessment in social environments. The attention is currently focused on the development of biomimetic materials, structures and control algorithms that can acquire and replicate certain functions and responses of living systems. In this paper we present a hierarchical neural network architecture for automatic recognition of facial expressions. Features are extracted by a first layer of classifiers consisting of four Kohonen self organising maps. Data converge in a predictive layer consisting of a multilayer perceptron which performs the final classification. This architecture was applied to a wide experimental dataset collected from different human subjects. Results compared with panel tests support the architecture developed.

4.1.3.1 Introduction

René Descartes was the first to suggest that facial dynamics can be considered non-verbal expression of emotions and moods. In the nineteenth century, Charles Darwin was the first to think of facial and emotional expression as adaptive, subject to evolution and necessary for non-verbal communication of human emotional states. In the twentieth century, anatomical studies were added to philosophical, behavioural, biological and physiological studies of facial expressions.

In the last few decades, several systems for automatic analysis, interpretation and categorisation of some basic human facial expressions have been developed [61, 50, 51]. These systems are characterised by different algorithms and they show a common architecture [47, 61]. In the first step data acquired from a human subject are analysed by some pre-processing tasks in order to localize and normalize the face. Subsequently some landmarks or markers are found in the facial image and specific algorithms are used to perform a feature extraction and analysis.

This can be achieved through several techniques such as optical flow [42, 51, 46], the evaluation of the geometric distance among landmarks [43], the scalar quantization of facial dynamics [43], or the evaluation of muscular activation parameters [48, 45]. In the last step, classification can be performed by applying techniques such as Markov models [51], artificial neural networks [53, 41, 62, 64], k-nearest neighbourhood algorithms or expert systems [43]. In the 1970s, Ekman's research culminated in a system for the detailed description of facial expressions (FACS, Facial Action Coding System), which was an important starting point in the exploration of human visual communication [47, 48]. Ekman developed the concept of Action Unit (AU) to describe the action of one or more facial muscles. A FACS coder decomposes an observed facial expression into a combination of AUs. A FACS decoder maps a combination of AUs into the corresponding facial expression.

Some facial expression recognition systems [61, 51] use a three-dimensional wireframe model of the face combined with a physics-based model of the skin and a model of the facial muscles. The deformation of skin and muscles are estimated and classification of facial expression is performed by a FACS decoder. This approach implies the need to deduce dynamically the deformations of involved muscles, requiring a considerable computational effort.

Here it is described the facial expression recognition approach adopted in the FACE project [56] currently under development at the University of Pisa. The objective of the project is the realisation of an anthropomorphic social robot for man-machine non-verbal communication. It is based on the simulation of biological behaviour using materials, structures and control algorithms that can acquire and replicate certain functions and responses of living systems.

The underlying principle of FACE's automatic facial expression recognition is based on Darwin's idea of adaptability and on the importance of experience in neurological processes. In this study an automatic system based on a Hierarchical Neural Network has been developed. Landmarks are selected in order to seek and identify four facial zones (right eye, left eye, nose, mouth). Features are extracted by a first layer of classifiers consisting of four Kohonen Self Organising Maps (KSOM) [53], one for each facial zone being examined. Classification is finally carried out by a second layer consisting of a Multilayer Perceptron (MLP) [41].

4.1.3.2 Facial expression estimator

Studies carried out over the past ten years [46] have demonstrated that there are specialized cells in the monkey brain for decoding messages transmitted by facial expressions. Subsequent studies [45, 62] have shown that human beings, whose brains are extremely similar to those of monkeys as far as the nervous components involved in emotional behaviour are concerned, also react to human faces that have assumed various emotional expressions, and, more generally, to an image of a human face that is recognized by its typical conformation based on the presence of the eyes-nose-mouth tern. A group of brain cells of the amygdala is able to communicate to an infant that two points, a small vertical strip and a small half-moon (eyes, nose and mouth) are a positive stimulus for its well-being and its survival [64]: the infant responds to a conformation of this type with a smile, a response that is for the most part instinctive, that inspires affection and attachment in adults.

The system here proposed, called Facial Expression Estimator (FEE), is based on this principle (Fig. 2.4). It can be divided in a pre-processing, processing and post-processing module. In the pre-processing step, once an input pattern has been received, some markers (marker locater) are located in order to identify the eyes, nose, mouth tern (zone locater). This information is used to split the facial image into four different patterns (structure splitter). The pre-processing step is needed to set the face correctly and to perform an initial reduction of input data to be sent to the processing module. The processing module consists of a Hierarchical Neural Network (HNN) realised by a first layer of parallel classifiers, one for each zone, whose outputs are analysed by a final classifier.

The FEE receives the patterns supplied by a CCD video camera. As shown in Table 4.1, each frame consists of a matrix of dimensions $W_0 x H_0$, whose elements $c_{x,y} \in [c_{min}; c_{max}] \subset N$, $x \in [0; W_0 - 1] \subset N$, $y \in [0; H - 1] \subset N$, $c_{min} < c_{max}$, are scalar and represent the 8-bit gray-scale pixels of the facial image.

Description	Variable	CCD Video camera
Dimension of the FEE input matrix	$W_0 x H_0$	$480x640$
Range of the acquisition system	$[c_{min}; c_{max}]$	$[0; 255]$

Table 4.1: Characteristic variables of the acquisition systems

For each matrix four markers are located by using eye tracking algorithms and facial metrology techniques [55]. Eye positions are used to evaluate the relative facial roll angle. A coordinate transformation is applied to each matrix in order to reset the facial roll angle. The line joining the eyes is calculated and the right corner of the right eye, the left corner of the left eye, the tip of the nose and the point below the lower lip are identified. Thus, right eye, left eye, nose and mouth zones are located. The algorithm revealed its robustness to evaluation errors of marker positions within 10% of the Euclidean distance between the outer corners of the eyes. Coordinate transformation allows the four zones to be located by means of four rectangles identified by the coordinates of the upper left vertex $(x_{k,left}, y_{k,top})$and the lower right vertex $(x_{k,right}, y_{k,bottom})$, with $k = 1...4$.

Once the vertexes are identified, the *k-th* zone pattern is:

$$c_{k,x,y} \mid x \in [x_{k,left}; x_{k,right}], y \in [y_{k,top}; y_{k,bottom}], k = 1...4$$

The zone patterns are processed by a linear filter in order to reduce their dimensions. The *k-th* zone is reduced to a dimension of $W_k x H_k$. Applying the linear filter, the *k-th* zone pattern becomes:

$$c'_{k,x,y} = \frac{1}{\Delta y_k \bullet \Delta x_k} \sum_{j=0}^{\Delta y_k - 1} \sum_{i=0}^{\Delta x_k - 1} c_{k, x_{k,left} + x \bullet \Delta x_k, y_{k,bottom} + y \bullet \Delta y_k + j} \mid$$

$$x \in [0; W_k - 1], y \in [0; H_k - 1], k = 1...4$$

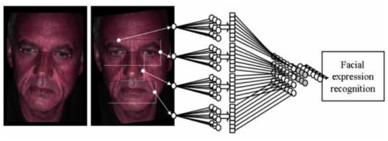


Figure 4.13: Architecture of the two classification levels created with the hierarchical neural network.

$$\Delta x_k = \frac{x_{k,right} - x_{k,left}}{W_k}, \Delta y_k = \frac{y_{k,top} - y_{k,bottom}}{H_k}, k = 1...4$$

For each facial image the values $c'_{k,x,y}$, with $k = 1...4$, represent the four input matrices for the processing module

4.1.3.3 Processing: The hierarchical neural network

The processing module consists of a two-layer hierarchical neural network [49, 60, 44]. The first layer is a parallel classifier composed of two-dimensional KSOMs [54] that receives the $c'_{k,x,y}$ input data from the pre-processing unit. The analysis of these data permits a pre-classification by means of extraction of features. The outputs of the first layer are sent to the second layer of the network, realised by a one-hidden-layer MLP, whose task is to combine the inputs properly and perform the final classification. The network architecture is showed in Fig. 4.13.

The response of each KSOM is a seven-element boolean vector obtained by a labeling phase. The positive value of the *i-th* element of the *k-th* vector means that the k-th zone has been pre-classified as belonging to the i-th facial expression.

For each facial expression to be classified, the input to the MLP is a 28-elements vector containing the four seven-elements vectors computed by the first layer. Each element of the input vector is connected to one neuron of the input layer of the MLP.

Each KSOM performs an unsupervised pre-classification separating the input data into crisp classes. The outputs of this first layer are used to form the input pattern for the second layer, which refines the classification and attributes the facial expression to a specific group. The implementation by means of two independent layers implies rapid and efficient training.

The first layer is composed of a set of four classifiers. Each input matrix is the input to only one classifier of the first module allowing each KSOM to act as an independent classification unit. In this way, each KSOM can be singularly optimised without affecting the other modules of the architecture. This allows the computational complexity to be reduced locally and a flexible system to be implemented.

The MLP output represents the classification of the acquired facial expression in seven crisp classes. The KSOMs output converges to the MLP whose outputs are equal to the number of facial expressions to be recognised (i.e. seven). In order to recognize an unknown pattern each component of the seven-dimensional vector of MLP output is sent to a *Threshold* function.

$$Threshold(x, t_{low}, t_{high}) = \begin{cases} no \mid x < t_{low} \\ yes \mid x \geq t_{high} \\ donotknow \mid t_{low} \leq x < t_{high} \end{cases}$$

The result consists of a seven-dimensional vector whose positive value of the *i-th* element indicates the pattern has been classified as belonging to the *i-th* facial expression. It can be observed that the resulting vector may indicate more than one expression or no expression. In other words the HNN result may indicate a pattern was classified, or unclassified, or misclassified, or that the pattern caused a confusion in the net.

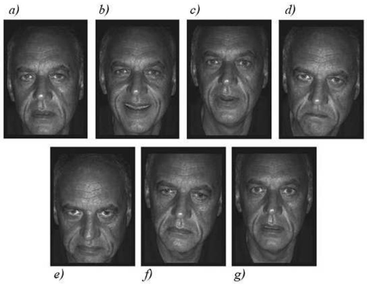


Figure 4.14: Facial expressions of: a) neutrality; b) happiness; c) surprise; d) anger; e) disgust; f) sadness; g) fear

4.1.3.4 Results and discussion

The FEE has been used to classify the seven basic facial expressions [47, 48]. We trained and tested it with two datasets collected in two different experimental campaigns. In the first experimental campaign, we collected a single measurement for each expression on a selected subject. The subject was asked to express in turn: neutrality, happiness, surprise, anger, disgust, sadness and fear. Acquired expressions are shown in Fig. 4.14.

To simulate a real situation, each pattern was corrupted by translating, rotating and scaling the original one by adding a gaussian noise. This procedure artificially created new patterns with different positions, orientations and scale factors. In order to realise a dataset, 200 different images for each expression with a total of 1400 patterns were generated. In order to check the generalization capability of the HNN, the dataset was randomly partitioned in two equal-sized folds. The first one was used to train the HNN (training set), and the second one to test it (test set).

For the KSOMs, we fixed $\alpha(t) = 0.999$, $f_\alpha = 0.99$, $\sigma(t) = 5$, $f_\sigma = 0.995$ and training epochs equal to 500; for the MLP, $\alpha(t) = 0.999$, with a decay of 0.999, $\gamma(t) = 0.99$, with a decay of 0.999. In order to select the appropriate network architecture, we experimentally proceeded to train and test the HNN

Conf.	**Right eye**	**Left eye**	**Nose**	**Mouth**	**1st level**	**2nd level**
number	**map**	**map**	**map**	**map**	**(neurons)**	**(IL|HL|OL)**
	W_1xH_1	W_1xH_1	W_1xH_1	W_1xH_1	$KSOM_{1...4}$	MLP
1	15x15	15x15	15x20	30x10	3x3	28|5|7
2	15x15	15x15	15x20	30x10	3x3	28|10|7
3	15x15	15x15	15x20	30x10	3x3	28|15|7
4	15x15	15x15	15x20	30x10	7x7	28|5|7
5	15x15	15x15	15x20	30x10	7x7	28|10|7
6	15x15	15x15	15x20	30x10	7x7	28|15|7
7	15x15	15x15	15x20	30x10	10x10	28|5|7
8	15x15	15x15	15x20	30x10	10x10	28|10|7
9	15x15	15x15	15x20	30x10	10x10	28|15|7
10	25x25	25x25	25x30	40x20	3x3	28|5|7
11	25x25	25x25	25x30	40x20	3x3	28|10|7
12	25x25	25x25	25x30	40x20	3x3	28|15|7
13	25x25	25x25	25x30	40x20	7x7	28|5|7
14	25x25	25x25	25x30	40x20	7x7	28|10|7
15	25x25	25x25	25x30	40x20	7x7	28|15|7
16	25x25	25x25	25x30	40x20	10x10	28|5|7
17	25x25	25x25	25x30	40x20	10x10	28|10|7
18	25x25	25x25	25x30	40x20	10x10	28|15|7

Table 4.2: Tested configurations of the HNN

in the configurations shown in Table 4.2.

Results were summarized by means of the confusion matrix which shows the degree of the achieved classification. The confusion matrix possesses a number of both rows and columns equal to the number of classes to be recognized. The generic element *r(i,j)* represents the degree of recognition of class i as belonging to class j. A more diagonal confusion matrix corresponds to a higher degree of classification. The *Threshold* function supplies an additional column to the confusion matrix. In Fig. 4.15, the confusion matrix diagonals, with $t_{low} = 0.3$ and $t_{high} = 0.5$, are shown for each of the 18 configurations.

Five expressions out of seven were well-classified in all the configurations. The system fails in the recognition of fear and surprise in configurations 1, 2,

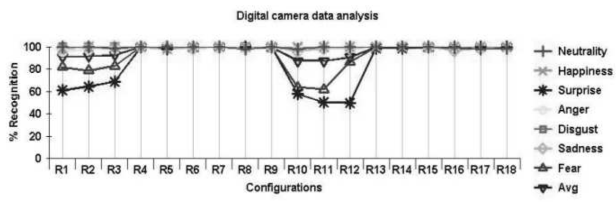


Figure 4.15: 1st experiment. Chart of facial expression recognition percentage obtained with input patterns from the digital camera ($t_{low} = 0.3$, $t_{high} = 0.5$)

3, 10, 11, 12, where the KSOMs have minimal dimensions. Consequently, in spite of different dimensions of the hidden MLP layer, we can suppose they are not able to pre-classify the input pattern, compromising the overall result. But here, the slight resemblance in fear and surprise (Fig. 4.14) could fool even an experienced eye.

In the second experimental campaign, 22 measurements for each of the seven facial expressions were performed on the subject, obtaining a basic dataset of 154 digital camera acquisitions. In order to avoid variations due to surroundings, measurements were carried out with the same lighting conditions. A neutral expression followed each new expression, and the entire cycle was randomly repeated for all the seven expressions. In order to simulate a real situation and to check the generalization capability of the HNN the 154 measurements were partitioned in two equal-sized training and test sets. Each pattern was "cloned" 100 times and corrupted by translating, rotating and scaling the original one by adding a gaussian noise. A total of 7700 images for both training and test sets were generated. In Table 4.3 and Fig. 4.16, the 18 diagonals of the 18 confusion matrices produced by the FEE in response to the 7700 test images with and are shown.

