ALMA MATER STUDIORUM - UNIVERSITÀ DI BOLOGNA
CAMPUS DI CESENA

Scuola di Ingegneria
Corso di Laurea Magistrale in Ingegneria e Scienze Informatiche

# ONLINE ADAPTATION OF ROBOTS CONTROLLED BY NANOWIRE NETWORKS

Elaborato in:

Intelligent Robotic Systems

| | |
|---|---|
| *Relatore:* | *Presentata da:* |
| Prof. Andrea Roli | Paolo Baldini |
| *Correlatore:* | |
| Dott. Michele Braccini | |

Anno Accademico 2020/2021

*To my family. To my girlfriend.*

# Contents

# Introduction

Since few decades, researchers have started to evaluate the possible usage of systems inspired by the human brain. This interest comes from the will of transposing the computational power and flexibility of this organ into artificial devices. Although different in types, the resulting systems are considered part of what is called Artificial Intelligence. Among them, lies the class of physical networks. This includes all the computational systems whose hardware aims or is used to mimic a biological neural network. Specifically, one subset is represented by electrical circuits. Nevertheless, their use had never been one of the main solution in the Artificial Intelligence. This is due to the hardly trainable state and mechanics. In the last decade however, a further subset of this category started to emerge, and to be assessed due to its high dynamical and neuromorphic properties: memristive networks. Their characteristics combined with novel computational approaches [19, 24] are paving the way for new types of computation. One promising area of application is robotics. The need is indeed for cheap and low consumption controllers to be developed. For this goal, electrical memristive networks seems to be an interesting and promising opportunity.

In this dissertation, the use of one of these systems is evaluated: the *nanowire network*. Its exploitation follows an approach derived from what is called Reservoir Computing [30]. The goal is to assess if these devices have the capabilities to be used in the robotic area. In fact, only classical computing tasks have been considered so far. Nevertheless, the reduced cost, power consumption and neuromorphic properties, make them potentially optimal for the development of a new generation of robots. The vision that this work follows is about futuristic artificial entities being able to adapt through the use of intrinsic properties, while being cheap and ecological. Therefore, the aim is to evaluate if this novel technology possess the characteristics for a paradigm change, that, in robotic, no one has yet explored.

The coveted evaluation requires various steps. The first consists in studying the network and preparing the robotic system. This explores the physical and chemical principles that enable the previously presented properties. The result is the definition of a simulator of the memristive device. Through its use, a behavioural analysis of the network is conducted. This allows to clarify the mechanisms for robot adaptation. The goal is to enable an adjustment according to the changes in the environment. This is a needed step towards the realisation of the project vision. Finally, all the previous results converge to the definition of the robot architecture, both from an architectural and biological point of view.

With the envisioned entity ready, the next step is its experimental assessment. The goal is to verify if the created system really possess the capabilities to be the starting point for a new generation of robots. To assess this question, it is firstly needed to assess its performance in simple tasks. This is evaluated through a basic collision avoidance, and integrated by an area avoidance assignment. Additionally, the phenotypical plasticity of the robot is assessed. This represents its ability to adapt to different environments and tasks. Finally, the last tests verified the performances in a memory oriented task. The system is thus evaluated to understand if he can exploit its internal dynamics to generate an endogenic awareness of time.

As final point of the thesis, possible future works are presented. All of them are based on the results obtained, or are inspired by the approach followed.

This thesis is organised as follows:

- Chapter 1: Nanowire networks
  The network is described, and its working principles are introduced from the physical and chemical point of view. The potential of this technology is assessed, also considering the economical aspect. From the production perspective, the influence of the density parameter is briefly discussed. Finally, the simulator used in the project is presented, highlighting its characteristics and limits.

- Chapter 2: Online adaptation
  The online, adaptive learning approach is presented, together with its strength points. Additionally, the concepts of phenotypical and developmental plasticities are discussed. Finally, a consideration on the use of Reservoir Computing in online methodologies is given.

- Chapter 3: Preliminary analysis
  The characteristics of the network are evaluated and studied. Particularly, the existence of a critical working state and an endogenic memory is searched. Additionally, the influence of external components in the network behaviour is assessed.

- Chapter 4: Robotic architecture
  The robotic control architecture is proposed from a biological and technical point of view. Finally, a unified, software version is presented.

- Chapter 5: Experimental setting
  The common characteristics of the experiments are grouped and explained. This includes (*i*) the selection of some parameters for the robot functioning; (*ii*) the definition of methodologies to ensure statistical reliability; (*iii*) the description of the adaptive approaches.

- Chapter 6: Experiment: collision avoidance
  The first experiment is described, performed and assessed. Specifically, the densities, loads, and adaptive strategies are evaluated according to their influence in the results. Additionally, a qualitative analysis of a successful instance is performed.

- Chapter 7: Experiment: area avoidance
  The second experiment is described, performed and assessed. The aim is to demonstrate a more reactive behaviour and to ensure phenotypical plasticity. The latter one is confirmed through a comparison with the same networks optimised for the collision avoidance. The densities and the loads are evaluated according to their influence in the results.

- Chapter 8: Experiment: T-maze
  The third and last experiment is described, performed and assessed. The main goal is to verify if the network dynamics can be exploited as an endogenic memory. A preliminary, manual test is described and explained. The results are then analysed according to the influence of densities, loads and adaptive strategies.

- Chapter 9: Future works
  Possible future works originated or inspired by this dissertation are presented. The main topics are improvements in: (*i*) the adaptive mechanics; (*ii*) technical aspects; (*iii*) experiments; (*iv*) analysis; (*v*) the nanowire network knowledge.

# 1 Nanowire networks

The nanowire networks are electrical circuits composed of nanometric wires. Those overlap, creating a graph-like structure. The wires are immersed in an insulating substrate, that allows the current to flow in the network without 'jumping' between the arches. In this dissertation, this specific category of electrical circuits is considered, mostly due to its memristive-like behaviour. Those devices can indeed exhibit structural and functional plasticity, mimicking biological networks [29]. Until now, the research on these systems mostly considered the use of crossbar architectures (see Figure 1). Nevertheless, randomly-created and self-organised devices are being studied and their computational capabilities assessed [26, 29].

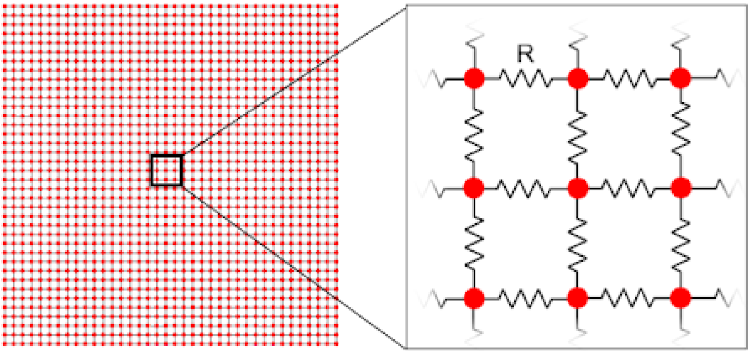

Figure 1: Circuital equivalent of a nanowire network organised according to a crossbar architecture. Pictures taken from [29] by courtesy of the authors.

The specific systems here considered consist in randomly generated networks composed of nanometric silver-wires, immersed in an insulating substrate and connected through ion-silver bridges [29] (see Figure 2). These latter ones are also called 'atomic' or 'resistive' switches, and cause the non-linear behaviour of the device [6, 15]. Their working principle depends on an applied voltage difference, that causes an attraction and aggregation of ions, creating the ion-silver bridges and increasing the junction conductivity. When the stimulation is removed, the network slowly returns to its stable state. This dynamic determines a network plasticity and can be seen as a short-term memory. The conductivity depends indeed on the recent history of the stimulation. From this

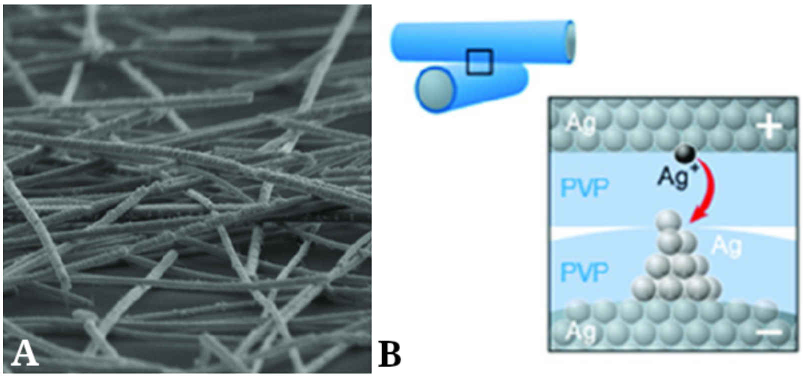

Figure 2: The picture (A) represents a random nanowire network. The figure (B) shows the process of formation of a silver-bridge between two junctions. Pictures taken from [29] by courtesy of the authors.

point of view, the network is strongly neuromorphic [18], with the conjunctions that represent the synapses.

One limiting factor in mimicking the brain computational model consists in the high quantity of neurons and synapses that it is composed of. This huge amount of connections is infeasible to be simulated by any modern computer. The nanowire networks offer a theoretical solution to this problem. The quantity of connections depends indeed only on the number of wires, their length, and the size of the device package. Moreover, they function in parallel[1]. These characteristics make the number of connections, and thus the size of the network, irrelevant with regard to the computational time. These systems are thus fully scalable. Additionally, the memristive behaviour determines a fading memory property, and advantages the neuromorphic computation. This allows spatio-temporal data to be effectively processed, making these systems more similar to the human brain [9].

Nevertheless, from an electrical point of view, the nanowire networks are passive components. This means that they do not proactively supply energy, but instead they consume it. The system thus needs and external signal to produce an output. The computational capabilities are indeed related to a process of mixing and transforming the input data, and not to an active behaviour.

[1] The systems are subject to the electric laws, and thus to the Kirchhoff's currents one

Moreover, the networks do not store the signals that they receive. Instead, they dissipate them immediately. From this point of view, the nanowire networks are completely dissipative components. This highlights a difference with the brain, that instead keeps the information for some time also in absence of inputs.

Finally, the use of these devices may result in a reduction of costs for the creation of biologically inspired neural networks. The creation only requires drop-casting nanometric wires in an insulating substrate, making the production low-cost [9, 29]. Moreover, although not assessed, exploiting specific hardware is usually more efficient than using general purpose ones (e.g., software neural networks). This suggests that the use of this technology may lead to an improvement in the overall power consumption. The possibility is supported also by the intrinsic memory of the networks. A downside of a classical Von Neumann architecture consists indeed in having the data and the computing areas separated, with the information transfer responsible for remarkable consumes [9]. Novel computational frameworks exploiting the local memory provided by the nanowire networks may thus decrease the overall computational cost by getting rid of this overhead.

## Device parameters

We previously explained how the connections of the network depend on the number of wires, their length, and the size of the device package. In this dissertation, these parameters are ignored, considering instead a derivate measure: the normalised density. This is calculated according to the following formula:

$$D = N\frac{L^2}{S^2} \tag{1}$$

Where:

- D is the normalised density;
- N is the number of nanowires;
- L is the mean length of the wires;
- S is the size of one side of the squared package.

The normalised, creation density determines the emergence of connected components. A critical point is found around the value 5, when most of the nodes start to be connected together[2].

## The simulator

The nanowire network here discussed was not physically available. The experiments used instead a simulator of the device[3]. Here, the network is represented as a graph, with the nanowires associated to nodes, and the junctions to arches. The voltage distribution is calculated with the Modified Nodal Analysis, or MNA [17]. The non-linear dynamic of the junctions is modelled by a custom algorithm. This primary simulator was thus expanded for its use as the robot controller. This involved the possibility of connecting multiple sources (i.e., the sensors), external loads (i.e., the motors), and to perform mutations of the set of nodes selected as input/output.

The use of a simulator brought some advantages. Differently from a physical device, its software representation allows for direct analysis of the network state. In hardware, this would require some shrewdness and tedious work. Moreover, it enables to easily test multiple configurations, not being limited by a single instance. This is also useful for collecting statistical data. As a drawback, however, the simulation requires much more computational time. This causes the need for shorter simulations to be executed. Finally, the use of the simulator to control a physical robot results problematic. One of the aims was initially to test the adaptive approach in the real world. This would require a real time information flow between a physical robot and the simulator. Moreover, it would need a fast computation of the model. If the first problem is manageable, this is not true for the second. The update frequency of the simulator is normally too low to smoothly drive the robot. The idea had thus to be discarded.

[2] These experimental data were informally provided by Carlo Ricciardi and Gianluca Milano from the Polytechnic of Turin

[3] This consists in an upgrade of a previously existing model provided by Gianluca Milano from the Polytechnic of Turin

# 2 Online adaptation

At the start of the robotic era, the automata were defined through physical mechanisms that allow the desired behaviour to emerge. In some cases, complex gear structures were used[4], in others, the body shape was designed to directly allow actions when immersed in a suitable environment [10]. The control system of these devices was thus a mixture of mechanical properties. With the advancement of technology and the introduction of silicon components, the definition of the robot behaviour started to be expressed through programming. The design is still complex, but surely easier than before. In the last decades, we are witnessing an additional change in the way we define how a robot behaves. Although the programming approach is still the most important one, automatic design techniques are emerging, with the 'evolutionary robotic' being a notable example [13]. This allows to create working robots also when a clear logic solution is unknown or extremely hard to process [3]. Additionally, if correctly performed, it allows the robot to be more resistant to noise. The affected areas are the design of the body-shape [33] and the control logic [13, 4].

One of the key points that differentiates the biological beings from the robots is the ability of the firsts to behave in dynamic and unknown environments. This capability partially derivates from the high computational power of the brain, that can elaborate a high quantity of ill-conditioned signals [28]. Nevertheless, the 'killer feature' comes from its *phenotypical* and *developmental plasticity* [4]. These respectively represent the ability of the brain to produce different phenotypes according to the environment and the development history of the living being. These definitions underline an aspect of the natural world that is often ignored in the robot development: a continuous adaptation. Current engineering processes typically consist in the definition of static control mechanisms. Also in case a learning technique is exploited, it is frequently used to perform a training before the deployment. This type of process is said to be *offline*. The results of these techniques consist in the realisation of very specialised and efficient robots, that are however bound to the world that they are developed for (see Figure 3 B). Indeed, the system is usually trained on a

[4] Notable examples are the Jaquet-Droz automatas [35] and the Karakuri puppets [36]

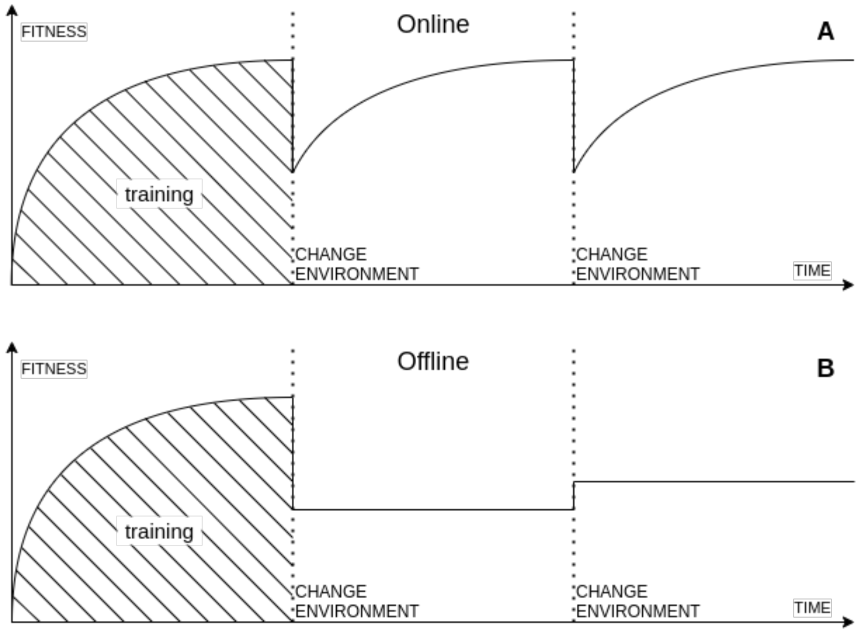

Figure 3: Concept of the learning techniques. In an *offline* learning (B), the robot cannot change its control software, and its performances are thus related to the environment. In an *online* learning (A), the robot is able to adapt and, after a training period, should be able to improve its performances. The figure is only for explanation and does not represent data of real runs. Moreover, the performance of the online approach may still be related to the environment, causing the adaptation to perform better or worse depending on the case. Nevertheless, the result should still consist in an improvement, compared to the offline approach.

limited set of common states, avoiding extreme situations. This means that the presence of unexpected noise or events may affect the behaviour or the performance of the robot. Therefore, the most common and effective use of these approaches is to create precise robots that have to work in specific, typically static, niches. Nevertheless, this methodology does not fit for all the tasks. Specifically, behaviours in very noisy and unpredictable environments may be too hard to train or define using these approaches. Additionally, the ‘ecological niche’ of the robot may change in time. In order to resolve these problems, a different methodology may be exploited: *online* learning. Contrarily to the offline approach, this contemplates a continuous adaptation of the robot. The advantages are theoretically more working autonomy and adaptability, being able to automatically modify the behaviour to face changes in the environment. Nevertheless, downsides reside in the difficulty of managing the adaptation and in the possible problems that can occur during it. Wrong choices may indeed lead to irreversibly damage the robot. To face or mitigate this issue, usually the systems are partially trained in simulation and then moved to the hardware when the results are sufficiently good (see Figure 3 A).