In this case as well, we can observer a worse system performance for the HNNs with 3x3 KSOMs (configurations 1, 2, 3, 10, 11, 12). The 17th configuration shows the best performance; the confusion matrix is reported in Table

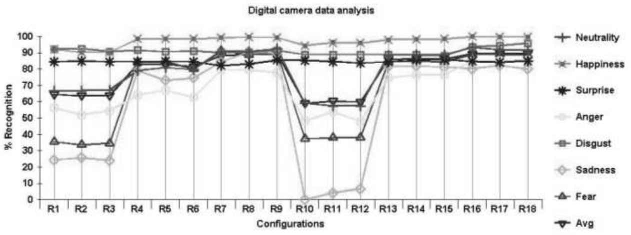


Figure 4.16: 2nd experiment. Chart of facial expression recognition percentage obtained with input patterns from the digital camera ($t_{low} = 0.5$, $t_{high} = 0.5$)

Conf. **number**	**Neutr.**	**Happ.**	**Surpr.**	**Ang.**	**Disg.**	**Sad.**	**Fear**	**Avg**
1	66.5	92.0	84.5	56.2	92.2	24.4	35.5	**64.5**
2	66.8	90.2	84.9	52.0	92.5	25.6	33.6	**63.7**
3	67.2	90.3	84.5	54.5	90.7	24.1	34.6	**63.7**
4	79.1	98.5	84.5	64.0	91.5	79.0	84.3	**83.0**
5	80.7	98.4	84.6	67.0	90.5	73.1	84.5	**82.7**
6	79.7	98.5	84.5	62.6	91.1	74.5	80.0	**81.6**
7	90.7	99.1	82.1	79.4	90.1	84.4	91.2	**88.1**
8	90.7	99.6	83.1	79.5	90.4	90.5	88.9	**89.0**
9	92.4	99.3	85.4	77.8	91.4	89.6	89.8	**89.4**
10	59.0	94.3	85.3	48.2	88.7	0.0	37.4	**59.0**
11	57.4	96.2	84.5	53.8	88.9	4.2	38.0	**60.4**
12	57.5	95.9	83.7	47.9	88.7	6.7	38.2	**59.8**
13	88.5	97.9	84.5	74.6	88.7	81.5	84.4	**85.7**
14	88.7	98.1	85.4	76.4	88.6	81.9	85.0	**86.3**
15	88.7	98.5	85.8	76.5	88.0	81.2	84.3	**86.1**
16	93.1	99.8	84.6	84.7	93.6	80.2	88.8	**89.3**
17	91.8	99.6	84.2	85.0	94.4	82.2	89.0	**89.5**
18	91.5	99.6	84.8	85.6	95.7	79.9	88.4	**89.4**

Table 4.3: 2nd experiment. Matrix of the diagonals of the 18 configurations ($t_{low} = 0.5$, $t_{high} = 0.5$)

	Neutr.	Happ.	Surpr.	Ang.	Disg.	Sad.	Fear
Neutrality	**91.8**	0.0	0.0	0.0	0.1	5.8	0.5
Happiness	0.2	**99.6**	0.0	0.0	0.3	0.0	0.0
Surprise	0.0	0.3	**84.2**	0.0	0.0	0.0	15.4
Anger	0.0	0.0	0.0	**85.0**	7.4	5.5	1.2
Disgust	0.0	0.0	0.0	1.3	**94.4**	0.5	2.5
Sadness	0.4	0.0	0.0	14.5	1.1	**82.2**	2.2
Fear	0.0	0.2	1.7	3.7	1.2	2.3	**89.0**

Table 4.4: Confusion matrix of configuration 17 ($t_{low} = 0.5$, $t_{high} = 0.5$)

	Neutr.	Happ.	Surpr.	Ang.	Disg.	Sad.	Fear	DON'T KNOW
Neutrality	**88.5**	0.0	0.0	0.0	0.0	5.3	0.1	0.8
Happiness	0.2	**99.5**	0.0	0.0	0.1	0.0	0.0	0.0
Surprise	0.0	0.0	**82.7**	0.0	0.0	0.0	15.1	0.6
Anger	0.0	0.0	0.0	**80.3**	5.5	3.8	1.1	2.0
Disgust	0.0	0.0	0.0	0.9	**91.2**	0.5	1.9	1.4
Sadness	0.2	0.0	0.0	2.7	0.6	**72.8**	2.0	6.2
Fear	0.0	0.2	0.8	0.9	0.6	2.0	**87.7**	1.9

Table 4.5: Confusion matrix of configuration number 17 ($t_{low} = 0.3$ and $t_{high} = 0.7$)

4.4. The average recognition percentage was 89.45%.

Since each input pattern may be classified as belonging to more than one facial expression, or it may be unclassified (i.e. as belonging to none of the seven facial expressions), the sum of each row of the confusion matrix may not be equal to 100. Setting the *Threshold* parameters to $t_{low} = 0.3$ and $t_{high} = 0.7$ in the 17th configuration, percentages of unclassification and misclassification are transferred to the indecision column (i.e. don't know) (Table 4.5), resulting in a minor reduction of the average recognition percentage (86.1%).

Interpretation of facial expressions is a particularly subjective process [63] and literature offers no rigorous standards for human facial expressivity classification. This can be evidenced by the existence of numerous human face

	Neutr.	Happ.	Surpr.	Ang.	Disg.	Sad.	Fear
Neutrality	**97.7**	0.0	0.0	0.0	0.0	2.3	0.0
Happiness	0.0	**100.0**	0.0	0.0	0.0	0.0	0.0
Surprise	0.0	3.8	**87.9**	0.0	0.0	0.0	8.3
Anger	0.0	0.0	0.0	**85.6**	3.0	9.8	1.5
Disgust	0.8	0.0	0.0	0.8	**91.7**	6.1	0.8
Sadness	22.7	0.0	0.0	5.3	0.8	**71.2**	0.0
Fear	0.0	0.0	18.2	0.8	1.5	6.1	**73.5**

Table 4.6: Confusion matrix of panelists responses.

databases characterized by heterogeneous structure and developed for very specific advisory and didactic applications. The language of facial expressions possesses an information content that, in large part, is lost in an abstract and static context. In this situation, the possibility of having a great number of examples, i.e. training and experience, aids in the classification of facial expression.

For these reasons, a human panel test assessed the dataset patterns and HNN results were compared with their judgments. Twelve untrained panelists were interviewed. Following the method adopted for FEE, a training phase and a test phase were carried out. In the training phase the 77 images of the training set extracted from the basic dataset were presented to each panelists. Panelists were asked to observe the face and to make a judgement on the presented facial expression. The panelist was then shown the correct response. In the test phase the 77 images of the test set extracted from the basic dataset were presented to the panelists. Each panelist was asked to observe and make a judgement on the presented facial expression. Responses were recorded. In order to compare FEE responses with *Threshold* parameters $t_{low} = 0.5$ and $t_{high} = 0.5$, panelists were required to give a response even in the cases they were undecided. The average of the confusion matrices resulted by the panel test responses is reported in Table 4.6. The average recognition percentage was 86.8%.

It can be noticed that FEE and panelists results show similar average recognition performances, respectively 86.1% and 86.6%. Moreover, a similar trend is shown across the diagonals of the confusion matrices, i.e. for each single facial expression. In particular, the panel test responses generate 22.7% of misclassification for sadness judging it as *neutrality.* In this case, the misclassification percentage of the HNN with a *Threshold* of $t_{low} = 0.5$ and $t_{high} = 0.5$ was 0.4% and the misclassification percentage of the HNN with a *Threshold* of $t_{low} = 0.3$ and $t_{high} = 0.7$ was 0.2%. Such a behaviour can be found also for *fear.* In this case panelists misclassification percentage with *surprise* was 18.2%, while FEE misclassification percentages were 1.7% and 0.8%. It is interesting to notice how, changing the *Threshold* parameters, the FEE misclassification percentage of *sadness* as *anger* decreases from 14.5% to 2.7%, transferring the misclassification to the *don't know* column.

4.1.3.5 Conclusions

A hierarchical neural network approach to automatic facial expression recognition. A human face is acquired and four zones are automatically identified. A first pre-classification layer consisting of one Kohonen Self Organising Map for each zone extracts relevant features. Features are sent to a predictive layer consisting of a multilayer perceptron which performs the facial classification task.

The dataset was composed of 7700 patterns obtained from images of the six basic facial expressions. In order to select the network parameters, the architecture was trained in several configurations. In order to assess the goodness of the network results, twelve untrained panelists were asked to classify a representative subset of dataset. A confusion matrix was computed both for the hierarchical neural network and the panel tests, showing that there is no significant difference between the two judgements. The average recognition percentages were respectively 89.45% (86.1% if a threshold function is adopted) and 86.8%.

4.1.3.6 Real-time facial expression recognition implementation overview

The facial expression recognition system by means of an artificial hierarchical neural network approach was initially developed as an off-line tool. The data analysis software DataEngine was used to perform the acquisition, learning and test tasks. Such an architecture is unable to gain real-time execution, and the use of other software programming environment (e.g. MatLab) should prevent any connection to other processing modules (e.g. sensory and actuating driving systems) without loosing computational efficiency. The framework architecture here presented was born as F.A.C.E. test field with the aim to overcome these problems. As the framework core and the necessary tools were completed, an application has been developed to show the framework capabilities.

The application (*ARIApp_FEE*) uses the software libraries and it has been developed as a Win32 application. The application has been tested in a standard personal computer with the following properties:

- CPU: AMD Athlon XP 2000+ running at 1.67 GHz
- RAM: 512 MB DDR
- Video card: NVIDIA GeForce4 Ti 4200
- Digital camera: Firewire 1394 PGR DragonFly (640x480, 24 bpp)

The starting point is the declarations of the needed structures:

- four maps have been declared as *IFNeuronGroups* structure to provide cortex input data for each face zone: left eye (15x15 dots), right eye (15x15 units), nose (15x20 units) and mouth (30x10 units).
- four Kohonen maps have been declared as *IFKohonen* structure. These maps represents the first classification layer of the hierarchical neural network. For each *IFKohonen* map (10x10 neurons) a target (7 neurons) and output (7 neuron) map has been defined.

- three maps have been defined for the multilayer perceptron, using the *IFNeuronGroup* structure. The input layer is simply a grouping alias map of the *IFKohonen* output maps (28 neurons). The hidden layer (7 neurons) and the output layer (7 neurons) of the multilayer perceptron represents the remaining two maps. Moreover, a bias unit (1 neuron) has been defined as *IFNeuron* structure.
- an *IFMLP* structure has been defined to perform the multilayer perceptron process.
- four *FileSensorDriver* structures to obtain data input from training and test files.
- A *CameraSensorDriver* structure to obtain data input from a digital camera.

During the application initialisation all entities are positioned into the three-dimensional space and an appropriate text label is assigned to each structure. Connection are established between Kohonen maps and cortex maps with synaptic weights in the range [0.0 ; 0.1]. The learning rate, the learning radius and the decay values are initialised with the above discussed values. The MLP process is initialised in order to perform the layers and bias connectivity. Synaptic weights of the MLP structure are initialised in the range [0.0 ; 0.1]. The learning rate and the momentum values are initialised with the above discussed values. Four *FileSensorDriver* structures are then initialised to prepare the input data coming from the files containing the training set and the test set. Each file contains 11 examples for each of the seven facial expressions to be classified. Thus 77 examples are available for training and 77 more examples are available for test. Moreover a *CameraSensorDriver* is initialised to set up the digital camera input which will be used to perform real-time test with a human subject. The face-tracking process will automatically find the four regions of interest inside the acquired image (left eye, right eye, nose and mouth) and the driver will stretch the images to fit the cortex dimensions. This driver is initially switched off and

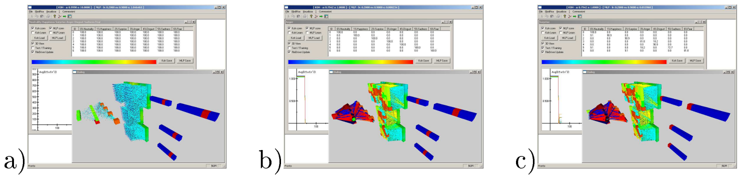


Figure 4.17: Hierarchical neural network rendering: a) initialisation; b) after training phase; c) during test phase

it will be activated when the *FileSensorDrivers* will be disconnected. Finally, all the declared structures are registered into the object manager (*3DWorld*) which will synchronize all the running processes. Extra user initialisation includes the setting up of the frame rate control and of the confusion matrix structure.

As the application starts, the structure can be visualised with the OpenGL support, as it is shown in Fig. 4.17.

During each update step a call to *SetInput* and to *Update* methods over the *3DWorld* object is sufficient to perform the processing of all the registered modules. The memory management monitor shows a stable memory usage of 34 MB of RAM, and a CPU usage equal to 99%.

During the training phase the application shows a frame rate equal to 23 fps (frame per second), where a frame represent the propagation of a single example. After less than 200 training epochs, the network gains the stability and the confusion matrix show a recognition percentage equal to 100% using a decision threshold equal to 0.5. The average squared error is equal to 2×10^{-4}. The error value for the *fear* facial expression is equal to 6.8×10^{-4}, while the error for the other facial expressions lies in the range $[1.4 \times 10^{-4}; 1.7 \times 10^{-4}]$.

Once the training is completed, the learning can be switched off. The test phase, running with a frame rate equal to 43 fps, shows a confusion matrix reported in Table. 4.7, with an average squared error equal to 5×10^{-2}.

The error graph over epochs is showed in Fig. 4.18. From epoch 0th to epoch 250th the learning process has been performed. After the epoch 250th,

	Neutr.	**Happ.**	**Surpr.**	**Ang.**	**Disg.**	**Sad.**	**Fear**
Neutrality	**100.0**	0.0	0.0	0.0	0.0	0.0	0.0
Happiness	9.1	**90.9**	0.0	0.0	0.0	0.0	0.0
Surprise	0.0	0.0	**90.9**	0.0	0.0	0.0	0.0
Anger	0.0	0.0	0.0	**81.8**	18.2	0.0	0.0
Disgust	0.0	0.0	0.0	0.0	**90.9**	0.0	0.0
Sadness	9.1	0.0	0.0	18.2	0.0	**72.7**	0.0
Fear	0.0	0.0	9.1	0.0	0.0	0.0	**81.8**

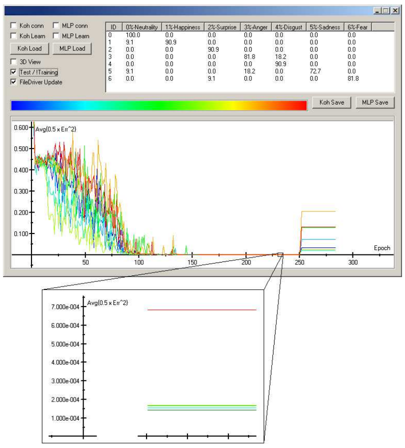
Table 4.7: Confusion matrix obtained in the test phase of *ARIApp_FEE* application

the graph is relative to the test phase.