The experiments carried on in this dissertation will not take place in the real world due to the unavailability of the physical control system[5]. Instead, we will consider a continuous adaptation in a simulated environment. This approach is in line with a series of researches about online learning in robotics, using Boolean Networks [4, 1, 23]. The addition to the literature will be the use of a novel control mechanism: the nanowire networks. Additionally, the influence of different types of adaptive approaches will be evaluated. The aim is to define a robotic system that is able to automatically adapt its phenotype according to the working environment and to the selective pressure to which it is subjected (i.e., the objective function). This requires the definition of an adaptive strategy that is embeddable into the robot itself. In this treaty, *phenotype* is meant as the behaviour of the individual, in that it is the visible aspect that emerges from the *genotype* (i.e., the network) through its interaction with the external world. The phenotypical and developmental plasticities are instead representative of the adaptation of the robot to the environment and the task. The first manifests in the adjustment of the input pre-processing (e.g., using multipliers; see Chapter 4 and 5) according to the environmental properties. The second gives adaptive pressure to the robot, requiring a modification towards the behavioural goal. Both use the same approaches of re-connection and weighing of the inputs (see Chapter 5), but the goals are different.

In this context, the *neural short-term plasticity* of the nanowire networks represents a factor in the background. This refers indeed to the ability of the

[5] The problems related to the use of the nanowire network simulator with real robots are discussed in Chapter 1

network to modify its behaviour at runtime for a short period, in order to adapt to the stimulus [29]. Compared to the other kinds of plasticity, it is visible that this does not represent a long term adaptation.

The final issue about the use of online learning and adaptation in this research, relates an architectural point. The robot controller is indeed a variant of the Reservoir Computing framework [30], in that it requires to exploit a non-modifiable, non-linear system: the nanowire network. Nevertheless, the original idea requires a readout to be trained to match an expected result. This is obviously in contrast with the aim of a continuous learning, in that the objective output is not known. Therefore, the readout is substituted by a fixed post-processing. The robot is thus trained by re-wiring the input connections to the network. This approach moves the training from the output, that requires a goal to be matched, to the input, that instead just needs feedback of the performance. An online adaptive approach can indeed search for the best connection that allows the robot to perform well. Therefore, in some ways, this research will also propose a new variant of Reservoir Computing for its use with online adaptation.

# 3 Preliminary analysis

To the best of the author's knowledge, the usage of a nanowire network as the control system of a robot has never been evaluated. A preliminary analysis of the characteristics of this technology is thus required. The goal is to understand how the nanowire network behaves and which parameters influence it. The analysis of the latter ones, will be limited to the variables effectively modifiable in the robotic architecture.

## Paths stimulation

The main characteristic of the studied network is the property of changing its conductance when subject to a voltage. During the stimulation, the input potential insists on the junctions between the wires. As described in Chapter 1, this causes the formation of silver-bridges and a consequent increment in the edge conductance. The influence of the stimulation on a junction depends (*i*) on its position in the network topology and (*ii*) on its resistance 'influence' in the input/output path. Specifically, according to the voltage divider, equivalent resistors in parallel branches are less subject to be excited. At the same time, junctions in short input/output paths are more sensible to the stimulation. This causes a complex distribution of the conductance across the network.

To understand the principles behind this distribution, the first step consists in analysing the influence of the resistors in the equivalent electric circuit (*ii*). In agreement to the voltage divider, the electric potential subdivides between the nodes according to the conductance of the links. Junctions that are highly resistive will be subject to higher voltages, that cause a faster stimulation (see Figure 4). The result is an oddly stimulated network that changes its state in an inhomogeneous way.

The second consideration relates the position of the nodes in the circuit (*i*). Junctions nearer to the source tend to be more subject to the stimulation. This is due to the fact that the network have a tendency to branch, increasing the parallel resistors as we move away from the input node[6]. This causes a

[6] This may not be true in low dense networks, where the branches often merge back together

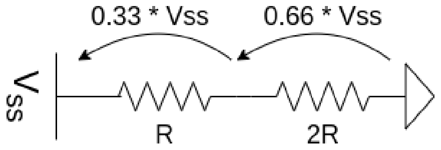

Figure 4: The distribution of the voltage, and thus of the stimulation, across the resistances of the network follows a circuital law. In the given schema, the 2R resistor will undergo a higher stimulation, causing its resistance to drift more than R. This behaviour obviously requires considering the resistors as plastic junctions.

reduction of the equivalent resistance of farther, branched junctions, decreasing the voltage that insists on them and thus their stimulation. Nevertheless, this property depends on the topology of the network, with denser ones more subject to this phenomenon.

The two described principles concur in the definition of higher level properties of the network. One of those is the influence of the input/output path length in the voltage distribution. Shorter path are indeed composed of fewer branches and resistors. The first, determines a more homogeneous stimulation in that the currents are less divided in the circuit. The second, causes the input voltage to fall across a smaller amount of junctions, increasing the stimulation of each of them.

Until now, the distribution of the stimulations has been discussed from a static point of view. Nevertheless, it is important to remember that the plasticity of the network adds a dynamic. The continuous stimulation is thus an essential factor to consider. In time, highly stimulated junctions become more conductive, causing a continuous reduction of their influence in the voltage divider. This consequently causes a decrement in their stimulation, favouring others. Therefore, the network presents a complex, dynamic behaviour that also depends on the duration and possible frequency[7] of the inputs. The capability of the junctions to relax between the stimulations plays indeed another primary role in defining the system behaviour. This relation will be further analysed in

[7] If pulsed inputs are used

Sections ‘Update frequency’ and ‘Critical state’.

## Connected loads

In the project, the controller interacts with the components of the robot through direct connections. Differently from other kind of systems, the inputs and outputs do not just transport data, but they also passively influence the network behaviour. The circuital equivalent of the controller is indeed subject to the laws of Kirchhoff and Ohm. Therefore, the addition of external connected components changes the electrical state of the whole system.

The main influence on the nanowire network are the motors. Those may be seen as additional loads of the circuit, changing its voltage distribution (see Figure 5). When the resistance is low, just a small amount of tension falls at the sides of the motor. This causes the network to be highly stimulated, in that the majority of the electric potential influences the device. Differently, when the connected load increases, also the voltage on the motor does (see Figure 6). This causes the controller to be less stimulated and thus to show lesser changes in the conductance. In other words, the network is less dynamic. This may be seen as a decrease in the synaptic plasticity.

Nevertheless, the analysis of the motor load is not limited to its influence in the network stimulation. With a decrease in the resistance, the robot is subject to a generalized slowing down[8]. This is again due to the voltage divider. Indeed, as the motor resistance decrease, as it does its influence, moving electrical potential from the load to the network. The result is lesser power at the output nodes. Therefore, the general idea is that an increase in the load determines faster movements, and vice versa.

The previously presented properties compete against each other: to have high sensitivity we have to sacrifice the output speed, and vice versa (see Figure 7). Given a complex task that needs both, a solution consists in the disjunction of the computation and actuation parts. This would allow module specific configurations and parametrizations, and thus the achievement of the desired state. Nevertheless, the hardware solution becomes complicated, requiring

[8] If a non-negated output connection is used

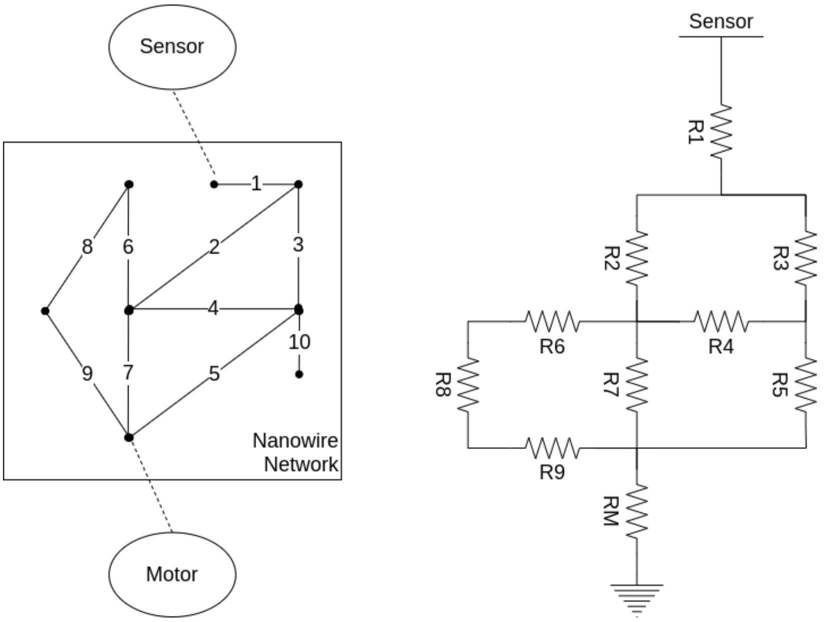

Figure 5: Circuital representation of the nanowire network. To improve the readability, the lines on the left represent the nanowires junctions. These are directly represented as resistances on the right figure. Note that the line 10 is not part of an input/output path, and thus it is ignored in the circuital scheme. Moreover, the motor is substituted by its equivalent resistance, underlining its influence in the network behaviour: the larger the load, the less the stimulated is the controller.

additional components to be installed. A more adaptive approach is to find a balance of the two properties.

## Network density

Another aspect to evaluate is the influence of the network creation density in the computation. This directly affects the number of the junctions, and thus the *cardinality* of the state space[9]. The results are a higher probability to find good scoring solutions, and an increase in the exploitable memory of the network.

[9] Analog devices can assume infinite values. Therefore, the increase in the state space has to be evaluated through the cardinality, comparing the dimensions of infinities

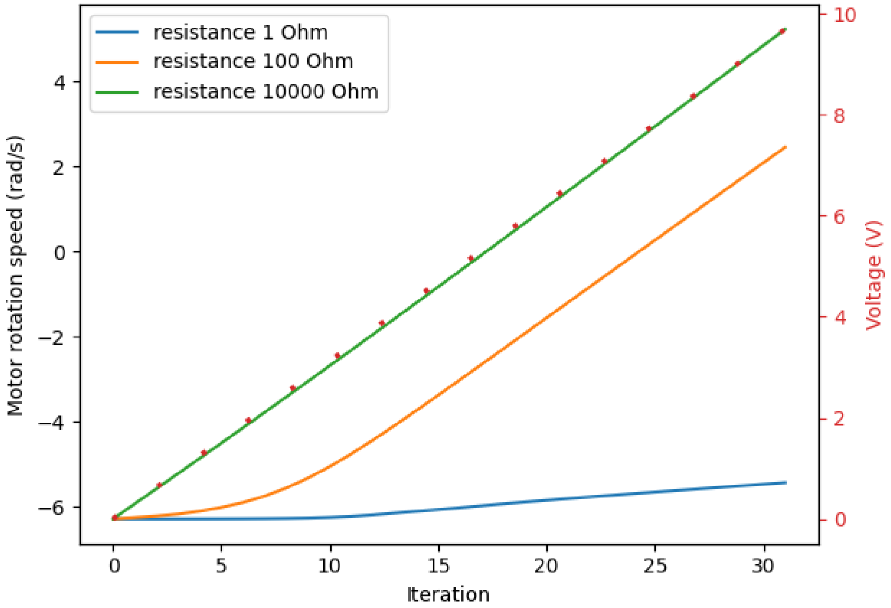

Figure 6: Qualitative relation between input (i.e., the red dotted curve) and output control signals, according to the motor load. It is visible that the slope stays low for small resistances, and increases for higher ones. This is due to the voltage divider, that moves electrical potential on the motor.

Nevertheless, the complexity of the search also increase[10].

From a technical perspective, the first consequence of increasing the network density is a wider parallelisation of the equivalent electrical circuit. This is due to the higher number of connections for the single wire. The result is a reduction in the overall network resistance. Consequently, the influence in the voltage divider decreases, causing a lower stimulation of the device[11].

A trend related to an increase of the network creation density, consists in a general dissolution of well-defined stimulated paths (see Figure 8). The cause is the greater amount of consecutive junctions separating an input from an output. Additionally, the increased parallelisation makes the network less stimulated. This causes a reduction in the electrical potential insisting on the

[10] Also, the simulation of the network increases in complexity, needing to simulate a much larger equivalent circuit

[11] If the same motor load is used

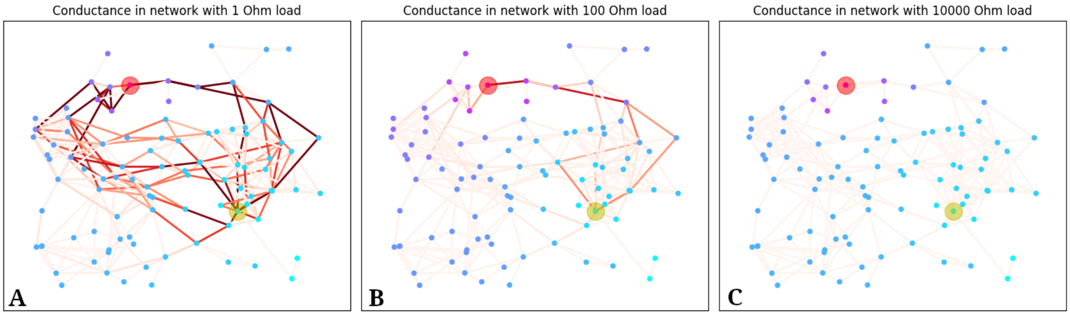
Conductance in network with 1 Ohm load
Conductance in network with 100 Ohm load
Conductance in network with 10000 Ohm load
A
B
C

Figure 7: Visual representation of the balancing between stimulation and output intensity of the system. With a low motor load (A), most of the electrical potential falls on the network equivalent circuit, making it more sensible to inputs but decreasing the output intensity. This is opposite in case of high motor resistances (C). For most tasks, a balance (B) may be the optimal solution.

single edge, and thus in the variation of its conductance[12]. The result is a lower, overall network sensibility[13] (see Figure 9).