Real-time facial expression recognition is performed switching off the file drivers and activating the *FEECameraSensorDriver*. Since face-tracking is performed, the human subject is able to move within the camera field view. The driver correctly detects the human face and the four zones of interest. Even the subject is moving and his distance from camera is changing, the system tries to operate a normalization both for image brightness, position and size. As a result, the human face appears to be always at the same size and position and with the same average brightness. Preliminary tests show real-time recognition percentage values very close to panelists responses, which are reported in Table 4.6. It is to be noticed that the hierarchical neural network was trained on a very small data set (77 examples, 11 for each facial expression) on a single subject. Beside of this, preliminary tests of the architecture shows the ability to correctly classify basic facial expressions (i.e. neutrality, happiness, surprise, disgust) even in other human subjects.

4.1.4 Case Study: Facial Automaton for Conveying Emotions as a Tool for autism (F.A.C.E.T.)

It is well documented that the processing of social and emotional information is impaired in people with autism. Recent studies have shown that individuals,



☐ Koh conn ☐ MLP conn
☐ Koh Learn ☐ MLP Learn
☑ Test / !Training
☑ FileDriver Update

ID	0%-Neutrality	1%-Happiness	2%-Surprise	3%-Anger	4%-Disgust	5%-Sadness	6%-Fear
0	100.0	0.0	0.0	0.0	0.0	0.0	0.0
1	9.1	90.9	0.0	0.0	0.0	0.0	0.0
2	0.0	0.0	90.9	0.0	0.0	0.0	0.0
3	0.0	0.0	0.0	81.8	18.2	0.0	0.0
4	0.0	0.0	0.0	0.0	90.9	0.0	0.0
5	9.1	0.0	0.0	18.2	0.0	72.7	0.0
6	0.0	0.0	9.1	0.0	0.0	0.0	81.8

Figure 4.18: Error graph over epochs. Learning error is showed from epoch 0th to epoch 250th. Test error is showed after epoch 250th

particularly those with high functioning autism, can learn to cope with common social situations if they are made to enact possible scenarios they may encounter in real life during therapy. The main aim of this work is to describe an interactive life-like facial display (FACE) and a supporting therapeutic protocol that will enable us to verify if the system can help children with autism to learn, identify, interpret and use emotional information and extend these skills in a socially appropriate, flexible and adaptive context. The therapeutic setup consists of a specially equipped room in which the subject, under the supervision of a therapist, can interact with FACE. The android display and associated control system has automatic facial tracking, expression recognition and eye tracking. The treatment scheme is based on a series of therapist-guided sessions in which a patient communicates with FACE through an interactive console. Preliminary data regarding the exposure to FACE of two children are reported.

4.1.4.1 Introduction

Autism is a developmental disorder characterized by qualitative impairments in social interaction and communication and a restricted range of activities and interests. It is in fact well documented that individuals with autism have impairments in processing social and emotional information. This is particularly evident in tasks assessing face and emotion recognition, imitation of body movements, interpretation and use of gestures and theory of mind [65, 66, 67, 68, 69].

Typically developing infants show preferential attention to social rather than inanimate stimuli; in contrast, individuals with autism seem to lack this early social predisposition [70, 71]. This hypothesis was recently substantiated in a neurofunctional study of facial perception in autism [72], in which adequate task performance was accompanied by abnormal ventral temporal cortical activities, which in turn suggested that participants had treated faces as objects. Klin et al [73] created an experimental paradigm to measure social functioning in natural situations, in which they used eye-tracking technology to measure visual fixations of cognitively able individuals with autism. When viewing naturalistic social situations, people with autism demonstrate abnormal patterns of visual pursuit, consistent with reduced salience of eyes and increased salience of mouth, bodies and objects.

In addition, individuals with autism use atypical strategies when performing such tasks, relying on individual pieces of the face rather than on the overall configuration [74]. Alongside these perceptual anomalies, individuals with autism have deficits in conceiving other peoples mental states.

According to the cognitive theory of mind-blindness [75], this impairment is related to the difficulty that autistic people have in conceiving of people as mental agents. Mind-blindness is thus the inability to perceive another persons mental state. Recent studies have shown that individuals, particularly those with high functioning autism, can learn to cope with common social situations if they are made to enact possible scenarios they may encounter. By recalling appropriate modes of behavior and expressions in specific situations, they are able to react appropriately. There are now a number of highly structured therapeutic approaches based on emotion recognition and social skill training using photographs, drawings, videos or DVD-ROMs (for example Mind Reading, produced by Human Emotions, UK). Their aim is to enable individuals with autism to interpret meanings and intentions of people and to anticipate their emotional reactions to typical situations they may encounter during the course of their daily lives. These methods show that basic emotion understanding can be taught; however since the learning process derived from these therapeutic approaches repeatedly uses a limited repertoire of predefined scenarios, it is biased toward the memorization of a scene, and its interpretation within a therapeutic setting, and so does not allow generalization or abstraction of the experience. In fact, the capacity of generalizing that learnt within a therapeutic setting is one of the principal problems of currently used treatments for autism. The literature regarding this aspect is increasingly focused on the new body of knowledge on the autistic disorder and on the recognition that precocious and individualized treatment can significantly improve the lives of patients [76, 77, 78, 79]. It remains unclear however, whether this type of treatment can influence core aspects of the disorder, such as social impairment. This implies that the effectiveness of a given treatment should be based on, amongst other criteria, a measure of social deficit. Unfortunately, several treatments described in the literature are beset by methodological problems which hinder the generalization of results for comparative purposes. For example, according to Kasari [79] treatment methodologies are characterized by elements such as a lack of coherent diagnostic procedures prior to the start of trails, inappropriate measurements to evaluate the results, evaluation of solely short term effects and most seriously, a lack of evaluation of the most clinically significant characteristics of the autistic disorder.

Currently, the only robotic systems for therapeutic purposes are the AURORA (autonomous robotic platform as a remedial tool for children with autism) and the Mobile Robotic Toys as Therapeutic Tools for Autism projects. In these projects mobile robots are used to encourage children with autism to take ini-

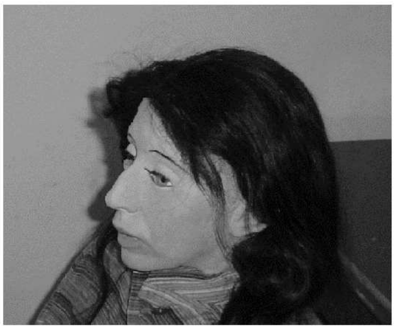


Figure 4.19: Facial Automaton for Conveying Emotions (FACE).

tiative and to become engaged with the robotic toys in a variety of different actions [80, 81, 82]. These robots are incapable of any biomimetic or emotional representation and do not include any three dimensional facial display.

We present here FACE (Facial Automaton for Conveying Emotions) [?] shown in figure 4.19, a facial automaton capable of expressing and recognizing basic emotions and describe an innovative robot-based treatment method which focuses on core aspects of the autistic disorder, namely social attention and the recognition of emotional expressions.

FACE acts as an interface between the patient and a trained therapist. A dedicated experimental set up enables both the creation of predefined social situations as well as the possibility of the therapist rapidly setting up individualized scenarios during a session. Moreover, the flexible and interactive modular architecture of the control system allows each session to be recorded, repeated or modified.

FACE could have greater visual impact for patients than other methods used for social training and could greatly reinforce them. It can also enable more complex and varied situations to be constructed during therapy. Moreover, as argued by Jacqueline Nadel, social imitation of a robotic experimenter can pave the way to the acceptance of social environment and human presence [83].

4.1.4.2 F.A.C.E.T.

Experimental Setup The android consists of a passive articulated body equipped with an anthropomorphic head. For the past 5 years the Interdepartmental Research Centre E. Piaggio of the University of Pisa has been involved in an ambitious project to develop FACE: a believable facial display system based on bio-mimetic engineering principles. The underlying philosophy and design approach of the display is founded on the simulation of biological behavior using materials, structures and control algorithms that can replicate some of the functions and responses of living systems. The architecture of the facial automaton consists of an anthropomorphic head and a facial tracking and expression recognition device. The head consists of an artificial skull covered by an artificial skin which is a thin silicone-based mask equipped with sensory and actuating system. It is fabricated by means of life-casting techniques and aesthetically represents a copy of the head of a subject, both in shape and texture. FACE is able to express and modulate the basic emotions in a repeatable and flexible way, to quantitatively analyze the emotional reactions of individuals through optical analysis of facial expression, to track a human face over time and to automatically store all data. FACE's control can be performed by an external supervisor or by an algorithm which implements a predefined design. The skeletal support structure is a resin based reconstruction of the head of a real adult subject, realized using CAD/CAM. Soft tissues of the head were fabricated from materials used for facial reconstruction in the world of animatronics and archeology [84].

The technical limitations of FACE derive from two main factors. Firstly, technical limitations arise from the materials design and engineering and secondly the control strategy adopted. As far as the former is concerned, FACE employs state of the art materials and is currently capable of actuating the 6 basic expressions (happiness, sadness, surprise, anger, disgust, fear) and modulating them in small steps. For the moment, the control pathway is fixed. As the clinical trails progress, information regarding the response of children

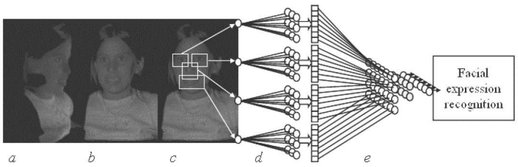


Figure 4.20: Facial tracking and expression recognition

with autism upon interaction with FACE can be used to refine and enhance the control algorithms. If necessary, control pathways which can be redefined by a therapist in real time can be implemented, so that during a session, FACE can assume an appropriate expression tuned to an individual subjects response. Using this method, it is envisaged that the patient could have the sensation of a verbal and visual dialogue with FACE without being aware of the therapists intervention.

4.1.4.3 The therapeutic protocol for autism

The experimental setup is illustrated in figure 4.21. It consists of a specially equipped room, provided with two remotely orientable video cameras, in which the child, under the supervision of a therapist, can interact with FACE through an interactive software by means of a liquid crystal screen and a keyboard or mouse (Interactive Module). Both FACE and the interactive module are connected to a computer (PC1). The child wears a system for recording physiological data (BioPac) which can be saved in a database (DB). The commercial Biopac® data acquisition system provides a versatile high performance, modular system that allows safe human measurement, such the electrocardiogram and skin temperature. The DB also contains data from the audio visual recording system present in the room and is connected to PC3. Other therapists or Hidden Observers can compile evaluation sheets during sessions, and the data scanned from these, can also be added to the DB and used for successive analysis. The

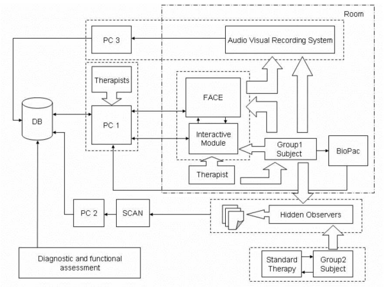


Figure 4.21: Experimental setup.

interactive software, which both therapist and child can access, implements the treatment scheme.

The Treatment Scheme Figure 4.22 illustrates the treatment scheme in the form of a flow diagram. Two distinct modalities are employed: the first is based on a repertoire of pre-selected social situations and the second allows the therapist to realize new situations as a consequence of the real time interaction between FACE and the child. The initial sessions are devoted to familiarization of the child with the robot, and will also serve to observe spontaneous reactions of the child when affronted with FACE. During the familiarization phase it is possible to identify verbal and non verbal expressions of the child which can be used to ascertain the degree of social attention toward the robot. This can be done by hidden raters through an original Grid for the Assessment of Social Attention derived from previous study on early autism at Stella Maris Institute [71, 95]. The grid is composed of a series of items (Looking at FACE; Looking at the therapist; Looking at objects; Orienting toward FACE; Orienting toward the therapist; Orienting toward objects; Smiling at FACE; Smiling at the therapist; Smiling at objects; Vocalizing to FACE; Vocalizing to the therapist; Vocalizing to objects) referring to behaviors representative of the child's social and nonsocial competencies. The raters compute the frequencies for each item, with the assistance of a specific coding software (Noldus® The Observer 5.0). Inter-rater reliability will be calculated using the Pearson coefficient. The first element to be evaluated during treatment is the capacity of the child to imitate the expressions of FACE. Factors such as spontaneous imitation, or imitation upon presses by the therapist can be considered, as well as the goodness of imitation. It is also be possible to increase the degree of a given emotion on FACE to induce or potentiate imitation if necessary.

When exposing the child to the collection of social situations a series of lessons are prepared on the basis of selected emotions to enable the child to develop abilities in two particular spheres:

- FACIAL EXPRESSION ASSOCIATION: this is the ability of the child to associate an emotion with that expressed by FACE. This is taught in two ways:

a) Facial Matching: by allowing the child to select, amongst several images, an image of a human face expressing the same emotion as FACE and

b) Emotion Labelling: through verbal labelling of an emotion expressed by FACE after being presented with several labels.

- EMOTION CONTEXTUALIZATION: This is the contextualisation of an emotion by presenting the child with different social situations and then asking him to select an appropriate response for FACE.

After the initial training phase, patients will be encouraged to learn through a trial and error approach, and therapists will assign a score based on the number of correct responses during the training sessions.

Evaluation of Treatment Effect To enable a quantitative evaluation of the treatment under progress, two checkpoints for patient evaluation are foreseen: the first at the start and the second at the end of the treatment sessions. The following evaluation protocols are administered at the checkpoints:

- A psychophysical test for evaluating face perception with emotive content. This test allows us to discriminate if facial expression processing is component-based or holistic and to verify its dependence on emotive content. Our aim is to compare the performance of autistic children, thought

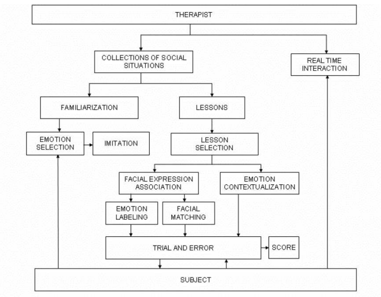


Figure 4.22: Treatment scheme.

to be generally impaired in these tasks, with that of typically developing children. We present facial stimuli on a computer monitor and all participants are tested individually. Stimuli can be composed of two equal or different flanked faces. Different faces are obtained through the Thatcherisation of only one feature of the face (i.e. eyes, nose or mouth) [96, 97]. Both faces in the stimulus can be either neutral or expressing emotions. The children are required to indicate whether faces are the same or different in two-alternative forced choice procedures. We present 90 stimuli to each subject.