Until now, the increment in the network density seems relatively useful for the computation. Nevertheless, the most interesting opportunities are related (*i*) to the theoretical increase of the network memory, and (*ii*) to the possible improvement in multi-signal analysis. The first (*i*) assumes that an increment in the density leads to a wider state space. This would allow an increase in the memory accessible by the tasks. The second (*ii*) hypothesizes that an increment in the amount of 'brain-pathways' may help in the transformation and exploitation of the sensory signals, possibly improving the network computational capabilities. Nevertheless, a systematic assessment of these properties is not explored in this research, and it is instead postponed to future works.

A final consideration is that the density should be carefully chosen for each task. Specifically, the need of memory and a high quantity of input signals may require a more complex network to be used. Additionally, in case a simulated controller is exploited, also the computational complexity should be considered.

[12] The stimulation curve is not linear

[13] A high voltage on a single resistor is more effective compared to a distribution across many junctions

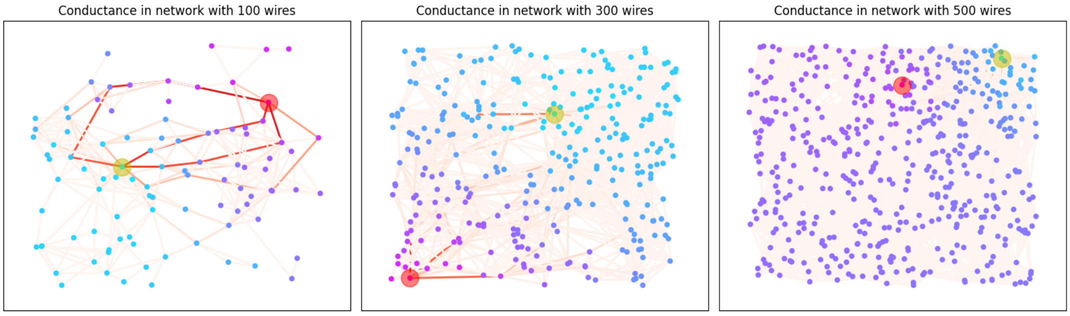
Conductance in network with 100 wires
Conductance in network with 300 wires
Conductance in network with 500 wires

Figure 8: Dissolution of well-defined paths as the network density increases. This is due to a general reduction in the single junction stimulation.

This is not a problem if physical devices are available.

## Update frequency

The nanowire network can be stimulated with inputs provided in two different ways: continue and periodic. The first one considers the data as continuously streamed. The second allows the signals to influence the network at regular intervals. In the simulator, the first approach is not implementable due to the working mode of the computer. The simulation of the continuous mode requires the periodic one to be run with frequencies higher than the perceivable excitement and relaxation time of the junctions[14]. Relatively low values shown to be enough, due to the low decay rate of the network conductance.

The update rate concept is of primary importance, in that determines some behavioural properties of the system. For high frequencies, we can assume no changes, with the conductance oscillation that remains almost the same. This is not true for low values. The idea is that slower update rates give the network time to relax. The excitement decay combined with a non-linear stimulation curve (see Figure 10) determines differences in the change of the input/output path conductance, and in its distribution across the junctions. Therefore, the main topic consists in understanding how the update frequency influences the

[14] The adjective *perceivable* is important in that a 'higher frequency' is physically impossible: in one case we talk about an analog signal, that by definition is not made of pulses

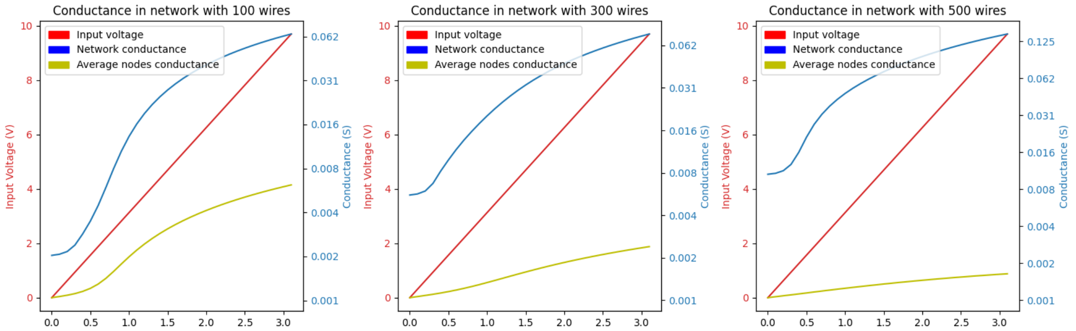

Figure 9: Variation of the network conductance according to the creation density. This is calculated on the input/output path. Also, the change of the average conductance of the junctions is plotted. The curves are obtained by a stimulation with an increasing input voltage. Note that the network conductance increases according to the density. This is due to a parallelisation of the equivalent resistors. Nevertheless, the variation due to the stimulation decreases. This can be perceived as a reduction in the network sensitivity.

evolution of the network state. Additionally, this may lead to the search of phase transitions depending on specific update rates (see Section ‘Critical state’).

To understand the influence of the frequency on the network behaviour, the tests analysed the changes of the motor speed[15] according to an increasing sensory input at different update rates[16]. The results highlight a substantial difference in the response to the stimuli, with lower frequencies showing more irregular outputs (see Figure 11). This is due to the property of the network to relax towards its stable state, changing the balance of the voltage divider. The result is a consequent change in the stimulation intensity, and thus in the network conductance. This property is visible only when the frequency is low enough. In these situations, the non-linearity of the relaxation and stimulation curves cause the irregular output behaviour.

The influence of the stimulation frequency is also visible from the point

[15] This is directly related with changes in the intensity of the network output signal

[16] The test used a network connected with a motor load of 100Ohm. This information is needed in that the behaviour also depends on the connected load

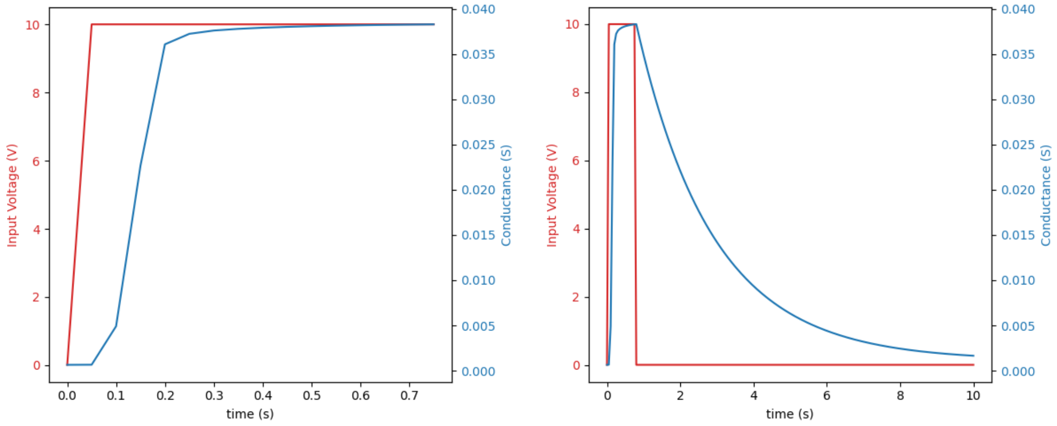

Figure 10: Dynamics of the input/output path conductance. The non-linearity of the behaviours in time is visible.

of view of the changes in the conductance distribution in the network. As already said (see Section 'Paths stimulation'), when a junction decreases its resistance, the voltage that will insist on it at the next step decreases as well. This usually causes less stimulated edges to change more in a subsequent stimulation. This behaviour seems to extreme when the frequency becomes low, with the conductance distribution that basically 'flips'[17] (see Figure 12). The result is the dissolution of well-defined paths, making the network state appear more chaotic. Contrarily, the use of high frequencies causes the input/output paths to excite in a more homogeneous and continuous way (see Figure 13). The result is a higher and more stable output[18].

## Separation and fading memory

As explained in Chapter 2, the control architecture is devised on a variant of the Reservoir Computing framework [30]. Nevertheless, to be used as a *reservoir*, the nanowire network needs to have some characteristics: (*i*) separation

[17] This is again related to the non-linearity of the excitement and relaxation

[18] With an increment in the conductance, the voltage 'moves' towards the load, that has a fixed resistance. This makes the output signal more intense. The improved stability is due to the better defined input/output path

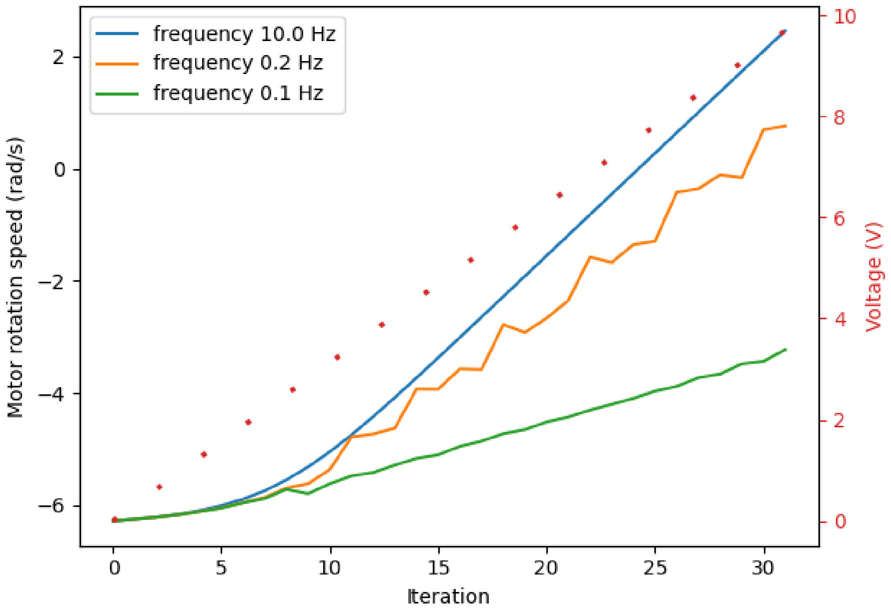

Figure 11: Motor speed according to the update frequency. The graph was generated providing a network with an increasing input signal, shown in red. When stimulated at high frequencies (i.e., 10Hz) the output shows a smooth increment. When the update rate is lower, we can instead observe an irregular behaviour. The least stable signal is found for values around 0.2Hz. This behaviour is analysed in the Section 'Critical state'.

property; (*ii*) fading memory[19] [24]. The first (*i*) requires that given different inputs, the internal trajectory of the system differs. The second (*ii*) requests a short-term memory of the network, meaning that old inputs should persist in the system state for some time.

The separation property is usually required in that the readout should be able to distinguish differences in the inputs just by looking at the reservoir state[20]. This is not really needed in our approach, in that the readout is just a direct connection between the network nodes and the actuators. The only post-processing is a negation of the output signal, but this does not involve any

[19] Also called *echo state property* [19]

[20] The readout should also be able to discriminate between different signals and noise. This capability is called approximation property [24]

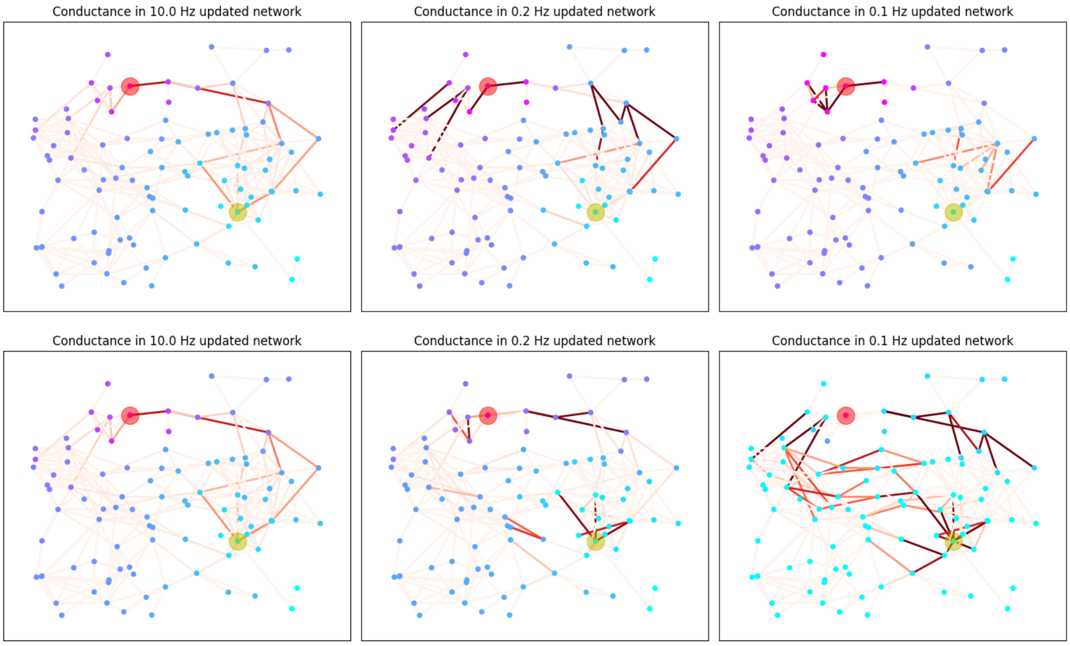
Conductance in 10.0 Hz updated network
Conductance in 0.2 Hz updated network
Conductance in 0.1 Hz updated network
Conductance in 10.0 Hz updated network
Conductance in 0.2 Hz updated network
Conductance in 0.1 Hz updated network

Figure 12: Changes in the network stimulation at different frequencies. The upper graphs show the penultimate stimulation; the bottom ones the last. It is visible that, as the frequency decrease, the network shows a more irregular behaviour. The junctions' conductance starts indeed to 'flip'.

high-level elaboration. Therefore, the most interesting property to verify is the fading memory. Some experiments of this research require indeed some sort of awareness of the past to succeed (see Chapter 8). Because of that, the analysis of the network memory capacity is essential for the continuation of the project.

The goal of the following experiments is to assess if the network ends up in different states depending on the input signal, thus showing the separation (*i*) and fading memory (*ii*) properties. The first (*i*) can be verified by checking the network conductance; the second (*ii*) by evaluating if the relaxation is immediate or if it takes time to complete. For the latter one, the analysis can be performed by simply providing the network with a squared input signal (see Figure 14). During the high state, the conductance tends to raise quickly, stabilising later in a balance point. This is due to the influence of the static

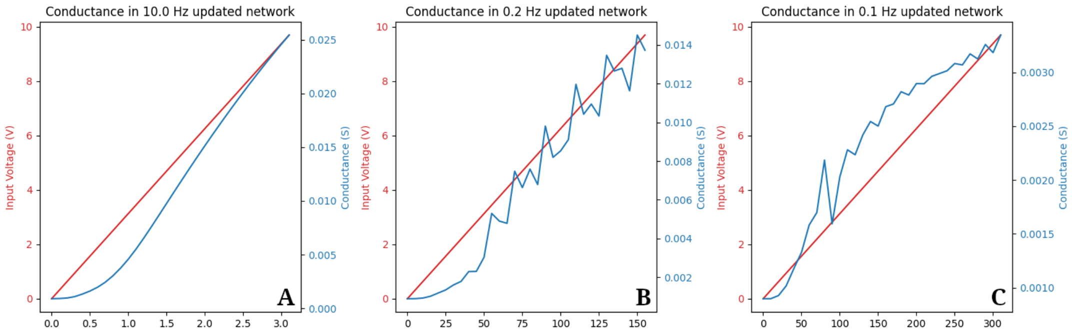

Figure 13: Changes in the input/output path conductance when stimulated at different frequencies. The input signal is a voltage value from 0 to 10V. As visible, for high frequencies we have a stable behaviour (A), while for lower ones it starts to be irregular (B and C). Note that the reached conductance in the first case (A) is almost ten times higher than the last one (C).

load[21]. When the input signal decreases, the network completely relax toward its relaxed state. This ‘depression’ process happens in the order of the seconds.