- Tests to evaluate the ability to recognize emotions. The reading-the-mind-in-the-eyes test, revised version, is used for assessing this function [98].
- ADOS-G to monitor the level of disability in the two areas addressed in this project: social skills and communication.
- A grid to evaluate social attention as mentioned above.

The database contains a log of all the information gathered and is extremely useful for a quantitative and statistical analysis of child behavior and response during treatment. For example, it is possible to correlate patient behavior (video images) with a given expression of FACE at a particular instant, and also determine physiological correlates (BioPac), verbal stimulus (audio information) and focus of attention (eye tracking) at the same instant. These variables can be related to information collected on the grid by trained observers, and used to provide an objective evaluation of the child being observed. The data can be analyzed using statistical methods, so as to confer a figure of merit to this method. Moreover, we evaluate the evolution of impairment level using ADOS-G and the other three tests described before and after treatment.

4.1.4.4 Preliminary Results

In order to obtain a preliminary evaluation of the behavior of an autistic child when exposed to the robotic FACE, we set up a preliminary experiment in which the reactions of two children, one typically developmental child and the other with autism were monitored and compared.

The child with autism was 7 years and 8 months and had been previously diagnosed with high functioning autism using the DSM-IV, ADI-R, and ADOS-G criteria [99, 100, 101]. He had been subject to psychological, neuropsychological and linguistic tests by expert clinicians at the Stella Maris Institute (IRCCS) of the University of Pisa, Italy, where he has been under treatment since the age of 5. His treatment consists of twice weekly individual psycho-educational sessions and parallel counseling with his parents. The diagnostic ratings for ADOS-G module were: a) Communication: 6 (cut-off for autistic spectrum: 3; cut-off for autism: 5); b) Social interaction: 9 (cut-off for autistic spectrum: 4; cut-off for autism:6); c) Total score: 15 (cut-off for autistic spectrum: 8; cut-off for autism: 12). The ratings of algorithm, obtained by ADI-R were: a) Social interaction: 15 (cut-off: 10); b) Communication: 19 (cut-off: 8); c) Stereotyped behavior and restricted interests: 4 (cut-off: 3). An evaluation of cognitive abilities using the Wechsler scale (WISC-R) [102], resulted in a total IQ of 83, Verbal IQ and Performance IQ of 85.

The experimental set up consisted of FACE resting on a fully dressed passive wooden mannequin body, in a specially equipped testing room within the Stella Maris Institute (figure 4.23a). The room was equipped with the following:

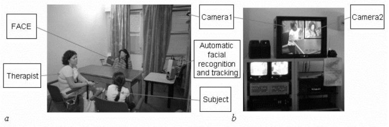


Figure 4.23: a) The experimental set up in the equipped room; b) The adjacent recording room, showing the monitors and recording system.

- Two video cameras in two corners of the room, one oriented toward FACE and one toward the subject.
- The volumetric acquisition system for face tracking and evaluation of the subjects expression.

The video cameras were operated by an engineer in an adjacent room in which the scenes recorded could be visualised on large screens (figure 4.23b). FACE was seated behind a large desk in a corner of the room such that it could not be easily touched by the subject, who sat in front of the desk beside a therapist. A heart rate monitor with a digital remote display was attached to the subjects chest to record any physiological correlation associated with the presence of the robot and its facial and head movements, as well as during the verbal interactions with the therapist. The therapists role was to probe the subjects emotional reaction to FACE and to mediate the interactions between subject and robot, and if necessary to circumvent any reactions of fear or aggression. Figure 4.23 illustrates the set during the session. The child in the photo is the typically developmental subject.

During the session with FACE, initially the child with autism did not show any interest in FACE. Successively, upon verbal suggestion by the therapist, he turned his eyes toward FACE and the therapist. When asked to express his opinion on FACE, he replied that it was a damsel, as though from a fairy tale, and that it was sad, thus attributing the robot with an emotion.

The control child was 8 years and 7 months. During the session, the child spontaneously observed FACE with great attention and expressed a positive reaction to it. When the robots facial movements were increased, the child became uncomfortable. The typically developmental child attributed the same emotion of sadness to FACE as did the autistic child. A time course of the physiological data correlated with the verbal suggestions of the therapist are plotted in figure 4.24. It shows the cardiac frequencies of the autistic (figure 4.24a) and typically developmental (figure 4.24b) subjects during the session with FACE. Verbal stimuli or the behavior of the therapist are indicated in the text boxes. The boxes indicate the initiation of a verbal stimulus. The autistic patients session was longer than that of the typically developmental child; this is due to the shorter attention periods of the autistic child. From the figures, it can be seen that after a request of focusing attention on FACE or to describe FACE's movements and appearance, the patient did not suffer a rapid increase or oscillation of cardiac frequency as the typically developmental subject did, in fact it seems that the patient is relaxing. In first approximation, the patient did not consider FACE and its movements threatening or surprising, implying that the robot fitted into his scheme of the environment. This initial result suggests that children with autism can be led to interact proactively with an android and that their predilection for interaction with non-human artifacts can be exploited in a positive manner through the use of FACE and the protocol here described.

4.1.4.5 Conclusion

During the next two years, a clinical study will be undertaken to determine the full potential of FACE and the protocol described in treating children with autism. It will involve 20 children diagnosed through ADI-R and ADOS-G as high functioning autistics; 10 children will be treated twice a week with this method over a period of 6 months and the others will undergo regular therapy. Halfway and at the end of this period, the patients will be evaluated using standard diagnostic and specific instruments for the evaluation of physiological and psychological data. Our hypothesis is that this method will diminish social impairment and increase expressiveness, facial mimicry, and shared attention, and thus it will lead to a better quality of life for children and adults affected

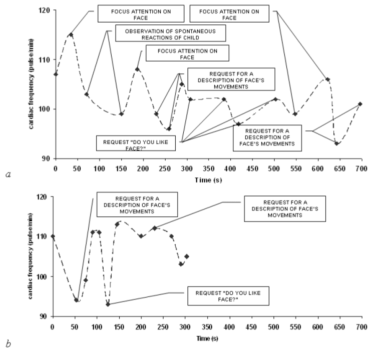


Figure 4.24: Heart rate during the preliminary evaluation. Labels show therapists verbal actions. a) autistic child; b) typical child.

by autism.

The concept of a believable humanoid display or even of its sub aspects has nevertheless far reaching implications and applications that could potentially span a wide variety of fields. These can include other possible medical applications such as pre-surgery study for facial reanimation transplant, speech therapy or a new communication way for deaf people. It can also pose the basis to introduce new channels of interactivity in other intelligent artificial systems, spreading out over philosophical and psychosocial fields, including the exploration of possible areas of inter-exchange with the neurosciences.

4.2 Leabra Neuron Model

The Leabra base model is a simplified version of the Hodgkin-Huxley model. Both models are shown in Table. 4.8.

Leabra uses a point neuron activation function that models the electrophysiological properties of real neurons, while simplifying their geometry to a single point. This function is nearly as simple computationally as the standard sig-

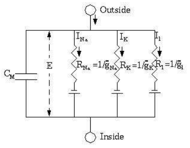
	Equations
Hodgkin-Huxley	$I = C_m \cdot \frac{\partial V}{\partial t} + \bar{g}_K \cdot n^4 \cdot (V - V_K) + \bar{g}_{Na} \cdot m^3 \cdot h \cdot (V - V_{Na}) + \bar{g}_l \cdot (V - V_l), \quad V(0) = V_0$ $\frac{dn}{dt} = \phi \cdot (\alpha_n \cdot (1-n) - \beta_n), n(0) = n_0$ $\frac{dm}{dt} = \phi \cdot (\alpha_m \cdot (1-m) - \beta_m), m(0) = m_0$ $\frac{dh}{dt} = \phi \cdot (\alpha_h \cdot (1-h) - \beta_h), h(0) = h_0$ $\phi = 3^{(T-6.3)/10}$  Outside I_{Na} I_K I_l C_M E $R_{Na} = 1/\bar{g}_{Na}$ $R_K = 1/\bar{g}_K$ $R_l = 1/\bar{g}_l$ Inside
Leabra Model	a) $g_e(t) = (1 - dt_{net}) g_e(t-1) + dt_{net} \left(\frac{1}{n_p} \sum_k s_k \frac{r_k}{\sum_p r_p} \frac{1}{\alpha_k} < x_i w_{ij} >_k + \frac{\beta}{N} \right)$ b) $g_i^{\Theta} = \frac{g_e^* g_e (E_e - \Theta) + g_l g_l (E_l - \Theta)}{\Theta - E_i}$ $g_i(t) = g_i^{\Theta}[k+1] + q(g_i^{\Theta}[k] - g_i^{\Theta}[k+1])$ c) $V_m(t+1) = V_m(t) + dt_{vm}[g_e(t)\bar{g}_e(E_e - V_m(t)) + g_i(t)\bar{g}_i(E_i - V_m(t)) + g_l(t)\bar{g}_l(E_l - V_m(t))]$ $y_j = \frac{\gamma[V_m - \Theta]_+}{\gamma[V_m - \Theta]_+ + 1}$ $y_j^*(x) = \int_{-\infty}^{+\infty} \frac{1}{\sqrt{2\pi\sigma}} e^{-\frac{z^2}{2\sigma}} y_j(z-x) dz$

Table 4.8: Top: Hodgkin and Huxley neuron model, based on chemical species. Bottom: Leabra model; a) Excitatory conductance; b) Inhibitory k-WTA function; c) Membrane potential; d) Activation function

moid activation function, but the more biologically-based implementation makes it considerably easier to model inhibitory competition, as described below. Further, using this function enables cognitive models to be more easily related to more physiologically detailed simulations, thereby facilitating bridge-building between biology and cognition.

4.2.1 Leabra Neuron Model Implementation

Leabra uses a kWTA (k-Winners-Take-All) function to achieve inhibitory competition among units within a layer (area). The kWTA function computes a uniform level of inhibitory current for all units in the layer, such that the k+1th most excited unit within a layer is generally below its firing threshold, while the k-th is typically above threshold. Activation dynamics similar to those produced by the kWTA function have been shown to result from simulated inhibitory interneurons that project both feedforward and feedback inhibition [105]. Thus, although the kWTA function is somewhat biologically implausible in its implementation (e.g., requiring global information about activation states and using sorting mechanisms), it provides a computationally effective approximation to biologically plausible inhibitory dynamics. For learning, Leabra uses a combination of error-driven and Hebbian learning. Implementation diagrams are shown in Table 4.9.

4.2.2 Associative Memory: Hippocampus Simulation

In literature, the working memory models inspired to biology are based on neural networks. Most of these architectures are not able to proceed in new learning processes without loosing memory of the past learning processes (catastrophic interference) [104]. In order to overcome these issues models able to gain short-term priming in co-operation with other modules have been developed [105]. In particular, hippocampus-based models operate a pattern separation avoiding the catastrophic interference. Input patterns are spread among different interconnected modules following the McCloskey and Cohen model [106].

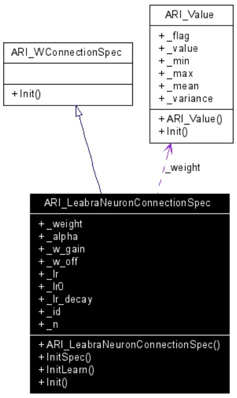
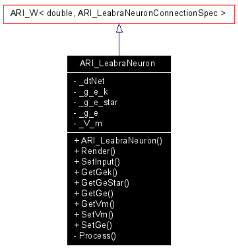
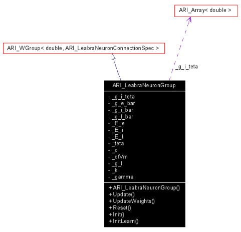
Connection	Neuron	Neuron Group
		

Table 4.9: Leabra model: inheritance and collaboration diagrams of structures for leabra model implementation

The Input Entorhinal Cortex and the Output Entorhinal Cortex maps represent respectively the input and the output of the net. Input and output maps have the same dimension in order to evaluate the activation and deactivation error by a one-to-one comparison of neuronal activity. Activation error represents the percentage of neurons that are firing in the Output Entorhinal Cortex and that are under threshold in the Input Entorhinal Cortex. Deactivation error represents the percentage of neurons that are under threshold in the Output Entorhinal Cortex and that are firing in the Input Entorhinal Cortex. In our application artificial neurons and learning algorithm were implemented following the model proposed by O'Really and Munakata [106, 107].

4.2.2.1 Pattern Separation and Completion

The McCloskey and Cohen architecture consists of several interconnected two-dimensional self-organising maps of artificial neurons. The Input Entorhinal Cortex and the Output Entorhinal Cortex maps represent respectively the in-

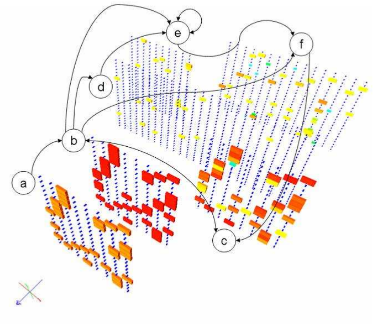


Figure 4.25: Hippocampus model proposed by McCloskey and Cohen; a) Input pattern; b) Input Entorhinal Cortex; c) Output Entorhinal Cortex; d)Dentate Gyrus; e) CA3; f) CA1

put and the output of the net. Starting from the base models available in the framework core dedicated models for neurons, maps and connections have been derived. K-WTA algorithm has been implemented inside the process groups specification. Projection types were specified in order to guarantee pattern separation and completion, i.e. polar connection from input entorhinal cortex to dentate gyrus. Training and test procedures have been specified at the application-level, making it possible to connect input entorhinal cortex to CA1 during the training steps, and disconnecting it during the test steps. As above mentioned in our framework each entity have been represented in a running process. In the framework I/O interface, in order to dispatch data from olfactory system to the core processes, dedicated modules were also derived. Synchronization is managed by a main process with the support of the framework functionalities.

An hippocampus-based working memory processing module and its implementation in the framework are showed in Fig. 4.25. The system will be used in order to analyse multi-sensor data signals transmitted by an artificial olfactory system (e-nose) [108].

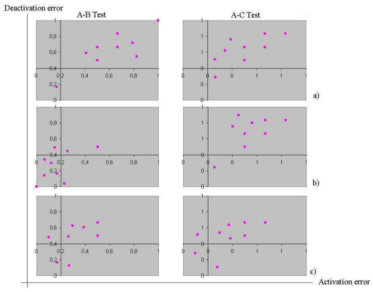


Figure 4.26: Error mapping during AB and AC tests over random data: a) before training; b) after 3-epochs AB training; c) after 5-epochs AC training

4.2.2.2 Paired-Associate Learning (AB-AC Test)

A training on the associations of the A-B elements was performed and followed by a test to assess the accuracy rate of responses. High accuracy rate was obtained. After, a training on the associations of the A-C list was performed and followed by a test for both A-B and A-C lists. Right associations can be observed for the A-C list, while memory for A-B list is still present with minor misclassification showing the avoidance of catastrophic interference. Preliminary results obtained using random input patterns are showed in Fig. 4.26. Activation and deactivation error is reported in a two-dimensional plane. During the test step, when the Input Entorhinal Cortex does not include the B or C pattern, the error is obtained comparing the Output Entorhinal Cortex to the complete original pattern. Tests show an association of both A-B and A-C lists with a recognition percentage comparable with literature results [107].

4.2.3 Case Study: An Electronic Nose for Olive Oil Classification

Starting from the base models available in the framework core dedicated models for neurons, maps and connections have been derived to process signals coming

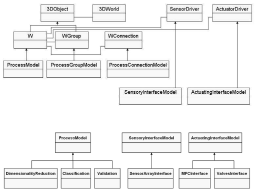


Figure 4.27: Hierarchical and collaboration chart of the framework structures for electronic nose simulation

from an electronic nose using conductive polymer sensors. As above mentioned in our framework each entity have been represented in a running process. In the framework I/O interface, in order to dispatch data from olfactory system to the core processes, dedicated modules were also derived. Synchronization is managed by a main process with the support of the framework functionalities. The use of specialized modules architecture derived from the frameworks base structures permits the realization of an independent application able to drive e-nose sensor array, to perform data analysis tasks and to control mass flow controllers (MFCs) and valves actuators. Fig. 4.28 shows processing and interface modules developed for e-nose control system.

The electronic nose consists of an array of eight chemoresistive conductive polymer sensors [109] interfaced to an electronic board. The sensor array is lodged in an exposure chamber where a gas carries the volatile samples. A chemico-physical interaction occurs between the input volatiles and sensing layers resulting in an electric resistance variation for each sensor. Response of each sensor is thus an analog signal versus time. Signals are acquired and digitalized by the electronic board and they are managed by the framework drivers.

The e-nose was used to assess the headspace of 15 Italian olive oil samples of different regions (Tuscany, Apulia and Sicily) classified by an official panel test as extra virgin, virgin and defected. A suitable number of vials (volume 125ml) was prepared by pouring 10 ml olive oil, sealing and waiting a few hours for equilibration. A block scheme of the experimental setup is shown in Fig. 4.28.

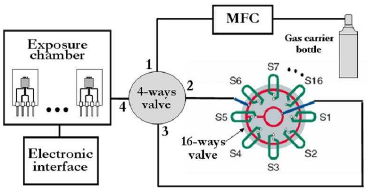


Figure 4.28: Block scheme of the e-nose experimental setup

The experimental protocol consisted of: 32 samplings of baseline acquisition; 5 samplings of exposure; 32 samplings of desorption. A flow rate of 200ml/min and a scan rate of 4 Hz were applied.

The goal was the association of an olive oil both to its geographic origin and to its quality. The learning of new associations must not interfere with the memory of the previously learned associations. In order to verify the ability of the processing architecture in associative and short-term memory, a paired-associate learning (AB-AC test) was applied. Elements A, representing the output signals of the e-nose for each olive oil, are associated to elements B and C, representing the geographic origin and the quality class respectively.

The experiment execution and data processing are controlled at run-time by the main process. The main process continuously builds the dataset. A labeling process runs in order to build the A-B list. A training process runs to learn the associations of the A-B elements. A test process assesses the accuracy rate of each response. After, a labeling process runs in order to build the A-C list. A training process runs to learn the associations of the A-C list. A test process assesses the accbase uracy rate of each response for both A-B and A-C lists. During the execution of the training processes each paired input pattern is mapped on the Input Entorhinal Cortex while the association response results into the Output Entorhinal Cortex. During the execution of the test processes, when the Input Entorhinal Cortex does not include the B or C pattern, the error is obtained comparing the Output Entorhinal Cortex to the complete original

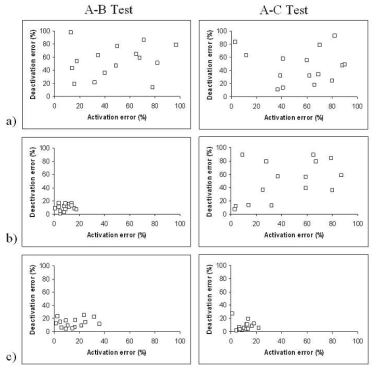


Figure 4.29: Error mapping during AB and AC tests over olive oil data: a) before training; b) after 3-epochs AB training; c) after 5-epochs AC training

pattern. Results of test processes are shown in Fig. 4.29.

For each sample activation and deactivation percentage error is reported in a two-dimensional plane. A first test on the untrained architecture was performed; as expected high classification error was obtained (Fig. 4.29a). Test results after the execution of the A-B training process are shown in Fig. 4.29b. As it can be noticed a minor misclassification is obtained for the A-B list, while A-C elements are still not recognized. Test results after the execution of the A-C training process are shown in Fig. 4.29c. Right associations can be observed for the A-C list, while memory for A-B list is still present with minor misclassification showing the avoidance of catastrophic interference and recognition percentages comparable with literature results [37]. Table 4.10 shows the mean activation and deactivation percentage errors.

	A-B Test Error				A-C Test Error			
	Mean		Standard Deviation		Mean		Standard Deviation	
	Act.	Deact.	Act.	Deact.	Act.	Deact.	Act.	Deact.
a)	48.5	54.1	27.3	25	56	46.3	26.4	25.7
b)	9.3	10.1	5	5	43.2	49.9	29.3	29.8
c)	15.4	13.2	10.5	6.8	10.9	8.3	5.7	7.2

Table 4.10: Standard deviations and mean values of activation and deactivation error in A-B and A-C tests. a) untrained architecture; b) after A-B training; c) after A-C training

4.3 An Artificial Neuron Model for Real-Time Data Processing

The complexity of a biological neuron may be reduced by using several mathematical models. Each of these reproduce some of the functionalities of real neurons, such as the excitability in response to a specific input signal. The most accurate model for a biological neuron has been developed by Hodgkin and Huxley [110] and it is able to exactly reproduce the shape of the action potential of a neuron by taking into account the ionic currents. Beside of this, the model is computationally expensive and it takes about 1200 FLOPs (FLoating Point Operations) to simulate one millisecond of a single neuron activity. Several attempts have been made in order to reduce the mathematical complexity of this model. The most effective result has been obtained by the Morris-Lecar model [118], which is able to describe the oscillations of the muscular fibers of the giant squid and it is still close to the Hodgkin-Huxley model accuracy. Unfortunately the computational complexity is still high, since it takes about 600 FLOPs for one millisecond of neuron activity. Since these bottom-up approaches are focused on the characterization of the biophysic properties of the cell membrane, a different approach has been adopted by Fitzhugh and Nagumo [119], taking into account the information of the nervous signal as a temporal distribution rather than an action potential shape. This top-down approach leads to the

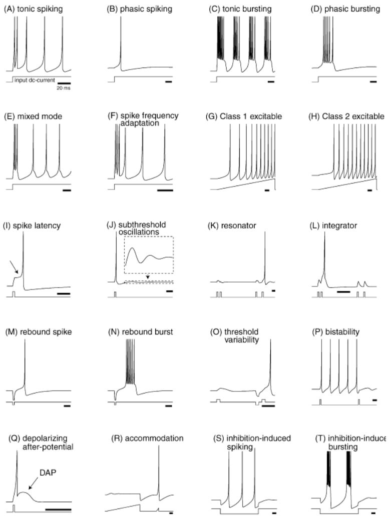


Figure 4.30: Some of the biological neuron behaviours.

development of parametric differential equations with the aim to match them with experimental results. The Fitzhugh-Nagumo model, wich takes about 72 FLOPS for one millisecond of neuron activity, is based on a variant of the Van Der Pol oscillator. Studies on the dynamics of non-linear systems swoed a large variety of behaviours. Actually, the use of mathematical analogies seems to be the only way to simulate a large number of interconnected artificial neurons. For this reason the integrate-and-fire model (and its variant models) is the simpler and most used model for classification and prediction tasks in pratical scenarios.

However, this reduction process leads to efficient artificial neuron models which are not able to reproduce most of the biological neuron behaviours. Some of such behaviours are showed in Fig. 4.30.

Izhikevich [120, 121, 113] recently developed a simple model for an artificial neuron wich is able to reproduce all the behaviours showed above. The model takes 13 FLOPs for simulate one millisecond of neuron activity and it is based on a top-down approach, using two differential equation with four parameters. The introduction of axonal delays shows the possibility to create a neural network able to perform classification and prediction tasks [114]. The connection of several maps follows and the Spike-Timing-Dependant Plasticity (STDP) rule [122], which permits the implementation of a real time learning rule based on signals which continuously flow from input systems. This architecture follows the theories of Edelman about the selection as the basis for the learning process [112, 115, 116, 117].

4.3.1 The Artificial Neuron Model Proposed by Izhikevich

The model proposed by Izhikevich for the artificial neuron simulation shows tha ability to reproduce the same accuracy of the Hodgkin and Huxley model. It can be resumed in the following relations:

$$\begin{cases} v' = 0.04v^2 + 5v + 140 - u + I \\ u' = a(bv - u) \end{cases}$$

A reset condition is needed, as it will be discussed later:

$$\text{if } v \geq +30mV, \text{ then } \begin{cases} v \leftarrow c \\ u \leftarrow u + d \end{cases}$$

The four parameters (a, b, c and d) are dimensionless values. The v variable represents the membrane potential of the neuron, while u keeps into account the activation of K^+ ionic currents and the deactivation of the Na^+ ionic currents. The I variable takes into account the synaptic currents and the bias currents as the input signal of the neuron. In Fig. 4.31 the velocity field of the dynamic system is showed. Depending on the values of the four parameters, the system may have a steady-state (which corresponds to a lack of activity in the neuron) and an unsteady-state (which corresponds to the presence of activity in the neuron). The reset condition is needed to perform the return of the system into the steady state after the neuron has fired.

Table 4.11 shows the values of the four parameters in order to obtain the neuron behaviours showed in Fig. 4.30.

	a	**b**	**c**	**d**
A) Tonic Spiking	0.02	0.2	-65	6
B) Phasic Spiking	0.02	0.25	-65	6
C) Tonic Bursting	0.02	0.2	-50	2
D) Phasic Bursting	0.02	0.25	-55	0.05
E) Mixed Mode	0.02	0.2	-55	4
F) Spike frequency adaption	0.01	0.2	-65	8
G) Class 1 excitable	0.02	-0.1	-55	6
H) Class 2 excitable	0.02	0.26	-65	0
I) Spike latency	0.02	0.2	-65	6
J) Subthreshold oscillation	0.05	0.26	-60	0
K) Resonator	0.1	0.26	-60	-1
L) Integrator	0.02	-0.1	-55	6
M) Rebound Spike	0.03	0.25	-60	4
N) Rebound burst	0.03	0.25	-52	0
O) Threshold variability	0.03	0.25	-60	4
P) Bistability	0.1	0.26	-60	0
Q) Depolarizing after-potential	1	0.2	-60	-21
R) Accomodation	0.02	1	-55	4
S) Inhibition-induced spiking	-0.02	-1	-60	8
T) Inhibition-induced bursting	-0.026	-1	-45	-2

Table 4.11: Values of the four dimensionless parameters used to obtain the corresponding neuron behaviour.

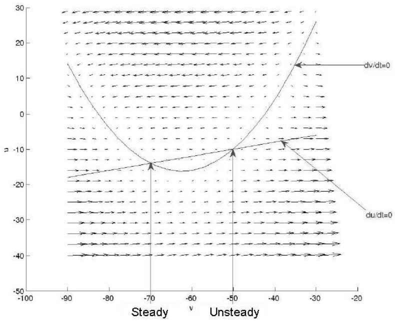


Figure 4.31: Velocity field of the system proposed by Izhikevich for the artificial neuron model. The presence of a steady-state and of an unsteady-state depends on the values of the a, b, c and d dimensionless parameters.

4.3.1.1 Selection of Neuronal Groups

The Theory of Neuronal Group Selection (TNGS) proposed by Edelman [112], suggests a novel way for understanding and simulating neural networks. To take into account this theory we have to use the *time* variable in the learning task, so that neural groups may raise from a selection process. This strategy has been adopted by Izhikevich, who simulated a minimal neural network which is able to show the property of *polychronization* [114]. In such a network a correspondence between synaptic weights and axonal delays exists as a result of the neuron bahaviour. One neuron can belong to many groups, which count is usually higher than the count of the neurons theirself. This guarantees a memory capability which is higher than the capability reached by the classical neuronal network.

Such an architecture has been implemented into the framework here presented, giving the possibility to connect the neuronal groups to sensory and actuating systems. The advantage of the use of the Izhikevich model makes it possible to gain time-space correlation on input signals.

4.3.1.2 Polychronization

The classical approach in artificial neural networks simulation takes into account the modulation of the action potential rithm as the only parameter for the information flowing to and from each neuron. Such a strategy seems to be in contrast with novel experimental results, since neurons are able to generate action potential which are besed on the input spike timings, with a precision till to one millisecond. The spike-timing synchrony is a natural effect that permits a neuron to be activated in correspondence of synchrounous input spikes, while the neuronal activation of the post-synaptic neuron is negligible if pre-synaptic spikes arrives asynchronously to the target neuron. Axonal delays usually lie in the range [0.1 , 44] milliseconds, depending on the type and location of the neuron inside the network. Such a property becomes an important feature for the selection of the neural groups as it is exposed by Edelman. In Fig 4.32 a polychronization example is showed. If pre-synaptic neurons fire at the same time (Fig. 4.32B) no relevant effect will be noticed in post-synamptic neurons. The neurons reaches the maximal activity if the pre-synaptic neurons fire with a specific timing, depending on the axonal delays of the synaptic connections, showing the capability to gain a time-space correlation on the input pattern (Fig. 4.32C-D).

4.3.1.3 Spike-Timing-Dependent Plasticity (STDP)

In the artificial neural network model, the synaptic connection are modified according to the STDP rule. The implementation of such strategy is computationally efficient and it is showed in Fig. 4.33

If a spike coming from an excitatory pre-synaptic neuron causes the fire of the post-synaptic neuron, the synaptic connection if reinforced since it given the possibility to generate an other spike in order to propagate the signal. Otherwise the synaptic connection is weakened. The values of the STDP parameters are choosen in order to permit a weakening that is grater than the reinforcement. Such a strategy permit the progressive removal of the unnecessary connections

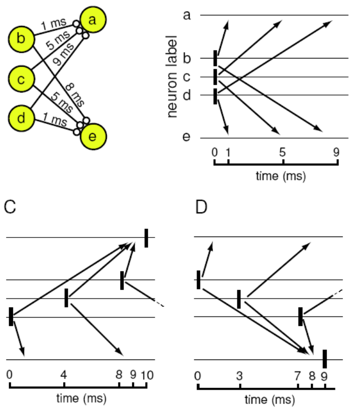


Figure 4.32: Polychronization example: A) synaptic connections coming from neurons b,c and d to neurons a and e have different axonal delays. B) C) D) The activity of the post-synaptic neurons is marked with a vertical dash. The arrows indicate the spike timings to the post-synaptic neuron

and the persistance of the connections between correlated neurons.