For the separation property (*i*), the analysis is more complex. Indeed, it requires verifying that for different input streams, the network shows different internal trajectories. Hence, a computational evaluation would require the analysis of an infinite set of input signals. This is obviously not possible. Additionally, to the best of the author’s knowledge, a mathematical approach have still to be proposed for this type of systems. As a consequence, only a basic analysis will be performed. The goal is to verify this property for two sample signals. The approach is to stimulate the network with an input composed of two half-sins (i.e., the sensing peaks), separated by a zero-voltage input. To differentiate the stimulus, the distance between the peaks will be gradually increased. This will determine the test of multiple signals with a similar pattern. The choice of limiting the analysis to the increase of the peaks distance is due to the hypothesis that similar inputs would be harder to distinguish.

[21] If the network has an initial resistance higher than the motor, it will be highly stimulated. In case this stimulation makes the conductance higher than the load, the network will start to relax again. This process will eventually converge to a stable point

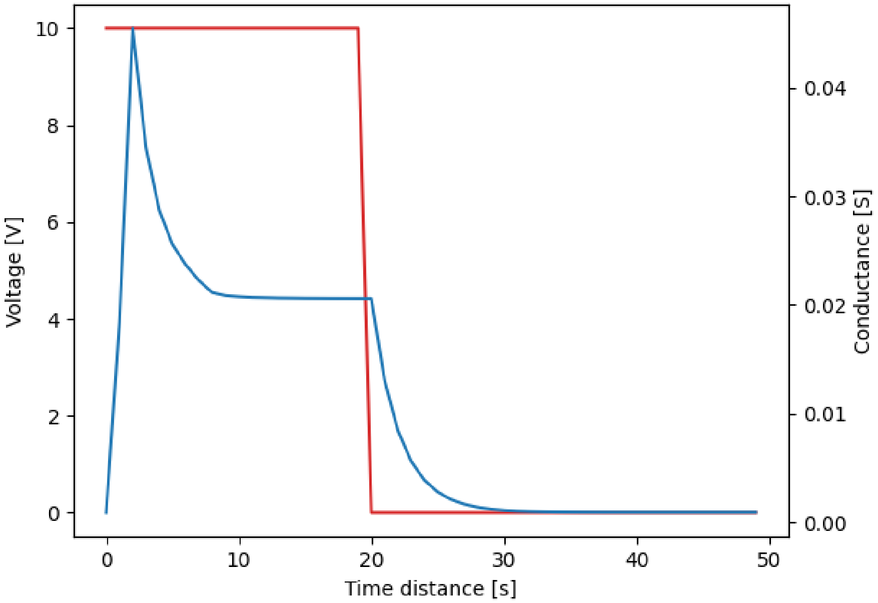

Figure 14: Conductance behaviour when a stimulus is applied and then removed. After the remotion, the conductance decreases to its initial state in some seconds. The strange behaviour visible in the time-window $[0, 20]s$ is due to the low frequency update (i.e., 1Hz) and to the weight of the cascading load (i.e., 100Ohm).

Therefore, a success in this discrimination would suggest a general ability of the system in differentiating the signals. The initial results show an increase in the input/output path conductance, according to the distance between the two peaks (see Figure 15). This behaviour is caused by the influence of an electric potential applied on an already stimulated network. During the initial peak, the system reduces its resistance, and eventually balances the voltage divider with the static output load (see Figure 14). When a second, near-in-time stimulation occurs, a lower electrical potential insists on the network, making its conductance increase less than before. This causes the input/output resistance to be higher than after the first stimulation. The process is contrary with farther peaks. Indeed, in this case the network have time to relax, and thus results more sensible to new stimulus (see Figure 16).

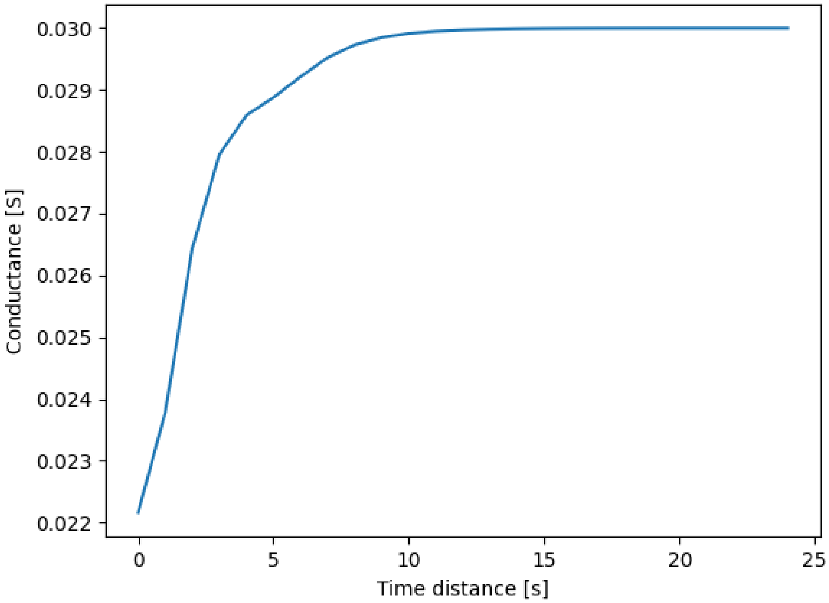

Figure 15: The behaviour of the input/output conductance when the network is stimulated with a signal composed of two peaks (i.e., half-sin waveforms) separated by an increasing time. It is visible that, as the time-distance increases, also the conductance does. The difference is clearly visible until almost 10s, after which it disappears. This means that the evaluated network is able to distinguish signals that have separated peaks distant at most 10s. For farther inputs, the memory almost completely fades, causing an indistinguishability of the signals.

## Critical state

Many studies about the usage of brain-inspired controllers, propose that the higher computational capabilities emerge at the critical state of a system [22]. This concept has been made famous by the conjecture ‘life exists at the edge of chaos’. It is believed that the critical state, defined as the phase transition between order and chaos, posses the ability of balancing adaptability and robustness. Specifically, ordered systems are known to be robust, but less adaptable to changes or to unexpected situations. Chaotic ones are instead incredibly hard to control, with the input signals that are constantly inflated. Following this line of thought, the possibility to induce a critical state in the

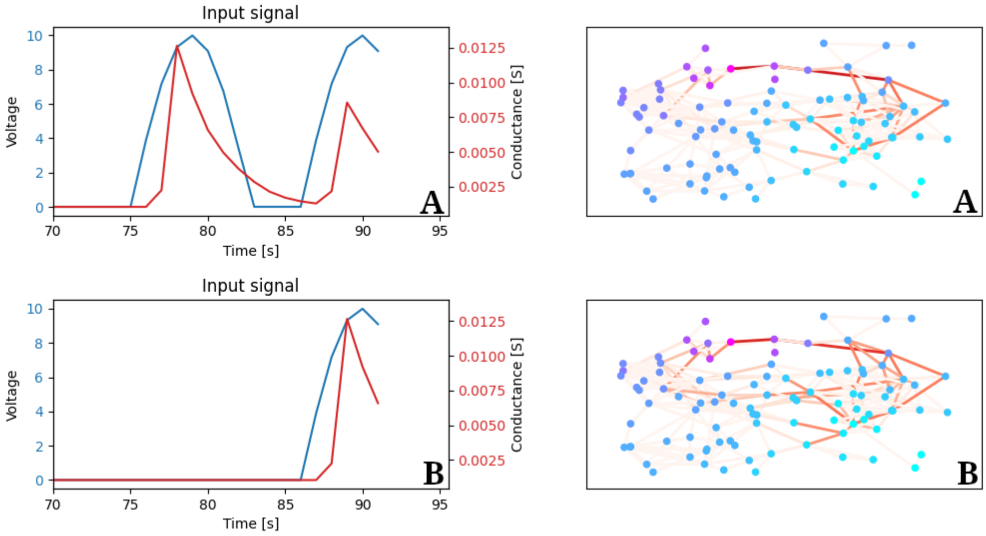

Figure 16: Detail of the separation property test. The top graphs (A) represent the results of a ‘near-peaks’ stimulation (i.e., 3s), the bottom ones (B) a ‘far-peaks’ one. In the first (A), the network does not have time to completely relax, resulting in a lower final stimulation. In the second (B), the network is ‘freshly’ stimulated, and thus it responds strongly. Indeed, after long periods of time, the influence of the first stimulation fades, making the network again sensible to new stimulations.

nanowire networks has been assessed.

The first consideration relates to the creation of those networks. As said in Chapter 1, a critical point of density is $\sim 5$, when most of the nanowires start to be connected together. This relates the number of wires with the topology of the network, according to the percolation theory. Nevertheless, this type of criticality is only related to the emergence of a large connected component, and not to the working state of the system. Considering the previous results about the influence of the stimulation frequency (see Section ‘Update frequency’), the possibility of inducing a critical state by varying the update rate of the system has been assessed. An alternative is the evaluation of the influence of the input voltage intensity. Nevertheless, this last possibility has not been deeply assessed.

As already explained (see Section ‘Update frequency’), when the network is stimulated with low frequencies we assist at an irregular behaviour of the output signal (see Figure 11). This phenomenon is due to the non-linearity of the stimulation and relaxation curves, that may cause a drift towards one of the two states (i.e., excited and relaxed). The irregular behaviour is thus not only present in the output signal, but also in the network conductance itself. The hypothesis is that the update frequency may determine a critical state of the system, where the resistance and output oscillate in irregular ways, without stabilising. This slightly resembles the concept behind the 'logistic map' [27], and from that it is inspired. Therefore, the test consisted in the search of a similar pattern in the network conductance. Running a simulation, the system was stimulated with a 10 voltage input at different frequencies. The behaviour of the average conductance of the network was thus evaluated. The results show that the signal stabilises to a fixed point attractor for high frequencies, and to a two point attractor for low ones. Nevertheless, the most interesting behaviour is recognised in a specific range of update rates, that does not present any periodicity[22] (see Figure 17). The discovery of a non-periodic signal confirmed some sort of chaos in the network dynamics, depending on the frequency. The analysis continued then with the evaluation of a wider 'population' of update rates, showing a chaotic-like behaviour near to specific ranges of values (see Figure 18).

All the previously discussed behaviours depend on the time that the network needs to relax, and on the non-linearity of this process. Specifically, the relaxation at high frequencies almost completely cope with the stimulation, allowing the network conductance to stabilise to a fixed value. At low update rates instead, the relaxation strongly influences the behaviour, creating a multi-points attractor. The working principles of this property suggest that the voltage intensity of the input signal may take part in the definition of the critical state, modulating the stimulus impact. Nevertheless, the preliminary analysis show only marginal changes in the range of the chaotic behaviour. Indeed, the variation of the stimulation intensity affects the average conductance, but not

[22] This consideration excludes the 'stability-reaching' period, discarding the first iterations results

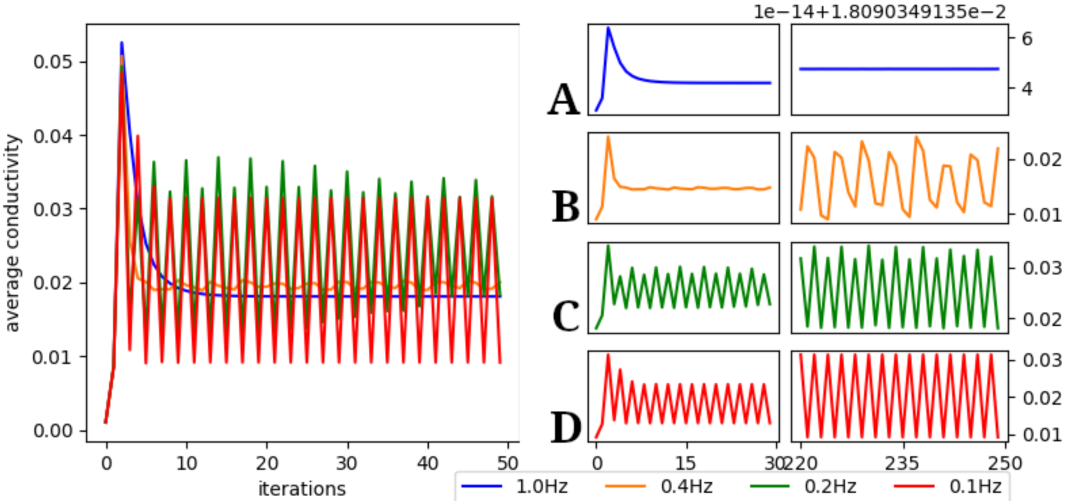

Figure 17: Average conductance behaviour in a network stimulated with a 10V input signal at different frequencies. Stable single, and two points attractors are visible for the top (1Hz; A) and bottom (0.1Hz; D) signals. The middle frequencies (0.4Hz and 0.2Hz; B and C) generated instead a non-periodic pattern. This is visible in the zoom on the right, clearly for the 0.4Hz signal (B). Although not clearly visible, also the 0.2Hz (C) behaviour is non-periodic, with some sporadic expansions in its range of conductance values.

the frequency-attractor relation. In other words, the range of frequencies that causes a chaotic behaviour in the network remains almost the same.

Finally, the behaviour of the conductance not only depends on the frequency and stimulation intensity, but also on the topology of the network. Different devices can indeed show structures that are more suitable to be stimulated. This complicates the search of a critical value. Therefore, the analysis should be specific for each instance, and evaluated in a complete setting. Indeed, the connection of external components changes the behaviour of the network, possibly moving its critical point.

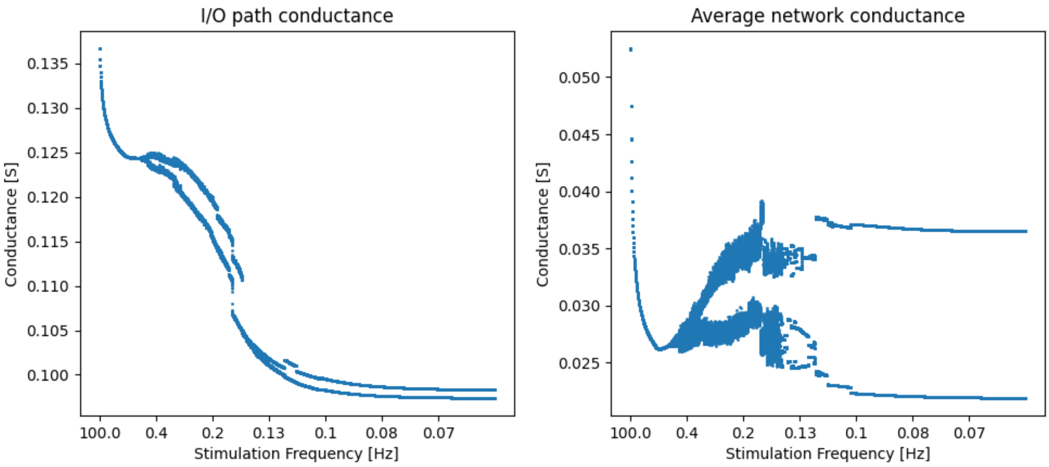

Figure 18: Behaviour of the conductance in a network stimulated with a 10V signal at different frequencies. For each value of the *x* axis, the values assumed by the conductance during its oscillation are plotted. The left graph shows the changes in the input/output (i.e., sensor/actuator) path. The right image plots the changes in the average conductance of the whole network. Note that this behaviour belongs to a specific device instance, and therefore it is not a valid generalization. Nevertheless, qualitative analysis seem to show the presence of the chaotic area in the same range of frequencies also for other networks.

# 4 Robotic architecture

A robot is an artificial system composed of many components. Its architectural design must define their role and the interaction between them. During the development of the project, this schema emerged not only from a technical refinement, but also from a parallel with the biological world. This approach helped in the design of a functional and optimised architecture, suggesting how the data should flow and how each component should behave. Additionally, this parallel allows for a more coherent administration of the project, matching the founding ideas of adaptation, phenotypical plasticity, neuromorphic computation, etc.

Because of this duality of perspective, this chapter will be dedicated to the definition of both the representations. The starting point will be the technical model, followed by a description of its biological inspiration.

## A technical perspective

The classical way to describe a system consists in the representation of all its components and their interactions. For robots, this definition is usually more complex, in that it considers elements of two different worlds: the physical and the logical/software one. This consequently often causes a detriment in the quality of the represented model[23]. From this point of view, the use of nanowire networks strongly improve the architecture, in that they work in the same physical world as the robot does. If we ignore that the system run in simulation, the only programmatic part relates to its adaptation. This results in a model composed mostly of hardware components. To provide a schematic description, we may imagine this robotic system as a composition of three parts: transducers, connections and controller.

One of the main building blocks are the transducers, that allow the robot to interact with the world and being embodied in it. This means that, taken the environment as a different system, the actions on one can perturb the state of

[23] The signals and data have to be converted between the two worlds, often requiring additional components to act as interfaces or converters

the other [37]. This is possible in that the transducers are converters of energy-forms, allowing for signals belonging to one of the systems to be perceived by the other. Practically, those converters take the shape of sensors and actuators. The firsts convert the outer signal in a way intelligible by the robot (see Figure 20); the seconds convert the will-of-action to produce a change in the world (see Figure 21). For the goal of this research, only three types of transducers are actually needed: the proximity and ground sensors[24]; the motors. All those are found in the EPuck robot (see Figure 19).

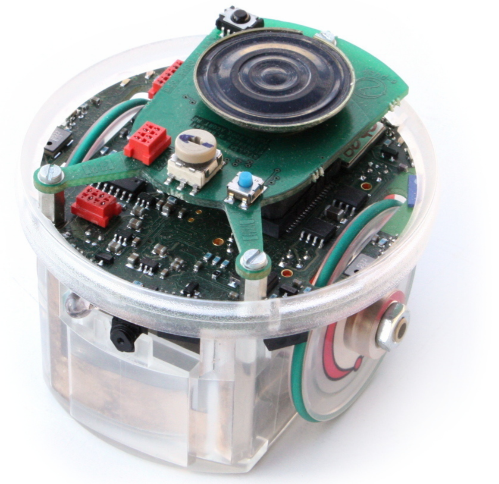

Figure 19: Image of the EPuck, the robot used in the experiments. The image comes from [25].

The other important component for a robot is the control center. This usually consists in a programmable controller, able to elaborate the perceived signals and to produce perturbations in the world through the use of the actuators. The logic deciding how to respond to sensing or to proactively perform actions is thus contained in the robot brain. Typically, it is static and/or immutable. The goal of this project is to create a system that can adapt in order to complete tasks or to learn behaviours, without the need of a clearly defined software

[24] This classification is only for the sake of simplicity. Both the proximity and ground sensors are indeed composed of an emitter (i.e., actuator) and a receiver (i.e., sensor) of infrared signals. The classification as sensors is due to their final goal

solution. The attempt is to achieve it through the use of a controller consisting of a nanowire-network and an adaptive mechanism. As for the general schema, this will take the inputs from the sensors and generate an according output to control the actuators. The network shall autonomously manage any kind of memory required for the task. The logic controlling the robot will be then completely embedded by the controller and modulated through its connections with the transducers (see Figure 20 and Figure 21).

The final components of the robot are the connections. Those create a link between the transducers and the nodes of the network. In the simplest idea, they just allow the data to flow in a bidirectional way. In the current model, however, it was decided to delegate to them also a processing/modulation of the transferred signals (see Figure 20 and Figure 21). This choice comes from the possibility to embed in them some sorts of amplifiers or attenuators, avoiding the need to model an additional entity. This results in the usage of those connections also to adapt input and output ranges to the ones expected by the network.

## A biological perspective

As said at the beginning of this chapter, the architecture of the robot may be seen as a parallel to the biological world. The controller, sensors, actuators, and in general all the components usually have some sort of correspondent in a living body. The focus should then be to find a viable model that helps in the study, design and representation of this robotic architecture. Nevertheless, it is important to work on a reduced schema, in order to keep the parallel simple and thus more effective. Indeed, the number of components in an EPuck, or in general in any kind of small wheeled robot, cannot compare with the complexity of most of the biological systems. The alternatives are then to select basic living beings or to simplify the model of more complex ones. For this research, the second possibility has been chosen. The selected model is a simplified abstraction of the human body, cleaned by all the exceeding organs that does not have a correspondent in the robot. This choice obviously reduces the contact points between the architecture and the biological system, however it also allows studying and researching in a better known area[25]. As a final clarification, a 1:1 mapping of the function of each organ is also not possible. This limit is due to the necessity of balancing between a technical implementation and its biological inspiration. The goal is indeed to take the best from the two perspectives, and not to accurately mimic the human body functioning. Constraints and premises given, the next step is to determine the mapping between the entities of the two models.

The main block of the robotic system is obviously the nanowire network. It is the central component in driving the robot completion of tasks. This role obviously links it with an area of the human neural system. Specifically, its memory capabilities associate it with the area of the hippocampus, while its motion-related computation with the motor cortex. To simplify the logic, this distinction in sub-areas has been ignored, mapping the nanowire network with the generic idea of *cortex* (see Figure 20 and Figure 21).

The robot acquires the environmental information through its sensors. This approach is the same that humans use to perceive the world. This entity is so almost directly mapped into its biological correspondent, with the only distinction related to the number and type of sensors involved (see Figure 20). The first difference is prominent, with the human body exploiting an incredibly higher number of sensors. Although different in scales, the final decision was to address those entities with *sensor* in both the models. The types names are instead maintained due to the difference in the perceived energy-form[26] (e.g., the human body does not possess any ground-sensor, making a possible parallel far-fetched).

The acquired signals have to be re-directed toward the network. This is a central problem in the project. The system aims indeed to exploit the nanowire network, reducing to the bare minimum any pre- or post-processing of the signals. This goal move then the effort from the transformation of the data to

[25] The author of this dissertation, and probably most of the readers, have more knowledge about the human body compared with simpler, but often unknown, living beings

[26] In reality, many sensors in the EPuck can find a counterpart in the human body. Nevertheless, this requires a denaturalisation of the goal, cancelling differences between, for example, IR and ground sensors and making them relate with *eyes*

the exploitation of the network topology. This is done by finding a convenient connection between transducers and neurons. Searching for a similar link in the human body, we find a group of organs and fibers ascending from the body sensors up to the cortex. Between them, the *thalamus* is of prominent importance. Its role consists in forwarding all the incoming signals and redirecting feedbacks to the correct location [34]. This is what the connections in the robotic system actually do (see Figure 20). Additionally, the thalamus control the flow of information to the cortex and acts as a filter to noise [2, 11]. This allows the possible needs of simple pre-processing to be performed without breaking the logic of the biological model.

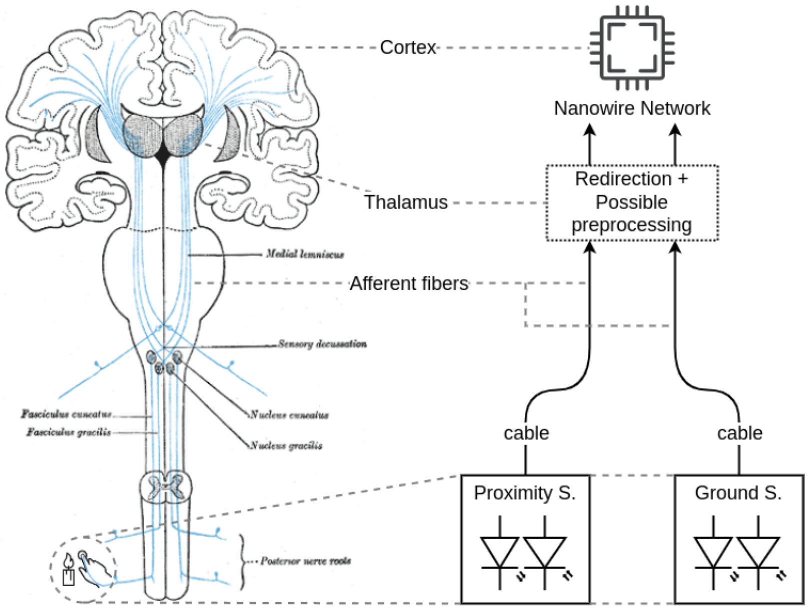

Figure 20: Correlations between the biological/human and electronic model for the representation of the robotic architecture. The scheme represents only the afferent fibers (i.e., ascending-sensory-pathways), and not the efferent ones (i.e., descending-motor-pathways). The sensory information is forwarded from the sensors up to the thalamus, and then to the cortex, via the fibers. Although in a biological human model many others organs are involved in the ascent, the image focuses on the common points with the robotic/technical model. The left side of the image comes from [8].

When the information reaches the cortex, it passes through the network, undergoing a transformation. The result is a control signal obtained by the modification, mixing and extrapolation of knowledge from the input. In the biological model, the obtained information would be forwarded to various part of the nervous system, like the cerebellum and the basal ganglia. The goal of this distribution is to allow an optimisation of the control signal and the production of feedbacks to the cortex [20]. This procedure is simplified in the electronic model, and assumes that this multi-areas elaboration is fully performed by the nanowire network. The final outcome is thus a simpler scheme that does not contemplate the possibility of explicit feedback arcs or external mechanisms to improve the result (see Figure 21). This last characteristic is also an extreme, to the point that minimum or null post-processing is applied to the signal. The idea is indeed that the nanowire network contains all the computational power needed to obtain a good result. The goal is thus just to find a valid configuration of nodes that allows the task to be accomplished. This translates to the possibility for the descending-motor-pathway to be represented as a simple link between motor-control-nodes and the robot actuators. Nevertheless, for the sake of architectural design, we need an abstraction that may represent those descending pathways. In the project, it has been identified with the part of the medulla called *pyramid*. This section of the brainstem is indeed an aggregation of descending fibers [20]. During their descent, most of those will eventually cross, getting directed to the area of the body that they control. This concept actually resembles what is done within the robot, with electrical connections running from the network nodes to the left or right motor. An additional advantage in choosing this element for the representation is that it is located at the lowermost part of the brain. This means that most of the fibers of the descending-motor-pathways have already been grouped together when they form the pyramids [16]. From a computational point of view, we may imagine that most of the computations have already been performed. This again perfectly matches with our definition of the robot architecture. The computation is indeed performed only by the nanowire network, and only few adjustments are done in cascade of it.

We already talked about the motors, but we did not precisely clarify how they are associated with the biological model. One of their goals is obviously to allow the robot to move. However, this is only a partial description. More generally, the motors help the robot to act in the environment, possibly modifying it. This use is comparable to what the muscles do in the human body (see Figure 21). Because of this similarity, we decide to adopt the *muscle* as the biological representative of the robot motors.

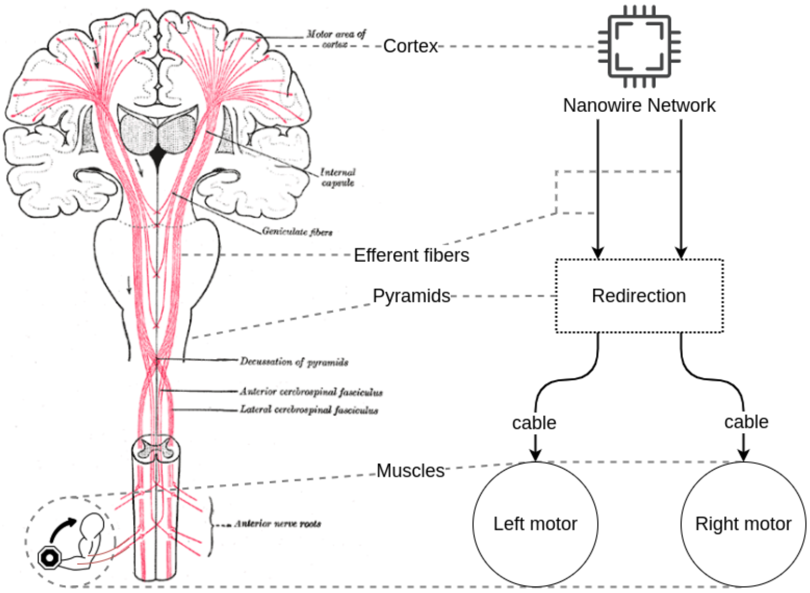

Figure 21: Correlations between the biological/human and electronic model for the representation of the robotic architecture. The scheme represents only the efferent fibers (i.e., descending-motor-pathways), and not the afferent ones (i.e., ascending-sensory-pathways). The control commands that originate in the cortex are down-streamed towards the muscles. We can also identify the bottom part of the brain, in which the fibers group and form the pyramids. The left side of the image comes from [7].

## A final outcome

Given the technical and biological perspective of the robot architecture, it is now time to merge the views to obtain the final schema. This is composed by: the cortex; the body (i.e., the group of sensors and muscles); the pyramid; the thalamus (see Figure 22). Each of them respectively matching with: the nanowire network; the EPuck; the node-to-actuator connections; the sensor-to-node connections. Those represent all the components defined in the views, and concur in providing all the functionalities of the robot.

Since the experiments run in a simulated environment, those entities had to be defined programmatically. This is done with a functional approach. The classes are thus seen only as data storages, without included methods.

The EPuck (i.e., the robot body) is composed of motors and sensors. This does not include the connections in that those are seen as re-connectable entities, that do not well match with the immutability of the EPuck. The pathways are thus seen as a different layer, that may be modified by choice. Indeed, between the four building blocks of the model, those two (i.e., the pyramid and the thalamus) are the only ones that are supposed to be externally/voluntarily modified in order to change the behaviour of the robot. The cortex represents indeed an electronic circuit that is not supposed to be, possibly, externally modified. The only possibility is thus to entirely substitute the control system.

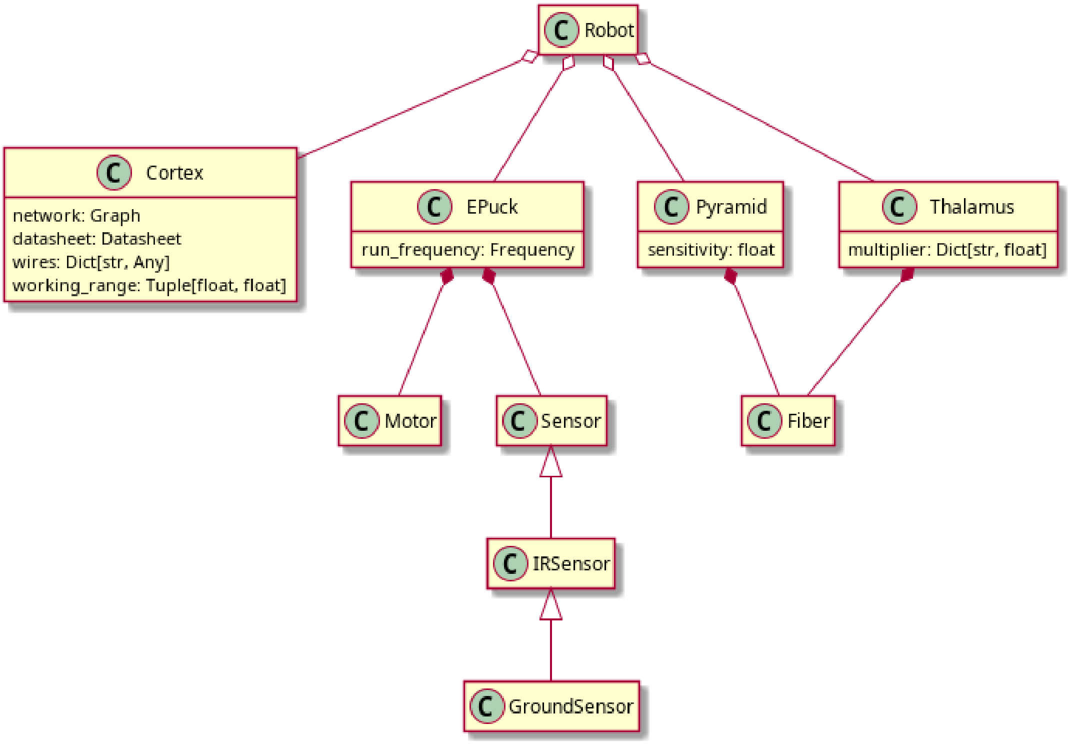

Figure 22: The UML schema of the robot architecture. It represents the composition of its building blocks. This follows a functional paradigm and thus its classes do not contain methods. The entities represent hardware components in a simulated environment. The inheritance relation between the IRSensor and the GroundSensor is due to the technical similarity between the two. Although they measure different information, the ground sensor is just an IR sensor that is pointed towards the ground.