4.3.2 Implementation of the Izhikevich artificial neuron model in the framework

In order to implement a network able to use the polychronization feature as it is described above, a software module has been realised according to the ARI framework architecture. An *IzhikevichNeuron* structure (see Fig. 4.34) has been implement in the framework as a running process directly deriving from the *W* structure.

Template arguments have been specialised to obtain an *OUTDATA* as a real number (double precision floating point value) which represents the membrane potential of the neuron, and a *CONNSPEC* as a *IzhikevichNeuronConnectionSpec* structure, which is shown in Fig. 4.35.

The connection structure for such a process uses real numbers to manage the

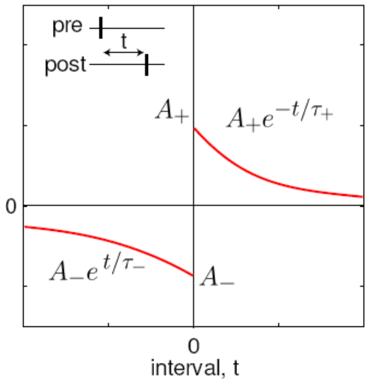


Figure 4.33: Spike-Timing-Dependent Plasticity (STDP). The sinaptic weight of the connection is incremented if the post-synaptic neuron fires after a time t from the fire of the pre-synaptic neuron. If t is negative, then the synaptic weight is decremented. The values of the parameters are: $\tau_+ = \tau_- = \mathit{20ms}$, $A_+ = 0.1$, and $A_- = \mathit{0.12}$.

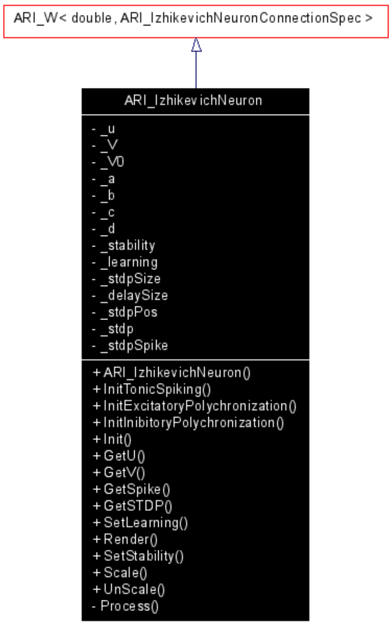


Figure 4.34: The IzhikevichNeuron structure

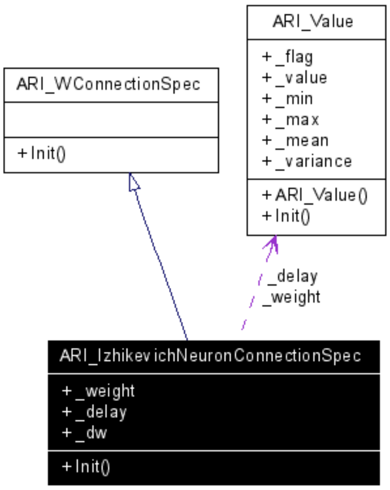


Figure 4.35: The IzhikevichNeuronConnectionSpec structure

synaptic weight and the synaptic channel delay. The values may be initialised by the user or randomly chosen by the framework according to the value initialisation parameters. A delta-weight value is needed for internal operations during the learning process, as it will be discussed later.

The *IzhikevichNeuron* structure is initialised using the *Init* method in order to setup the internal parameters (a, b, c, d) which specify the behaviour of the artificial neuron. Several initialisation wrapper methods are provided to use predefined behaviours as they are showed in Table 4.11. The STDP algorithm (Fig. 4.33) is implemented with a time-window of size equal to 1000 milliseconds. During this period the delta-weight values are updated according to the STDP rule, while weights are updated at the end of each period. During each period the structure traces the firing activity and the STDP status of the neuron, storing the information in two static arrays. The structure defines the private virtual method *Process* in order to perform the learning task. If the neuron is firning, the Process method reset the internal status (u, v) and the

STDP value is reported to a value equal to 0.1. Otherwise the STDP value is decreased with a time-constant equal to 20 milliseconds. Subsequently the input connections are browsed to update input current, whose contribute depends on the thalamic input neurons and on those neurons who fired with a timing equal to the connection delay. According to the STDP rule, the STDP value of the post-synaptic neuron is increased if it fired synchronously with the the pre-synaptic neuorn, and it is decreased if the pre-synaptic fire caused no firing in the post-synaptic neuron. Finally the status is updated following the Izhikevich model, and, if 1000 milliseconds are enlapsed, the synaptic weights of the connections coming from the excitatory neurons are updated with the actual delta-weigth values. During this step the weights are clamped within a convenient range and the delta-weight values are decreased with using a decay coefficient equal to 0.9.

The I/O buffering operations simply manage internal members and recall the base class methods. The rendering function provides the graphic visualisation of the soma and of the input connections.

The *IzhikevichNeuronGroup* structure (Fig. 4.36a), which represents a group of *IzhikevichNeurons*, has been derived from the *WGroup* base structure. The *IFNeuronGroup* structure will be used by high-level processes in order to perform the monitoring of the activity of the neurons during the learning and test tasks. Methods are provided to obtain the activation percentage (*GetActPerc*) and to retrieve the sub-group identification relating to a specified input pattern. A specific structure (*ARI_ING_Record*) has been realised to store the neuron reference and the activation time for each neuron belonging to the sub-group. Such records can be enumerated using the the iterator methods (*First*, *End*, *Next*, *Get*).

Moreover, an *IzhikevichMap* structure (Fig. 4.36b) has been derived from *W* base structure in order to speed-up the artificial neural group processing. This structure includes all the previous described structures, optimising the memory usage and computational efficiency.

A *ThalamicRandomSensorDriver* structure has been developed to train the

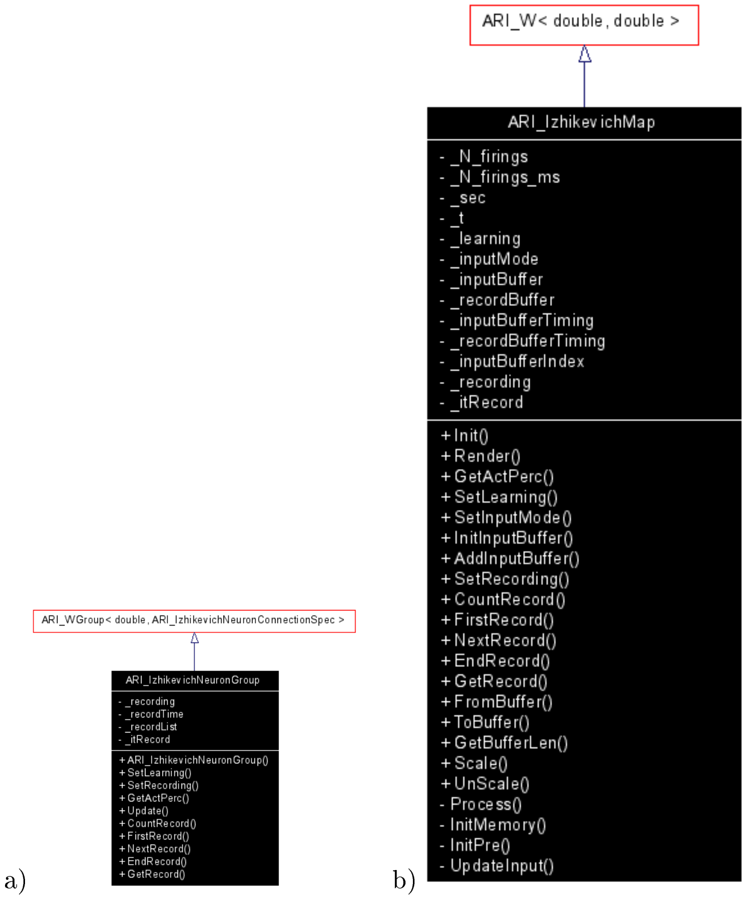


Figure 4.36: a) The IzhikevichNeuronGroup structure. b) The IzhikevichMap structure

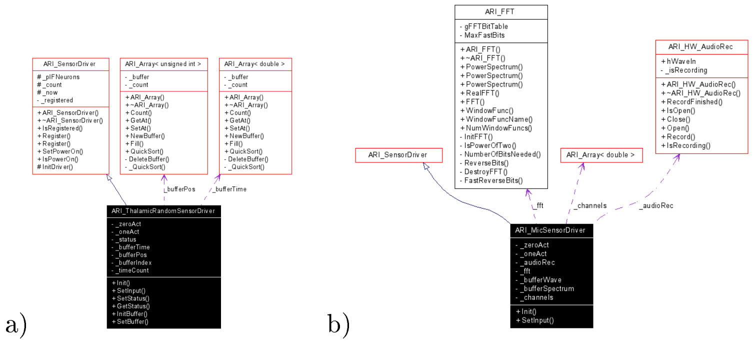


Figure 4.37: a) The ThalamicRandomSensorDriver structure. b) The MicSensorDriver structure

architecture with random signals. As it is described by Miller [125], such signals are the basis of the cortico-thalamic interplay of neural assemblies and temporal chains in the cerebral cortex. A *MicSensorDriverStructure* has been used to test the architecture with audio signals. For such signals the power spectrum has been obtained using the *ARI_FFT* structure. Both structures are showed in Fig. 4.37.

4.3.3 Simulation of a Minimal Group

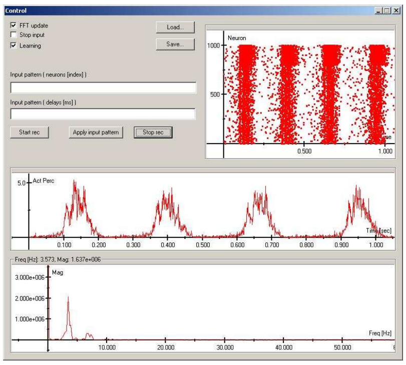
A minimal neural network of 1000 artificial neurons consisting in 200 inhibitory neurons and 800 excitatory neurons has been realised as a demonstration application to test each structure implmented in the framework. The *ThalamicRandomSensorDriver* was used to generate random input signals to the network during the learning process. During the test phase the *MicSensorDriver* was used and 32 components in the frequency range [500Hz ; 1000Hz] of the power spectrum of a real-time audio signal have been sent to a subset of the excitatory neurons.

The network design has been inspired to the anatomical structure found in the mammalian cortex. In respect to the total number (N) of neurons, a percentage equal to 80% consists of excitatory neurons, while the remaining 20% are inhibitory neurons. Cortical pyramidal neurons showing a Regular Spiking (RS) behaviour have been adopted fot the excitatory subsection, which correspond to appropriate values for the Izhikevich neuron model (a=0.02, b=0.2, c=-65, d=8). Inhibitory neurons have been simulated adopting the model of the cortical interneurons which exibits Fast Spiking (FS) property (a=0.1, b=0.2, c=-65, d=2). Each neuron is connected to M different neurons in order to obtain a connection probability (M/N) equal to 0.1, but inhibitory neurons are connected only to excitatory neurons. Moreover, the synaptic weights of the connections arising from the inhibitory neurons remain unchanged during the learning process, while those regarding the connections from the excitatory neurons change according to the STDP rule. Axonal delays are fixed in the range between 1 millisecond and 20 milliseconds. The time resolution has been set to one millisecond. The network activity has been monitored for more than 24 hours of simulation time while the application was running six times faster than real time. During the test phase with audio signals the network was slowed down in order to gain the correct input timing.

As the application starts, all the connections have the same synaptic weight. The network need many seconds to get stabilised through depression and strengthening of the synaptic weights. During this first phase, the network shows the presence of an high amplitude rithm, with frequency in the range between 2 Hz and 4 Hz (delta waves), as it is shown in Fig. 4.38.

Such rithms are very close to one of the four fundamental kind of waves that can be noticed in the central nervous system. Such waves are known as deep-sleep waves because they appear during the sleeping phase without dreams, in infants and in subject with special mental illness. Such waves arise from the way the thalamus operate, that is simulated in the model just by a regular random input with a frequency of one millisecond.

After 3600 seconds of network activity the spiking rithm becomes uncorrelated and frequency in the range between 30 Hz and 70 Hz appear (gamma waves) as it is shown in Fig. 4.39. The arise of such rithms is called PING



☑ FFT update
☐ Stop input
☑ Learning

Freq [Hz]: 3.573, Mag: 1.637e+006

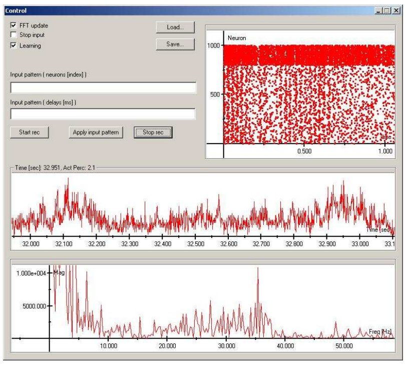
Figure 4.38: Simulation of the first 1000 milliseconds of the polychronizating network. Delta waves in the frequency range [2 ; 4] Hz are visible. top-right graph) Activity of the 1000 neurons over time. middle graph) Activation percentage over time. bottom graph) Amplitude of the activation percentage power spectrum over frequency.

(Pyramidal-Interneuron Network Gamma) and it seems to be related to the spikes of the pyramidal cells which excite the inhibitory interneurons. Such interaction allows a mutual inhibition which temporarily switch-off the network activity [124].

As 24 hours of time simulation are enlapsed, the network becomes stable and the oscillation rithm is assessed in the frequency range between 2 Hz and 7 Hz. Once the thalamic input is discarded, an appropriate input pattern may be used to test the neuronal group selection. An external stimulus will produce the activation of a corresponding neuronal group, depending on the timings of the input pattern. Since each neuron may belong to many groups, the number of the existing group is far grater than then the number of the neurons in the network.

According to the results obtained by Izhikevich, we will notice two kind of selection:

- neural selection: the STDP rule selects the conduction delays of the ini-



☑ FFT update
□ Stop input
☑ Learning

Input pattern (neurons [index])

Input pattern (delays [ms])

Time [sec]: 32.951, Act Perc: 2.1

Figure 4.39: Simulation of 1000 milliseconds of the polychronizating network after 3600 seconds of network activity. Gamma waves (30 Hz) are visible.

tially unstructured network. As a result a large number of groups appear, and each of these is able to perform a reproducible spike sequency with a precision of one millisecond.

- group selection: each stimulus that is used as input pattern is able to select one group inside the network, showing that the network is able to perform classification tasks. Such classification is realised by a memory capability which is far greater than the number of entities involved into the network. Such a structure is able to perform an its own representation of a specified context.

Chapter 5

Conclusions

In this work the author describes a high-efficiency architecture for parallel and real-time management of heterogeneous multi-transducers data processing. The interfaces with the external sensors and actuators, the specific control and processing methods and the data flowing through inner communication channels can be defined. For such entities the framework offers extendable structures, whose base implementation allows the realisation of high-efficiency data processing. The framework base architecture has been described. Systems equipped with multiple transducers, task running cooperative processes, off-line and real-time data aquisition and analysis tools, general stand alone applications represent some of the potential application areas.

Limitations of existing acquisition, analysis and control tools have been overcame. A library-oriented interface has been preferred to a user-oriented interface. Real-time analysis and actuation is gained for all the transducers and for all the running processes. Multi-process cooperation is possible thanks to a homogeneous communication language. The user can create extensions of new models of entities and processes. The data acquisition from sensor devices is granted by a protocol interface that is able to dispatch data coming from input systems. The data processing may be specified by the user inside the framework core. The actuator driving is granted by a protocol interface that is able to dispatch data from the framework core. Filters for sensory and actuating systems can be redefined according to the particular device technology; the efficiency of the filtering and buffering processes over the data coming from sensors and over the data directed to actuating devices is delegated to appropriate interfaces. The portability is allowed by a layered structure, an OS abstraction, and by the specification of the I/O drivers. A modular, reusable and object-oriented architecture grants a parallel distributed processing, making the framework base architecture available to the researcher as a structured programming environment. Such features make the framework a solution for high-complex simulation tasks, representing a powerful instrument for the development of complex simulation tools operating as off-line and real-time applications.

The framework architecture has been designed to take into account the problems related to the development and the simulation of artificial neural networks. Many neuronal models and control strategies have been investigated and developed inside the framework. The integrate-and-fire neuronal model with the support of the base network structures (i.e. self organizing maps, multi layer perceptrons) has been developed to realize applications for classification and prediction purpouses (i.e. real-time facial expression recognition). Thanks to the leabra neuronal model an architecture for the simulation of a short-term associative memory have been adopted to perform olive oil classification by means of an electronic nose. Finally, the Izhykevich neuronal model have been investigated and a minimal neuronal group running a spike-timing dependant plasticity learning strategy showed very interesting classification properties. Currently this structure is under study and it will be placed in a more general learning context.

FACE (Facial Automaton for Conveying Emotions), an ongoing project of the research group at the Interdepartmental Research Centre “E. Piaggio” of the University of Pisa, has been used as the test field for the control architecture here proposed. The goal of the project is to develop a believable android both in appearance and behaviour. The aim of the android is to act as a human-machine interface for non verbal communication within a simplified environment. The biomimetic principles followed in designing the materials and the control system of FACE were described. In designing the physical features of an android, the environment it is placed in and its function and objective are of prime importance. FACE is placed in an environment in which the only stimuli come from the non-verbal communication of a human placed in front of it. Its aim is to be a credible human-machine interface that can establish non-verbal communication through learning and imitating the emotional behaviour of the interlocutor. As pointed out by Jaqueline Nadel, the process of imitation is innate to humans, and place a crucial role in distinguishing between actions arising from within or actions induced by others. Moreover, imitation establishes a reciprocal nonverbal communication process in which the roles of imitator and model are continuously exchanged. The research group believe that a truly biomimetic approach to implement human-like facial dynamics and behaviour is through a process of imitation-based learning. In order to establish imitation-based communication, FACE is human-like and possesses facial mimicry thanks to a muscular architecture. The signals that come from the sensory systems are fused within a sensory-motor map and processed by a neurocontroller. How efficient FACE is in reaching its objective thus depends not only on the control system but also on the efficiency of the sensory and actuating apparatus that it is equipped with. Thus the biomimetic features themselves and how they are integrated take on a key role in terms of FACEs believability and powers of non-verbal communication.

While the framework has been developed as a general purpouse instrument for the control of complex systems, FACE represents the main scenario where the architecture is placed and tested. Several applications have been implemented in order to make all the FACE's devices to be able to communicate each other and with control processes. During the first set of applications, the sensory-motor map was used to supply information to an automatic system in order to recognize facial expressions. The system consisted of a hierarchical neural network connected directly with the area of the sensory-motor map relative to the artificial vision system of FACE. The network architecture was based on a parallel self organising classifier followed by a predictive map. The integrate-and-fire model was used as the base neuronal entity. The network was able, after appropriate supervised training, to recognize the facial expressions of the human interlocutor in front of FACE. A behaviour based high-level control system has been developed for the FACET project, making the robot able to operate as a therapeutic tool for children with autism. Currently an architecture of the neurocontroller inspired by the structure of the hippocampus and allowing the navigation within a simplified behavioural space is under study. This architecture enables the development of models and simulations of entities and processes that are in line with the recent discoveries on the role of astrocytes in cognitive processes.

Acknowledgment

The author wishes to thank the Department of Electric Systems and Automation of the University of Pisa, the Interdepartmental Research Center “E. Piaggio” of the University of Pisa and the Institute of Clinical Physiology of the Italian Council of Research of Pisa.

Bibliography

[1] Baron-Cohen S., 1997. Mindblindness: an essay on autism and theory of mind. Cambridge, MA: MIT Pr.

[2] Bernstein NA., 1967. The coordinator and regulator of movement. New York: Pergamon Pr.

[3] Berthoz A., 1997. Le sens du mouvement. Paris: Editions Odile Jacob.

[4] Breazeal C., Scassellati B., 2000. Infant-like social interactions between a robot and a human caretaker. Adaptive Behav, 8:49-74.

[5] Brooks R.A., Breazeal C., Marjanovic M et al., 1999. The Cog Project: building a humanoid robot. In Nehaniv C, ed. Computation for metaphors, analogy, and agents (Lecture notes in artificial intelligence Vol 1562). New York: Springer. p 52-87.

[6] Carpi F., De Rossi D., inventors., 2003. Attuatore elettromeccanico contrattile a polimero elettroattivo con elettrodi deformabili elicoidali. Italian patent PI/2003/A/000043.

[7] Carpi F, De Rossi D., 2004. Dielectric elastomer cylindrical actuators: electromechanical modelling and experimental evaluation. Mater Sci Eng C. Forthcoming.

[8] Damasio A., 2000. The feeling of what happens. London: William Heinemann.

[9] Dautenhahn K., Werry I., Rae J. et al., 2002. Robotic playmates: analysing interactive competencies of children with autism playing with a mobile robot. In Dautenhahn K, Bond A, Canamero L et al, eds. Socially intelligent agents creating relationships with computers and robots. The Netherlands: Kluwer Acad Publ. p 117-24.

[10] De Rossi D., Carpi F., Lorussi F et al. 2002. Electroactive fabrics for distributed, conformable and interactive systems. IEEE Sensors 2002, The First IEEE International Conference on Sensors. 2002 Jun 12-14; Orlando, FL, USA.

[11] De Rossi D, Della Santa A, Mazzoldi A. 1999. Dressware: wearable hardware. Mater Sci Eng C, 7:315.

[12] De Rossi D, Lorussi F, Mazzoldi A et al. 2003. Active dressware: wearable kinesthetic systems. In Barth FG, Humphrey JAC, Secomb TW, eds. Sensors and sensing in biology and engineering. Berlin: Springer. p 379-392.

[13] De Rossi D., Osada Y., 1999. Polymer sensors and actuators. Berlin: Springer-Verlag. Duchenne de Boulogne GB. 1990. The mechanism of human facial expression. Cambridge: Cambridge Univ Pr.

[14] FaceIt® [computer program], 2003. Minnetonka: Identix Inc. Software development kit.

[15] Giszter SF, Moxon KF, Rybak I et al., 2000. A neurobiological perspective on humanoid robot. IEEE Intell Syst, 15(4):64-9.

[16] Hara F, Kobayashi H, Iida F et al., 1998. Personality characterization of animate face robot through interactive communication with human. First International Workshop in Humanoid and Human Friendly Robotics, International Advanced Robotics Program (IARP). 1998 Oct 26-27; Tsukuba, Japan.

[17] Lien J.J., 1998. Automatic recognition of facial expressions using hidden Markov models and estimation of expression intensity. Technical report CMU-R1-TR-31. Pittsburgh: Carnegie Mellon University.

[18] Lorussi F., Tognetti A., Carpi F. et al., 2003. Recruited dielectric elastomer motor units as pseudomuscular actuator. San Diego: EAPAD-SPIE. MIMICS (Materialises Interactive Medical Image Control System) [computer program]. 2003. Version 7.3. Leuven: Materialise NV. Segmentation and visualisation tools.

[19] Pei Q., Pelrine R., Stanford S. et al., 2003. Electroelastomer rolls and their application for biomimetic walking robots. Synthetic Met, 1356: 129-131.

[20] Robins B, Dautenhahn K, Boekhorst R et al. 2004. Effects of repeated exposure to a humanoid robot on children with autism. In: Proccedings of Universal Access and Assistive Technology (CWUAAT). 2004 Mar 22-24; Cambridge, UK. London: Springer-Verlag.

[21] Sarkar N., Northrup S., Kawamura K., 2001. Biologically-inspired control architecture for a humanoid robot. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2001). 2001 Oct 29-Nov 3; Maui, HI, USA.

[22] Schaal S., 1999. Is imitation learning the route to humanoid robots? Trends Cogn Sci, 3:233-42.

[23] Scilingo E.P., Lorussi F., Mazzoldi A. et al., 2004. Strain sensing fabrics for wearable kinaesthetic systems. IEEE Sens J. Forthcoming.

[24] Willis C., 2003. Android world [online]. Accessed 10 Feb 2004. URL: http://www.androidworld.com

[25] Lee K., 2000, IEEE 1451: A Standard in Support of Smart Transducer Networking, IEEE Instrumentation and Measurement Technology Conference, vol. 2, Baltimore, MD USA, May 1-4, 525-528.

[26] IEEE Std 1451.1, 1999, Standard for a Smart Transducer Interface for Sensors and Actuators - Network Capable Application Processor (NCAP) Information Model, Institute of Electrical and Electronics Engineers, Inc., Piscataway, New Jersey 08855.

[27] IEEE Std 1451.2, 1997, Standard for a Smart Transducer Interface for Sensors and Actuators - Transducer to Microprocessor Communication Protocols and Transducer Electronic Data Sheet (TEDS) Formats, Institute of Electrical and Electronics Engineers, Inc., Piscataway, New Jersey 08855.

[28] Lee K., Schneeman R., 2000, Distributed measurement and control based on the IEEE 1451 smart transducer interface standards, IEEE Transactions on Instrumentation and Measurement, 49(3).

[29] Steinberg A.N., 2001, Data fusion system engineering, IEEE Aerospace and Electronic Systems Magazine, 16(6).

[30] Luo R.C., Chih-Chen Y., Kuo L.S., 2002, Multisensor fusion and integration: approaches, applications, and future research directions, IEEE Sensors Journal, 2(2).

[31] Gutierrez-Osuna R., 2002, Pattern Analysis for Machine Olfaction: A Review, IEEE Sensors Journal, 2(3).

[32] Snopok B.A., Kruglenko I.V., 2002, Multisensor systems for chemical analysis: state-of-the-art in Electronic Nose technology and new trends in machine olfaction, Thin Solid Films, 418(1), 21-41.

[33] Brunet J., Pauly A., Mazet L., Germain J.P., Bouvet M., Malezieux B., 2005, Improvement in real time detection and selectivity of phthalocyanine gas sensors dedicated to oxidizing pollutants evaluation, Thin Solid Films, 490(1), 28-35.

[34] OpenGL, 2004, The Industry Standard for High Performance Graphics, http://www.opengl.org.

[35] MathWorks, 2004, Introduction to the Matlab Neural Network Toolbox, http://www.mathworks.com/products/neuralnet/.

[36] MIT GmbH, 2004, DataEngine software tool for data mining, http://www.dataengine.de.

[37] O'Reilly R.C., Munakata Y., 2000, Computational Explorations in Cognitive Neuroscience: Understanding the Mind by Simulating the Brain, MIT Press, Cambridge.

[38] Intel, 2005, OpenSource ComputerVision Libraty, http://www.intel.com/technology/computing/opencv/index.htm.

[39] Viola P. and Jones M.J., 2001, Rapid Object Detection using a Boosted Cascade of Simple Features, IEEE CVPR.

[40] Lienhart R. and Maydt J., 2002, An Extended Set of Haar-like Features for Rapid Object Detection, IEEE ICIP 2002, Vol. 1, pp. 900-903.

[41] Anderson, J.A., 1995. Introduction to Neural Networks. Cambridge, MA: MIT Press.

[42] Basilli, J.N., 1978. Facial Motion in the Perception of Faces and Emotional Expression. J. Experimental Psychology, vol 4 pp 373-379.

[43] Basilli, J.N., 1979. Emotional Recognition: the Role of Facial Motion and the Relative Importance of Upper and Lower Areas of the Face. J. Personality and Social Psychology, vol 37 pp 2,049-2,059.

[44] Chimenti, M., De Rossi, D., Di Francesco, F., Domenici, C., Pieri, G., Pioggia, G., Salvetti, O., 2003. A neural approach for improving the measurement capability of an electronic nose. Meas. Sci. Technol. 14, 815-821.

[45] Damasio, A., Damasco, H., Van Hoesen, G.W., 1982. Prosopagnosia: anatomic basis and behavioral mechanisms. Neurology, 32, 331-341.

[46] Desimone, R., 1991. Face-selective cells in the temporal cortex of monkeys. Journal of Cognitive Neuroscience, 3, 1-24.

[47] Ekman, P., Rosenberg, E.L., 1997. What the face reveals: basic and applied studies of spontaneous expression using the facial action coding system (FACS). Oxford University Press, New York.

[48] Ekman, P., 1989. The argument and evidence about universals in facial expressions of emotion. London: Handbook of psychophysiology. Wagner H, Manstead A, eds. John Wiley.

[49] Ersoy, O.K., Deng, S.W., 1995. Parallel, Self-organizing, Hierarchical Neural Networks with Continuous Inputs and Outputs. IEEE Transactions on Neural Networks, 6(5):1037-1044.

[50] Essa, I.A., 1995. Analysis, interpretation and synthesis of facial expressions. Perceptual Computing Technical Report nr 303. MIT Media Laboratory.

[51] Essa, I.A., Pentland, A., 1997. Coding Analysis Interpretation and Recognition of Facial Expression. IEEE trans on pattern analysis and machine intelligence 19(7).

[52] KinneBrock, W., 1992. Neural Networks. Munchen, Oldenburg Verlag.

[53] Kohonen, T., 1988. Self-Organization and Associative Memory. Springer, second edition.

[54] Kohonen, T., 1997. Self-Organising Maps. of Springer Series in Information Sciences. Vol. 30. Springer, Berlin, Heidelberg, New York, 2nd extended edition.

[55] O'Mara, D., 2002. Automated facial metrology. Ph.D. thesis, University of Western Australia, February.