# 5 Experimental setting

In Chapter 4 the structure of the robot has been discussed. Nevertheless, the control architecture only represents what we want to optimise. The process of optimisation and assessment has indeed its own structure and approach, that has to be discussed. This is inspired from the adaptive computation framework and determines another parallel between the natural world and the robotic system here analysed. According to that, the approach should assume an online adaptation, allowing phenotypical plasticity (see Chapter 2).

## Automatic design

The automatic design is a methodology for the modelling that exploit some optimisation techniques [14]. The approach used in this dissertation is called *adaptive*, and is inspired on the well-known 'evolutionary computation' framework. Indeed, the main difference relates to the target of the optimisation: an individual versus a population. Therefore, the discussion initially considers the idea of evolutionary computation, as it is a similar but better known methodology. The differences with the adaptive approach are subsequently illustrated.

The evolutionary computation is a methodology for the optimisation of solutions belonging to a high set of possible configurations. It overcomes the problem of finding a mathematical function that can be used to evaluate the performance by looking at the design. Indeed, that may not exist at all [32]. The main inspiration of this approach is the evolution of natural beings. Its strength point is the possibility to improve the solutions that we already have, without having to explore all the possible ones. This allows to greatly reduce the computational complexity, although possibly precluding to find the optimal configurations. The process follows a simplification of the natural reproduction approach, identifying the steps of (*i*) crossover and (*ii*) mutation as a way to improve the *population*. At each *epoch*, a *generation* of *individuals* is 'executed'. This represents the life of each entity, that gains feedback on its performance and is categorised in a range between successful and unsuccessful.

Just like in the living world, outstanding individuals have more chances to survive and to reproduce, hopefully passing their good-scoring heritage to their descendants. This process includes the crossover (*i*), where two good scoring individuals mix their genetical material to form an *offspring*. During the reproduction however, some mutations (*ii*) may take place, changing some genes. This process is what helps the solutions to escape local optima.

In the developed system, the evolution process is simplified and substituted by an adaptive approach[27]. The optimisation does not operate on generations, but on single individuals. The crossover (*i*) step cannot thus be executed, leaving the duty of the adaptation to the mutation (*ii*) alone. This is due to the computational complexity in evaluating more individuals in a generation. The computational systems available for the tests are indeed not able to run more instances of the robot in acceptable time. This problem limits the ability of the system to evolve and thus change the approach from evolution to adaptation. The new framework runs a robot in an epoch to evaluate its performance, expressed as fitness. The best solution is then modified and tested in a subsequent execution. Nevertheless, if no configuration reaches the minimum threshold, the individual is created again from scratch (see Figure 23). This last feature, if well configured, allows the system to search a good enough solution from which start the adaptation, avoiding optimising very bad ones. This is possible due to the reduced search space of the parameters. More complex tasks would instead be probably penalised by this approach. It is important to underline that the optimisation process does not create nor adapt the nanowire network and the robot body. The only modified entities are the connections between the network nodes and the sensors. The mutation can take place as a simple rewiring or also include a weighting of the connection links. In any case, this follows some rules in order to avoid naive systems to be created, and to allow the adaptation to proceed. The first one, requires at least two wires separating inputs and outputs, thus avoiding a complete remotion of the network dynamic[28]. The second, is the use of a Gaussian mutation for the variation of the signal multiplier.

[27] The robot is always the same, and only its behaviour adapt

[28] The dynamic is given by the strengthening of the junctions between the wires

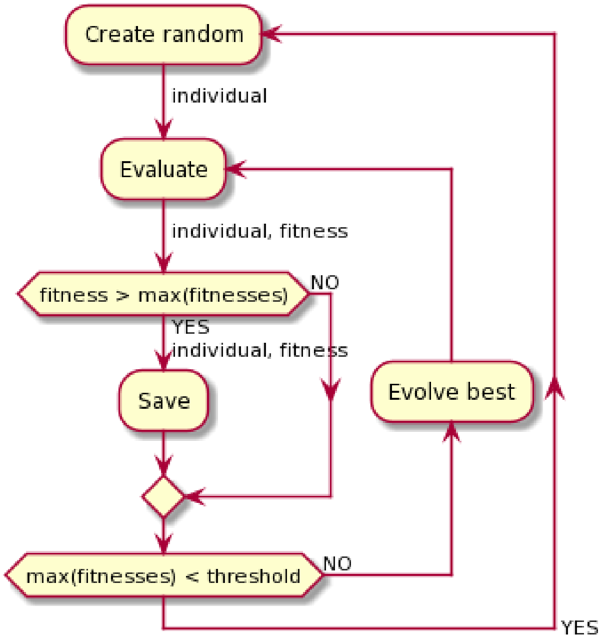

Figure 23: Adaptive cycle of a robot configuration. Note that the processes of creation and adaptation of the solution modify only the connections between the robot and the network.

## Statistical evaluation

The previously defined approach represents the adaptation of a specific robot-controller system. Nevertheless, the tests required by this research have to generate robust data and to evaluate different configuration parameters. The process requires then multiple adaptation-lines to be optimised. This means that the architecture has to adapt different starting configurations, such that their results can be statistically evaluated. In the code, the adaptation-line takes the name of *Simulation* and consists in the result of a sequence of epochs. In those, different configurations are tested for a specific period of time, in order to calculate their performance[29] (see Figure 24). The test architecture consists then in multiple Simulations, each of which will generate an optimised solution starting from the initial configuration. Those adaptation-lines can differ in

[29] The calculation method is task-specific and is not discussed here. See Chapters 6, 7 and 8 for more information

multiple parameters, such as: (*i*) the task; (*ii*) the motor load; (*iii*) the creation density of the nanowire network; (*iv*) the random seed. Specifically, (*iv*) allows generating reproducible and robust results, in that the probability that they are due to a lucky configuration decrease. Those different instances, created from the same starting parameters, are called *Replicas*.

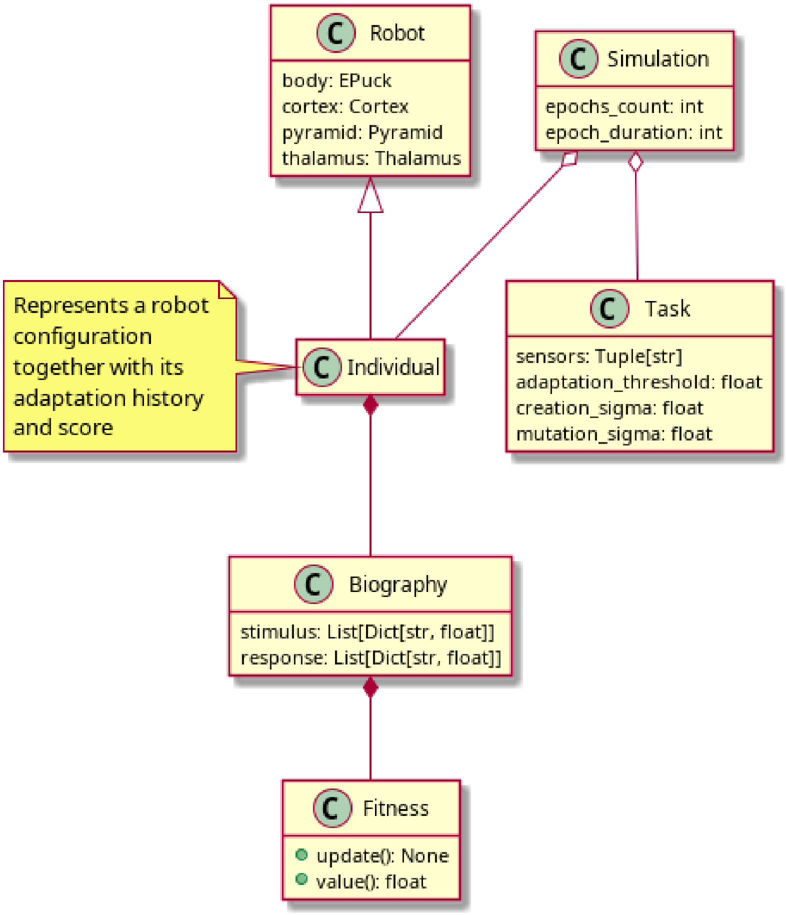

Figure 24: Architecture of the adaptive system. An Individual directly represents a Robot, considered as the union of brain, thalamus, pyramid and body. However, it also includes a reference to the story and fitness obtained during its execution. The Simulation represents instead the adaptation-line of an individual in time, subject to a defined task. Except for 'Fitness', all classes follow the functional programming style and thus do not contain methods.

## Experiments parameters

Every experiment runs in a specific environmental niche. This is a static arena in which the robot performance is evaluated. The environments differ according to the goal task that we are optimising[30]. Different niches may provide different sensorial information to the robot, that may consequently need different types of sensors.

Overall, the experiments use some default parameters. Those may change depending on the test, but most of them remain constant. First, the update frequency of the simulation (i.e., the robot and the network) is 10Hz. This is a sufficient update rate to allow the robot to move smoothly, while saving computational resources from the nanowire network simulation. This parameter never changes in any experiment, and also represents the time-duration of an epoch step[31]. The number of epochs in each adaptation-line is normally 30. Experimentally, it resulted in a good amount of mutations for the configurations to succeed, at least for easy tasks. At the same time, it allows for relatively fast executions. This value changes only for the T-maze experiments (see Chapter 8). The epoch duration in steps is not discussed, in that it differs in each task. The replicas of the configurations are set to 30. Their goal is indeed to avoid ‘lucky’ or ‘cursed’ instances to heavily influence the overall results. Therefore, a limited number of repetitions should be enough for this goal, guaranteeing statistical significance. Each replica is obviously created with a unique seed.

## Robot and network parameters

An important point in the run of the experiments is related to the setting of some robotic parameters. The search space of the possible configurations is indeed theoretically unlimited, and thus requires some preliminary choices to be defined.

The first point relates to the need to adapt the input and output signals

[30] Each environment will be discussed in detail in the tasks specific sections of this research. See Chapters 6, 7 and 8 for more information

[31] The frequency of the simulation, together with the duration in steps, determines the time-duration of an epoch

intensity to the range accepted by the network. This indeed works with values in [0, 10]V, that may not be compatible with the ones provided by the sensors or the ones needed to control the motors. In the simulation, this parametrisation is different from the one that would be used in a physical robot due to the simulator APIs, that provide a higher level approach. An example is the range of motor signals that the output of the network will be converted to. This is set to [6.28, -6.28] due to the working mode of the simulator, that directly uses the rotation speed[32]. The use of a decreasing range is due to the choice of using proximity measures. Indeed, their generated signal belongs to an increasing positive range, with a low reading when no obstacle is visible. Due to the network working mode, the output would be then normally zero, rising in intensity when a wall gets near. As a consequence, the robot would normally remain still. An initial solution to this problem may consist in negating the input. That would however cause the network to be normally highly stimulated. The limiting factor resides in the response to stimuli. Indeed, the dynamics of the excitements is faster compared to the relaxations. This causes a normally stimulated system to react slowly compared to a relaxed one. Therefore, we prefer the control network to be normally 'idle', getting excited faster as a response to unexpected events. Another approach consists instead in centring the range of the sensors' signal on the zero, making it start from a negative value. As a plus, this would allow the robot to move also backward, compared to the first approach. Nevertheless, a negative point consists in exceeding the working range of the network (i.e., [0, 10]V). Moreover, the robot would normally move backward. An additional possibility consists in negating the centred signal. Indeed, this approach does not present very negative points. Nevertheless, it results that making the zero signal correspond both in the input and output may cause inefficiencies. Considering a scenario in which both the input and output signals are zero, the robot would stay still forever. This would hardly happen instead if only one of the two signals is null. In that case, we would have that the robot keeps moving (in case the input is zero) or that the network keeps being stimulated (in case the output is zero). This would thus

[32] 6.28rad/s is indeed the rotation speed in a specific direction (i.e., counter- or clockwise)

increase the possibilities for the robot to escape locking situations. Since no satisfiable solution has been found, the decision consists in zero-centring and negating the output control signal to the motor. Indeed, this approach resolves all the previously presented problems.

In Chapter 3, the influence of the motor loads has been discussed. Due to their physical state, those are impossible to be modified through learning, and thus have to be selected at the start of the simulation. Moreover, an indecision in the choice of a 'general good value' emerged, mostly due to the need of balancing the stimulation of the network and the motor speed. Therefore, the decision is to attempt the use of four different orders of magnitude: [1e+03, 1e+04, 1e+05, 1e+06]Ohm.

Additional configurations relate to the use of the nanowire network. The first point to discuss is the utilisation of a single connected component. The network is indeed composed of multiple groups of connected wires, whose number depends on the creation density (see Chapter 1). The use of all of them would add additional checks[33], and thus computational complexity, to the simulation, slowing it down. Additionally, a risk would be the reduction of the robot to a Braitenberg-like vehicle [5], connecting single inputs to outputs. If the plastic property of the network would still be evaluable, that would not be true for its mixing capabilities. The decision was thus to use only the largest connected component.

Another aspect concerns the selection of the network creation density to use during the experiments. An initial choice was to use the value at which the percolation of the network begins: $\sim 5$ (see Chapter 1). Nevertheless, to allow a broader study, three densities were chosen to be tested for each task: [5.0, 7.5, 10.0].

[33] The Modified Nodal Analysis works indeed only with a connected circuit. The presence of multiple components would require the analysis of each of them

# 6 Experiment: collision avoidance

A practical evaluation of the nanowire network as a robotic controller requires the test of its performance in a given task. This helps to understand the strength and the weak points of its usage to achieve a specific goal. Given the novelty of this control system, it is needed to start to evaluate its behaviour in a well known environment and with a relatively easy task. Therefore, the best choice is trying to achieve a collision avoidance behaviour.

## Task and arena

The collision avoidance task evaluates the ability of the robot to avoid crashes against obstacles and walls. This usually takes place in an arena full of items, that have to be avoided. The task implicitly requires the robot to move. This can follow unspecified approaches, like random walk, exploration, etc.

Being the first experiment, it is aimed at evaluating the robot capability to complete basic tasks. Therefore, the arena is a simple, squared maze with a central block (see Figure 25). The initial setup does not contain other obstacles other than the path, in that the goal is only to evaluate the robot capability in turning when it faces a wall. The only irregularity is in the shape of the central object that defines the path. The use of a rectangle allows indeed to form a strict passage on the left and right sides. This adds a basic complexity to the task, not allowing the robot to simply going in a wide circle. Finally, the environment is completely static, with only the robot moving in it.

Due to their simplicity, this kind of basic arenas are usually used to assess primary capabilities of novel controllers or strategies [4, 13]. Indeed, running unknown systems in extremely complex mazes does not allow to properly understand their behaviour.

## Objective function

The goal of the experiment is to optimise the robot in order to obtain a configuration that can successfully complete the task. Nevertheless, the evaluation of this ability must be rigorous. This is achieved through the definition of

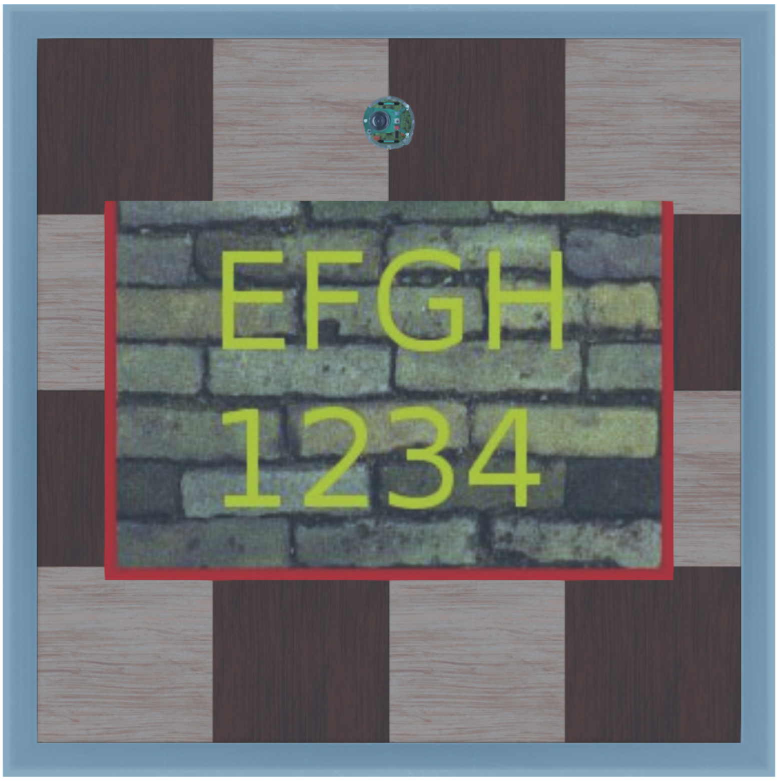

Figure 25: Top view of the arena used for the collision avoidance task. The big object with the writings in the center is the obstacle that the robot have to circumnavigate.

an objective function that can mathematically and deterministically evaluate the robot behaviour[34]. The performances on each step are then summed and normalised in a [0, 100] range:

$$(1-\sqrt{\theta(n)})\cdot(1-|v_l(n)-v_r(n)|)\cdot\frac{v_l(n)+v_r(n)}{2} \tag{2}$$

where:

- $n$ is the actual step;
- $\theta(n) \in [0,1]$ is the max proximity perceived by a sensor at the step $n$. The larger the value, the nearer we are to an obstacle;

[34] The function is defined as a modification of the one described in [4]

- $v_l(n), v_r(n) \in [0, 1]$ respectively are the normalised left and right velocities of the robot wheels at step $n$;

The goal of this objective function is to reward robots that run quickly, in a straight line, turning only when needed. Specifically, $\frac{v_l(n)+v_r(n)}{2}$ favours fast movements in the arena. $1 - \sqrt{\theta(n)}$, rewards behaviours that avoid walls and obstacles. This is made more influent through the use of the square root in the function. $1 - |v_l(n) - v_r(n)|$ prise configurations that prefer a straight movement to a turn.

## Configuration

As said at the beginning of the chapter, the collision avoidance experiment aims to verify the ability of the optimisation system to make the robot complete a simple task. Additionally, it assesses the behaviour of the resulting configurations and the resistance to faults. The run take place in the described arena, and uses the proximity sensors included in the EPuck. Those are needed to perceive the neighbourhood of obstacles and walls[35], and are disposed in an uneven way around the robot, with a higher concentration on its front side. For the collision avoidance task, all of them will be used. This adds complexity, in that the adaptive approach has to balance the influence of each sensor.

Most of the robotic and experimental parameters used in this test are already discussed in Chapter 5. Therefore, we will discuss only the ones that differs from the standard. The epoch duration is 200. The fitness threshold value for the adaptation is 40.0. Finally, the parameter that changes between the runs is the type of the signal used as input. This varies both in the measured property (i.e., proximity or distance) and in the signal pre-processing (i.e., unprocessed, adapted to a custom range, weighted). The assessed types are listed in Table 1.

Note that the 'weighted' approach (see Configuration V in Table 1) was not originally used in this experiment. Indeed, it was developed for the T-maze task (see Chapter 8). Nevertheless, its evaluation may give more information about its validity, and it is thus assessed also there.

[35] The measure exploits the time-of-flight of an infrared signal, sent and received by the sensor after its reflection on a surface

| Configuration | I | II | III | IV | V |
|---|---|---|---|---|---|
| Signal type | $P^1U^3$ | $P^1R^4$ | $D^2U^3$ | $D^1R^4$ | $P^1W^5$ |

1 Proximity measure: neighbourhood of an obstacle;

2 Distance measure: distance of an obstacle (i.e., negation of the proximity measure);

3 Unprocessed measure (or direct): the input signals are used without modifications;

4 Custom Range: the signals are adapted to a custom range equal for each sensor. If it is smaller than the original one, the exceeding values are maximised (if higher) or minimised (if lower);

5 Weighted: the signals are multiplied by a varying factor different for each sensor.

Table 1: Experiment specific configurations.

## Results

The first step for the analysis of the results quality, is the choice of threshold levels for the categorisation between good and bad solutions. The selection has been done discretionary, with the value selected according to observations. The threshold classifies the scores with a ‘cautious’ approach, in that no failing configuration should be able to perform more than it. Instead, some successful configurations may lay below this value. The threshold chosen for the discrimination is 50.

According to the threshold value, the percentage of instances that were successfully optimised during the experiments are presented in Table 2. The data are differentiated according to the experimental Configuration (see Table 1).

| I | II | III | IV | V |
|---|---|---|---|---|
| 24.72 | 76.39 | 35.28 | 77.78 | 62.22 |

Table 2: Percentage of solutions that achieved a value of fitness over 50.0. For each Configuration of Table 1 (i.e., *x* axis), the data are calculated on a total of 360 samples (i.e., 30 replicas * 3 densities * 4 loads).

The results show a general capability of the optimiser to allow the robot to succeed. The range of success varies between 24.72 (see Configuration I in Table 2) and 77.78 (see Configuration IV in Table 2), according to the Configuration. It is already intuitable that the success heavily depends on the chosen parameters and strategies. These results will be further analysed in the following sections, concentrating on single aspects.

### Density influence

As said in Chapter 3, Section ‘Network density’, the density influences the signals distribution and the memory capacity of the system. This makes it a primary important parameter to be evaluated. The need of specific network properties is tailored to the specific task we are optimising, and theoretically not to the chosen adaptive approach. Therefore, the hypothesis is that the density influences the result of the adaptation in the same way, almost regardless of the mutation mechanism[36]. The use of an aggregated index, obtained by the data of the many executions, is thus admissible.

The used measure is the mean improvement given by a density: $\Delta D_\delta$. Taking the successful instances of a Configuration, the value is calculated as the amount obtained by a specific density, divided by the average amount of the run. The result is then averaged between all the tested adaptive approaches. The formula is shown in Equation 3[37].

$$\Delta D_\delta = \frac{1}{\#C} \sum_{c}^{\#C} \frac{3s(D_\delta, c)}{s(D_{5.0}, c) + s(D_{7.48}, c) + s(D_{10.0}, c)} \tag{3}$$

where:

- $\delta \in \{5.0, 7.48, 10.0\}$ is the evaluated density value;
- $\#C$ is the number of tested configurations. In this experiment is 5;
- $c$ is the currently assessed run/configuration (see Table 1);

[36] Obviously, very bad approaches may lead to zeroing the performances. Therefore, we assume that good enough adaptive methodologies are used

[37] Note that the ‘3’ in the fraction in the formula is used to average the successful instances of the density values

- *s* returns the number of successful instances for a density value in the evaluated configuration;

The formula returns 1 if the density does not influence the result. A negative value is measured if the amount of successful instances is lower than the average. A positive one, if it is higher. The calculations use the data collected during the experiments (see Table 3), and are reported in Chapter Appendix. The results for each value of density (i.e., 5.0, 7.48, 10.0) respectively are: 0.9, 1.07, 1.034. The measures are calculated on 600 samples each[38].

| | I | II | III | IV | V |
|---|---|---|---|---|---|
| 5.00 | 21.67 | 69.17 | 30.83 | 73.33 | 55.00 |
| 7.48 | 27.50 | 79.17 | 36.67 | 81.67 | 69.17 |
| 10.00 | 25.00 | 80.83 | 38.33 | 78.33 | 62.50 |

Table 3: Percentage of solutions that achieved a value of fitness over 50.0. For each Configuration of Table 1 (i.e., *x* axis), the data are differentiated according to the network creation density (i.e. *y* axis). Each value is calculated on a total of 120 samples (i.e., 30 replicas * 4 loads).

It is clearly visible that the lower value of density causes a general decrease in the amount of successful instances. The higher value obtains higher results, generating an increment. Finally, the 'middle density' (i.e., 7.48) is the one that performs better. Additionally, a lesser variation related to the 'custom range' input pre-processing is visible, with both the configurations II and IV achieving less divergent results between low and high densities. This is calculated as $\Delta D_{best} - \Delta D_{worst}$. Specifically, they achieved an average variation of 0.13 versus a value of 0.23 obtained by the others. Therefore, we confirm that the adaptive approaches do not change the qualitative influence of the density (i.e., if it is better or worse than the average), but they can shrink or expand the range of the variation (i.e., *how much* better or worse the result is).

[38] 5 configurations * 30 replicas * 4 loads = 600 samples

### Load influence

The other parameter to evaluate is the load. It influences the power on the motor and the stimulation of the system. The assessment of possible changes in the performances related to a variation of the load are thus worth to be analysed. Nevertheless, differently from the density, this parameter does not seem to influence the amount of successful configurations. Indeed, the results vary, finding the lower load being the best in some cases (see Configuration III in Table 4), and higher ones outstanding in others (see Configuration IV in Table 4). Overall, the results do not seem to follow any pattern.

| | I | II | III | IV | V |
|---|---|---|---|---|---|
| 1e+03 | 24.44 | 76.67 | 60.00 | 71.11 | 54.44 |
| 1e+04 | 23.33 | 78.89 | 41.11 | 77.78 | 73.33 |
| 1e+05 | 28.89 | 73.33 | 26.67 | 84.44 | 57.78 |
| 1e+06 | 22.22 | 76.67 | 13.33 | 77.78 | 63.33 |

Table 4: Percentage of solutions that achieved a value of fitness over 50.0. For each Configuration of Table 1 (i.e., *x* axis), the data are differentiated according to the motor load (i.e. *y* axis). Each value is calculated on a total of 90 samples (i.e., 30 replicas * 3 densities).

Although the motor load does not influence the amount of successful instances, it may still impact the scores' distribution. Nevertheless, this also does not show any interesting pattern (see Chapter 'Appendix'). The values seem indeed to be independent of the load. Therefore, it can be concluded that the motor resistance does not influence the results of the collision avoidance task.

### Adaptation strategy

We already started to discuss the influence of adaptive strategies (see Section 'Density influence'). Nevertheless, their effect has not been evaluated. It is important to understand how much an approach can influence the results. By looking at the data, the difference generated by the use of a pre-processing of the input signal is astonishing. The amount of successful configurations increases by an average of 47.09[39] when a ‘custom range’ is used, and by an average of 32.22[40] for the ‘weighted’ approach. This highlights a capability of these pre-processing approaches to make more or most of the instances converge to good results. Therefore, an improvement compared to the ‘unprocessed’ approach is confirmed.

The previous results shown the ‘custom range’ strategy successfully optimising the higher amount of configurations. Nevertheless, this is not directly a symptom of a higher quality approach. Although the ‘weighted’ method performs worse in this aspect, its scores’ distribution shows that the maximum and minimum reached fitness are higher than the others (see Figure 26). Considering the finer grain optimisation of the weights, the approach may perform better in the long run. Therefore, choice of the adaptive strategy should depend also on more strategic considerations.

## Qualitative analysis

Beside achieving the desired result, of primary importance is also the comprehension of the network behaviour. This can hardly be done statistically, in that it requires a qualitative analysis. Therefore, the chosen approach consists in taking one successful solution and analysing its signal propagation. Due to the length of the process, this is performed only for one instance and configuration. The chosen one uses a custom range adaptation and a proximity measure. It has been selected due to the results obtained in Section ‘Results’, that suggest the use of a pre-processed input signal for a successful and reliable optimisation. The plots of a similar analysis, concerning the use of a distance measure, are visible in Chapter ‘Appendix’. Due to the differences in approach, most of the results are coherent but opposite, compared to the ones discussed in this section.

[39] Average score when the ’custom range’ is used $(76.39+77.78)/2$ minus the average score when it is not used $(24.72+35.28)/2$

[40] Score when the weights are used 62.22 minus the average score when they are not used $(24.72+35.28)/2$

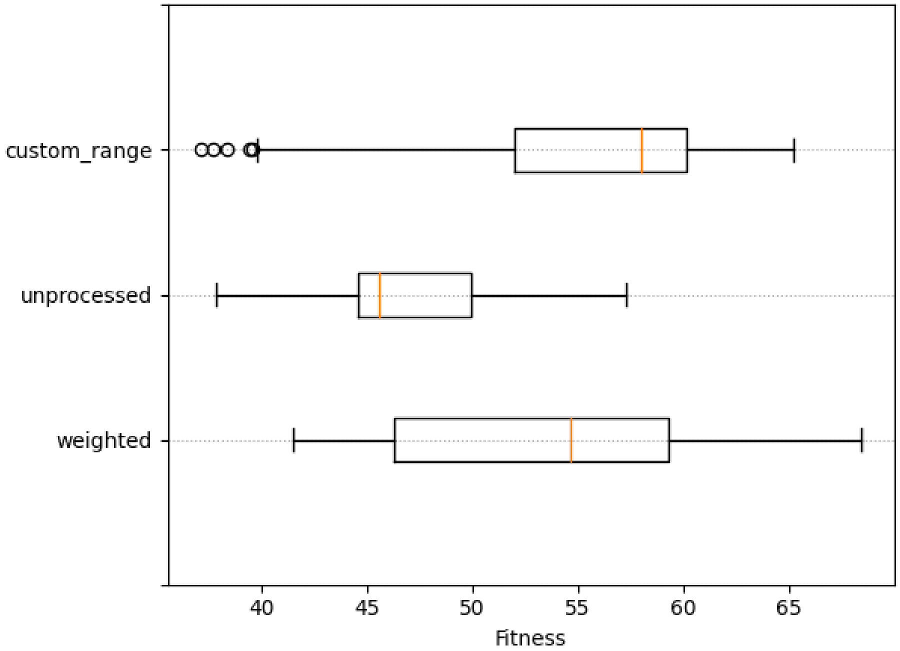

Figure 26: Fitness distribution according to the pre-processing strategy of the input. The data come from the configurations I, II and V of Table 1.

### Overall behaviour

The first point in the analysis of the network behaviour, concerns the input/output signals relation. The goal is to understand how the sensors' perception influences the direction. In other words, how the network transforms signals into what is perceived as the robot behaviour.

The appearance of an obstacle causes an increase in the left/right motors signals ratio (see Figure 27 A, C and D). This determines the right turning of the robot (see Figure 27 B). The initial orientation in the arena is clockwise, meaning that the optimal approach is a starboard manoeuvre (see Figure 25). The change of direction is related to the perception of the sensors. Specifically, the frontmost ones are those that firstly perceive the obstacles (see Figure 27 A). This makes the behaviour mostly dependent on those. Contrarily, a weaker relation would cause the robot response to be slow or even not enough for an efficient avoidance. According to this consideration, the connections of the back sensors are less important for the success of the task.