[56] G. Pioggia, A. Ahluwalia, F. Carpi, A. Marchetti, M. Ferro, W. Rocchia, D. De Rossi, 2004, FACE: facial automaton for conveying emotions, Applied Bionics and Biomechanics, vol. 1(2), pp. 91-100.

[57] Pioggia G., Igliozzi R., Ferro M., Ahluwalia A., Muratori F., De Rossi D., An Android for Enhancing Social Skills and Emotion Recognition in Autistic Patients, IEEE Transactions on Neural Systems and Rehabilitation Engineering, Vol. 13, No. 4, December 2005.

[58] Casalini S., Pioggia G., Ferro M., Caudai C., De Rossi D., FACE e la sua mente, in "La Bioingegneria del Sistema Cervello-mente", cap. 5., Biondi Ed., Collana di Bioingegneria, Patron, in press.

[59] Rumelhart, D.E., Hinton, G.E., Williams R.J., 1986. Learning internal representations by error propagation. Parallel Data Processing, vol.1, Cambridge, MA: The M.I.T. Press, pp. 318-362.

[60] Rumelhart, D.E., McClelland, J.L., 1986. Explorations in the Microstructure of Cognition. Parallel Distributed Processing, volume 1 and 2 of Bradford Group. MIT, Cambridge, (MA.).

[61] Terzopoulos, D., Waters, K., 1993. Analysis and synthesis of facial image sequences using physical and anatomical models. IEEE Trans. on Pat. Anal. and Mac. Int., Special Issue on 3D Mod. in Image Analysis and Synthesis, 15(6): 569-579.

[62] Tranel, D., Damasio, A.R., Damasco, H., 1988. Intact recognition of facial expression, gender and age inpatients with impaired recognition of face identity. Neurology, 38, 690-696.

[63] Wilson, E.O., 1975. Sociobiology: The new synthesis. Cambridge, Mass., Harvard Un. Press.

[64] Young, M.P., Yamane, S., 1992. Sparse population coding of faces in the inferotemporal cortex. Science, 56, 1327-1331.

[65] S. Baron-Cohen, H. Tager-Flusberg, D. J. Cohen, 1994, Understanding other minds: Perspectives from autism. New York, USA: Oxford University Press.

[66] S. Davies, D. Bishop, A. S. R. Manstead, D. Tantam, 1994, Face Perception in children with autism and Aspergers Syndrome, Journal of Child Psychology and Psychiatry, vol. 35, pp. 1033-1057.

[67] G. Dawson, A. Meltzoff, J. Osterling, J. Rinaldi, 1998, Neuropsychological correlates of early symptoms of autism, Child development, vol. 69, pp. 1277-1285.

[68] I. M. Smith, S. E. Bryson, 1994, Imitation and action in autism: A critical review, Psychology Bulletin, vol. 116, pp. 259-273.

[69] J. Teunisse, B. Degelder, 1994, Do autistics have a generalized face processing deficit?, International Journal of Neuroscience, vol. 77, pp. 1-10.

[70] E. S. Spelke, A. Phillips, A. L. Woodward, D. Sperber, D. Premack, A. J. Premack, 1995, Infants Knowledge of objects motion and human action, in Causal Cognition: A Multidisciplinary Debate, Ed. Oxford, England: Oxford University Press, pp. 44-78.

[71] S . Maestro, F. Muratori, M. C. Cavallaro, F. Pei, D. Stern, B. Golse, F. Palacio-Espasa, 2002, Attentional skills during the first 6 months of age in autism spectrum disorder, J. Am. Acad. Child Adolesc. Psychiatry, vol. 41(10), pp. 1239-1245.

[72] .T. Schultz, I. Gauthier, A. Klin, R. Fulbrigth, A. Anderson, F.R. Volkmar, P. Skudlarski, C. Lacadie, D. J. Cohen, J. C. Gore, 2000, Abnormal ventral temporal cortical activity among individuals with autism and asperger syndrome during face discrimination, Arch. Gen. Psychiatry, vol. 57, pp. 331-340.

[73] A. Klin, B. A. Warren Jones, R. T. Schultz, F. R. Volkmar, D. Cohen, 2002, Visual Fixation Patterns During Viewing of Naturalistic Social Situations as Predictors of Social Competence in Individuals with Autism, Arch. Gen. Psychiatry, vol. 59, pp. 809-816.

[74] S. Baron-Cohen, H. A. Ring, E. T. Bullmore, A. S. Wheelwright, C. Ashwin, S. C. R. Williams, 2000, The amygdala theory of autism, Neuroscience and Behavioral Reviews, vol. 24, pp. 355-364.

[75] S. Baron-Cohen, 1997, Mindblindness: An Essay on Autism and Theory of Mind. Cambridge (MA), USA: MIT Press.

[76] S. J. Rogers, H. C. Lewis, 1989, An effective day treatment model for young children with pervasive developmental disorders, J. Am. Acad. Child Adolesc. Psychiatry, vol. 28, pp. 207-214.

[77] P. Howlin, 1998, Practioner review: psychological and educational treatments for autism, J. Child Psychol. Psychiatry, vol. 39, pp. 307-322.

[78] S. Wieder, B. Kalmanson, 2000, Educational guidelines for preschool children with disorders in relating and communicating, in Clinical Practice Guidelines: Redefining the Standards of Care for Infants, Children, and Families with Special Needs, ch. 13, Bethesda, MD, USA: ICDL (Interdisciplinary Council on Developmental and Learning Disorders) press.

[79] C. Kasari, 2002, Assessing change in early intervention programs for children with autism, J. Autism Dev. Disord., vol. 32(5), pp. 447-461.

[80] K. Dautenhahn, I. Werry, J. Rae, P. Dickerson, P. Stribling, B. Ogden, 2002, Robotic playmates: analysing interactive competencies of children with autism playing with a mobile robot, in Socially intelligent agents creating relationships with computers and robots, K. Dautenhahn, A. Bond, L. Canamero, B. Edmonds, Ed. Dordrecht, Netherlands: Kluwer Academic Press, pp. 117-124.

[81] F. Michaud, A. Duquette, I. Nadeau, 2003, Characteristics of mobile robotic toys for children with Pervasive Developmental Disorders, in Proc. IEEE Conference on Systems, Man, and Cybernetics, Washington, D.C., USA.

[82] F. Michaud, C. Théberge-Turmel, K. Dautenhahn, A. Bond, L. Canamero, B. Edmonds, 2002, Mobile robotic toys and autism, in Socially intelligent agents creating relationships with computers and robots, Ed. Dordrecht, Netherlands: Kluwer Academic Publishers, pp. 125-132.

[83] J. Nadel, A. Revel, P. Andry, P. Gaussier, 2004, Towards communication: first imitation in infants, children with autism and robots, Interaction Studies, vol. 1, pp. 45-75.

[84] J. Prag, R. Neave, 1997, Making Faces: Using Forensic and Archaeological Evidence. College Station (TX), USA: Texas A&M University Press.

[85] A. G. Feldman, 1979, Central and reflex mechanisms in motor control (translation from Russian). Moscow: IAPC Nauka/Interperiodoca.

[86] A. G. Feldman, 1986, Once more for the equilibrium point hypothesis, J. Mot. Behav., vol. 18, pp. 17-54.

[87] D. De Rossi, F. Di Puccio, F. Lorussi, P. Orsini, A. Tognetti, 2002, Feldmans muscle model: implementation and control of a kinematic chain driven by pseudo-muscular actuators, in Proc. 13th Conference of the European Society of Biomechanics, Wroclaw, Poland.

[88] R. Pelrine, R. Kornbluh, Q. Pei, J. Joseph, 2000, High Speed Electrically Actuated Elastomers With Strain Greater Than 100%, Science, vol. 287, pp. 836-839.

[89] Q. Pei, R. Pelrine, S. Stanford, R. Kornbluh, M. Rosenthal, 2003, Electroelastomer rolls and their application for biomimetic walking robots, Synthetic Metals, vol. 135-136, pp. 129-131.

[90] F. Carpi, P. Chiarelli, A. Mazzoldi, D. De Rossi, 2003, Electromechanical characterisation of dielectric elastomer planar actuators: comparative evaluation of different electrode materials and different counterloads, Sensors and Actuators A, vol. 107, pp. 85-95.

[91] J. Kozlowski, G. Serra, 1999, The Complex Phase Tracing method for fringe pattern analysis, Appl. Opt., vol. 38, pp. 2256-2262.

[92] G. Pioggia, M. Ferro, J. Kozlowski, A. Marchetti, F. Di Francesco, A. Ahluwalia, D. De Rossi, 2004, Automatic facial expression recognition by means of a neural approach, in Proc. of the 2nd International Symposium on Measurement, Analysis and Modeling of Human Functions ISHF2004, Genova, Italy.

[93] T. Kohonen, 1997, Self-Organising Maps, in Springer Series in Information Sciences, 2nd ext. ed. vol. 30, Berlin: Springer.

[94] D. E. Rumelhart, J. L. McClelland, 1986, PDP Research Group, Parallel Distributed Processing: Explorations in the Microstructure of Cognition. Vol. 1-2, Cambridge, MA: Bradford Group, MIT Press.

[95] S. Maestro, F. Muratori, F. Barbieri, C. Casella, V. Cattaneo, M. C. Cavallaro, A. Cesari, A. Milone, L. Rizzo, V. Viglione, D. D. Stern, F. Palacio-Espasa, 2000, Early behavioral development in autistic children: the first two years of life through home movies, Psychopathology, vol. 34(3), pp. 147-152.

[96] P. Thompson, 1980, Margaret Thatcher-A new illusion, Perception, vol. 9, pp. 483-484.

[97] M. B. Lewis, 2001, The Ladys not for turning: Rotation of the Thatcher Illusion, Perception, vol. 30, pp. 769-774.

[98] S. Baron-Cohen, S. Wheelwright, J. Hill, Y. Raste, I. Plumb, 2001, The Reading the Mind in the eyes test revised version: A study with normal

adults, and adults with Asperger Syndrome or High-Functioning autism, Journal of Child Psychiatry and Psychiatry, vol. 42, pp. 241-252.

[99] Diagnostic and statistical manual of mental disorders (DSM-IV), 4th ed., 1994, American Psychiatric Association (APA), Washington, DC.

[100] C. Lord, M. Rutter, A. Le Couter, 1994, Autism diagnostic interview-revised: a revised version of a diagnostic interview for caregivers of individuals with possible pervasive developmental disorders, Journal of Autism and developmental disorders, vol. 24, pp. 659-686.

[101] C. Lord, S. Risi, L. Lambrecht, E. H. Cook, B. L. Leventhal, P. C. Di Lavore, A. Pickles, M. Rutter, 2000, The autism diagnostic observation schedule-generic: a standard measure of social and communication deficits with the spectrum of autism, Journal of Autism and developmental disorders, vol. 30(3), pp. 205-223.

[102] Wechsler D., 1974, Manual for the Wechsler intelligence scale for children-revised (WISC-R), San Antonio, USA: The Psychological Corporation.

[103] Shimamura A., Jurica P., Mangels J., Gershberg F., Knight R., 1995, Susceptibility to Memory Interference Effects following Frontal Lobe Damage: Findings form Tests of Paired-Associate Learning, Journal of Cognitive Neuroscience, 7(2), 144-152.

[104] Sougnè J.P., French R.M., 2002, Synfire Chains and Catastrophic Interference, Annual Conference of Cognitive Science Society.

[105] O'Reilly R.C., Munakata Y., 2000, Computational Explorations in Cognitive Neuroscience: Understanding the Mind by Simulating the Brain, MIT Press: Cambridge.

[106] McCloskey M., Cohen N.J., 1989, Catastrophic interference in connectionist networks: The sequential learning problem, in G. H. Bower (Ed.), The psychology of learning and motivation, 24, 109165, New York: Academic Press.

[107] O'Reilly R.C., Norman K.A., McClelland J.L., 1998, A Hippocampal Model of Recognition Memory, M.I. Jordan, M.J. Kearns & S.A. Solla (Eds), Advances in Neural Information Processing Systems 10, 73-79, Cambridge, MA: MIT Press.

[108] Chimenti M., De Rossi D., Di Francesco F., Domenici C., Pieri G., Pioggia G., Salvetti O., 2003, A neural approach for improving the odour recognition capability of an electronic nose, Measurement Science and Technology, 14(6), 815-821.

[109] Gallazzi M.C., Tassoni L., Bertarelli C., Pioggia G., Di Francesco F., Montoneri E., 2003, Poly(alkoxy-bithiophenes) sensors for organic vapours, Sensors and Actuators B-Chemical, 88(2), 178-189.

[110] Hodgkin A. L., Huxley A. F., A quantitative description of membrane current and its application to conduction and exitation in nerve. Journal of Physiology 117, 500-544, 1952.

[111] Darwin Charles, 1859, On the Origin of the Species by Means of Natural Selection or the Preservation of Favoured Races in the Struggle for Life, Murray, London.

[112] Edelman G. M., 1995, Darwinismo neurale, Einaudi, Torino.

[113] Izhikevich E. M., 2000, Neural excitability, spiking and bursting. International Journal of Bifurcation and Chaos, Vol. 10, N°6, pp 1171-1266.

[114] Izhikevich E. M., 2005, Polycronization: computation with spikes. Neural Computation.

[115] Edelman G. M., 1991, Il presente ricordato: una teoria biologica della coscienza, Rizzoli, Milano.

[116] Edelman G. M., 1993, Topobiologia: introduzione all'embriologia molecolare, Bollati Boringhieri, Torino.

[117] Edelman G. M., 1993, Sulla materia della mente, Adelphi, Milano.

[118] Morris C., Lecar H., Voltage oscillations in the barnacle giant muscle fiber. Biophysiology Journal, vol. 35, pp. 193-213,1981.

[119] FitzHugh R., Nagumo, 1961, Impulse and physiological states in models of nerve membrane. Biophysiology Journal, vol. 1, pp. 445-446.

[120] Izhikevich E. M., 2004, Which model to use for cortical spiking neurons?. IEEE transaction on neural networks, vol. 15, n° 5.

[121] Izhikevich E. M., 2003, Simple model of spiking neurons. IEEE transaction on neural networks, vol. 15, n° 6.

[122] Izhikevich E. M., Gally J. A., Edelman G. M., Spike-timing dynamics of neural groups. disponibile sul sito: http://www.izhikevich.com.

[123] Turrigiano G.G., Leslie K.R., Desai N.S., Rutherford L.C., and Nelson S.B., 1998, Activity-dependent scaling of quantal amplitude in neocortical neurons Nature, 391:892-896.

[124] Whittington, M. A., Traub, R. D., Kopell, N., Ermentrout, B., and Buhl, E. H., 2000, Inhibition-based rhythms: experimental and mathematical observations on network dynamics. Int. J. of Psychophysiol., 38:315-336.

[125] Miller R., 1996, Cortico-thalamic interplay and the security of operation of neural assemblies and temporal chains in the cerebral cortex. Biol Cybern 75 3: 263-267,(b).