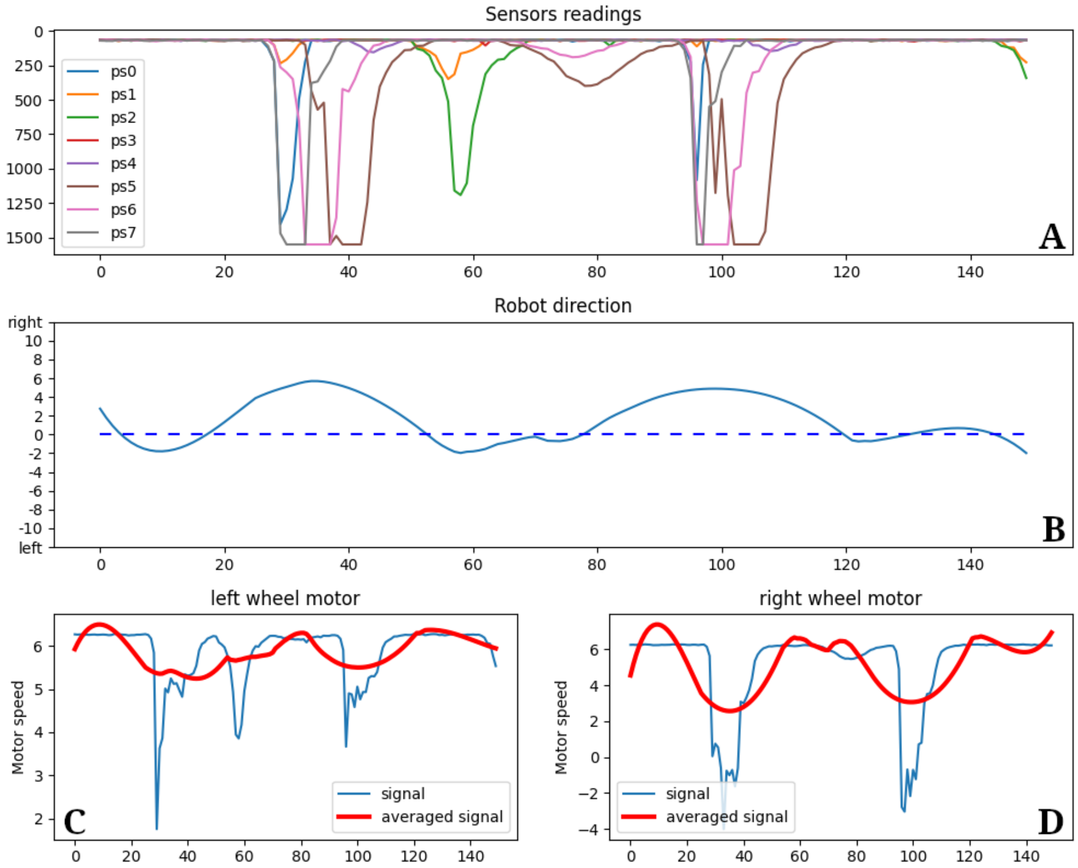

Figure 27: Input/output relation and consequent behaviour. The plot (A) represents the signal perception of the sensors. An increase towards 1550 (bottom) means the approaching of an obstacle. The plot (B) shows the direction of the robot. It is calculated as the left minus right motor speed. When the signal increases, the robot turns right, and vice versa. The plots (C) and (D) respectively show the motors speeds during the iterations. The red line is a smoothing of the effective behaviour to better understand the movement.

### Fault resistance

One underestimated point in the analysis of the quality of a control system is its ability to resist and recover from faults. The first concern the stability of the behaviour after the loss or the fault of a component. The second, the ability to learn again how to behave without the lost appendix. This latter one requires ad-hoc tests, and it is postponed to future works. Nevertheless, the first can be evaluated qualitatively, and it is thus here verified.

One of the expected properties of a good solution is an efficient mixing and transformation of the input signals. Additionally, the exploitation of multiple sensors is desired. This is expected to make the behaviour more stable and robust. The hypothesis is that an ability in the information manipulation may supply the fault of some sensors. The correlation of some input signals supports this conjecture[41].

To verify this property, the first step consists in the analysis of the influence of each sensor in the resulting behaviour. The second, in iteratively zeroing the inputs, in order to spot high changes in the output signal.

To assess the influence of the sensors, it is possible to analyse the correlation between inputs and outputs time series. The measure uses the Pearson coefficient to assess similarities between each signal entering the network and the ones exiting it. An initial consideration is that the results are mostly negative (see Figure 28). The working mode of the system causes indeed the motor signal to rise when the input is low (i.e., no obstacles; see Figure 27)[42]. Therefore, the behaviours of sensors and motors appear as negated. Some signals however do not influence at all the outputs. This is due to their positioning in the robot[43] and connection to the network. Indeed, the input node may cause the sensors to stimulate only one motor. This causes the almost zeroed correlation between some sensing and controlling signals.

The correlation analysis allows individuating sensors that do not influence the motors. Nevertheless, this does not guarantee a relation between the remaining ones. The input signals are indeed mutually correlated due to their positioning, that causes and obstacle to be perceived by many of them. Therefore, the sensors may produce very similar outputs, that may bias the Pearson measure. This invalidates the evaluation of the mediating capabilities of the

[41] Due to the sensors' disposition around the robot, most of them are able to perceive the same obstacles. This makes the input signals correlated

[42] The configuration uses proximity measures. Therefore, the neighbourhood of an obstacle would cause an increase in the signal. Consequently, due to the negation of the output of the network, the motor signal decreases

[43] Sensors placed on the back of the robot are less subject to stimulus caused by obstacles

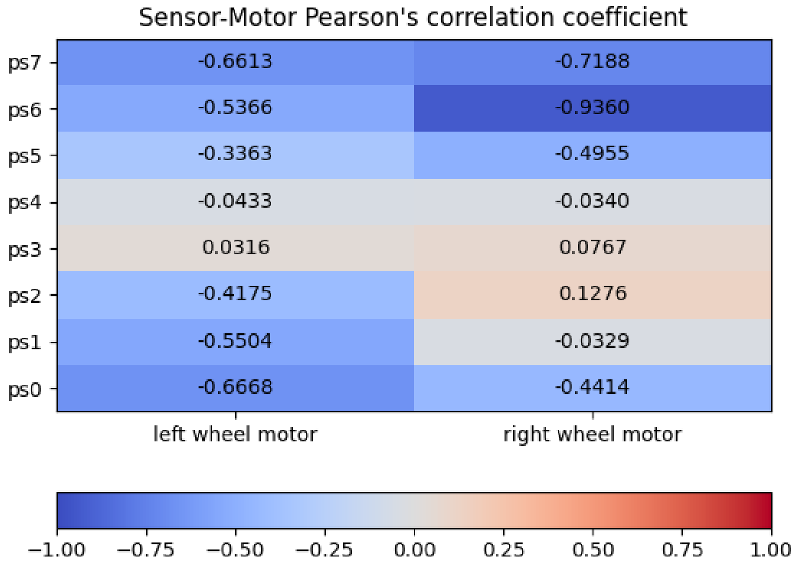

Figure 28: Pearson correlation between sensor and motor signals. The higher the absolute value is, the higher the correlation.

network. To approach this problem, an additional analysis has been carried on. The goal is to understand how influential the failure of a sensor is. The approach is thus opposite, analysing if the absence of a signal severely impacts the behaviour of the robot. This allows to understand if the system is controlled by a just few sensors or if it acts as a mediator between many. For the evaluated configuration, the behaviour seems to be the latter. Indeed, no sensors completely influence the output, meaning that at least a minimal signals blending takes place (see Figure 29). This property translates in a substantial resistance to failures, with the motor control signal remaining mostly stable also after loosing an input. The automatism of this behaviour makes it more interesting, not needing a specific training.

### Topology of the connections

All the previously analysed properties depend on the connections of the transducers to the network. The influence is often approximable to a physical

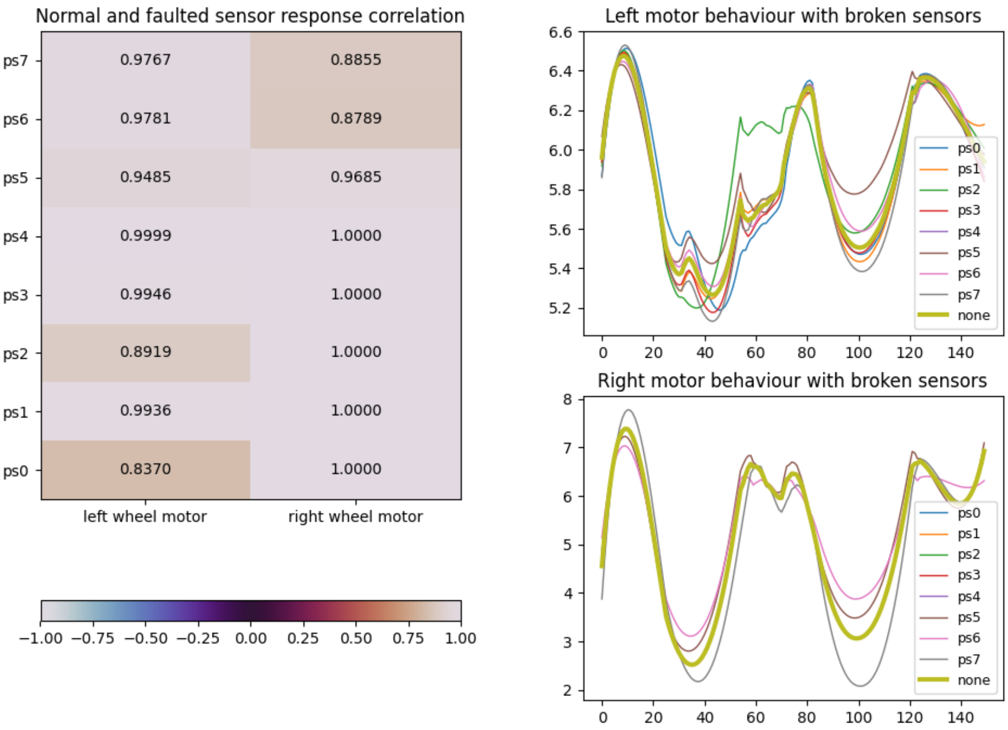

Figure 29: Changes in the motors signals due to simulated fault sensors. The breaks are tested subsequently, and one at the time. The map on the left shows the correlation between the normal signal and the one obtained by breaking the sensor. Values near to 1 means small or no change. Although some sensors are more influential than others, it is visible that no one is overwhelming. This is confirmed also by the right graphs, that compare the behaviour without breaks (i.e., the ‘none’ line) with the one produced by each faulted sensor.

neighbourhood between the input and output nodes[44] (see Figure 30). This is a simplification, but it helps in qualitative and shallow analysis. Specifically, it supports in understanding how the distance and position of the input and output nodes impacts on the influence and in the mixing capabilities of the network, allowing complex behaviours to rise.

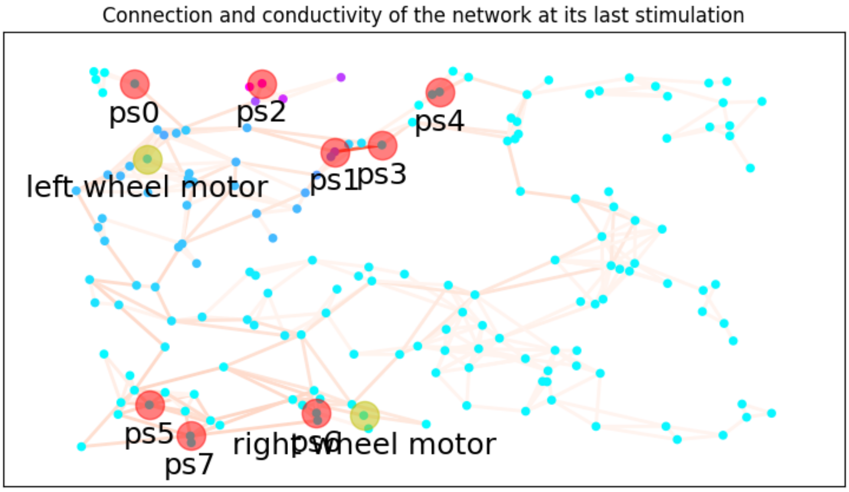
Connection and conductivity of the network at its last stimulation
ps0
ps2
ps4
left wheel motor
ps1
ps3
ps5
ps7
ps6
right wheel motor

Figure 30: Connection of the transducers to the network. The sensors concentrate around the motor that they influence the most. This is likely to change with more complex tasks, that require a complete blending and transformation of the signals. We can see that the right sensors (i.e., ps0/1/2) tend to organise around the left motor. This organization appears also around the right one.

A systematic and more reliable measure, consists in considering the input/output path resistance. In a normal circuit this is static, but in the evaluated network it varies runtime. This possibly biases the sensors' equilibrium, making the behaviour change according to specific circumstances. Due to the relative simplicity of the task, and to the adaptive pressure, this property did not clearly emerge. Although the resistance effectively changes, the behaviour

[44] This is not always true, in that the influence depends on the topology of the network and on the interaction with other inputs. In dense systems this is more likely to be true compared to sparser ones

of the robot remains the same (see Figure 31). A final consideration is that many sensors seems to organise around a motor to influence its behaviour. This confirms the previous consideration that the output is a combination of each sensor signals also from a topological point of view.

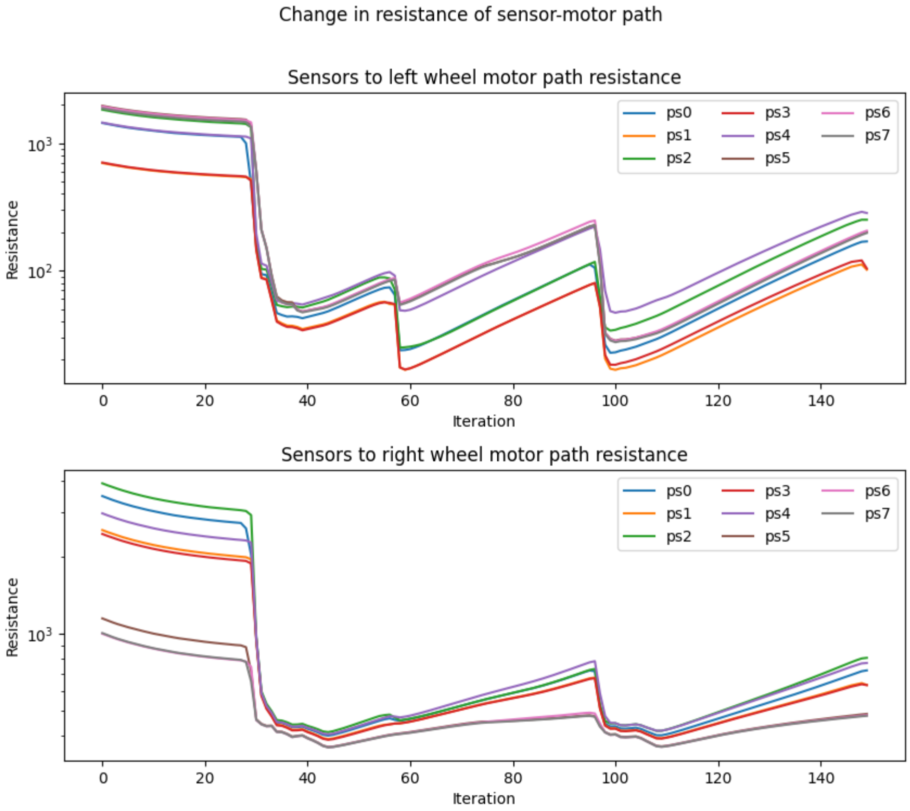

Figure 31: Variation of the resistance during a robot run. As visible, the path-resistance of near nodes is lower than for farther ones. This allows some sensors to have more influence than others (e.g., left ones on the right motor). Nevertheless, no signal presents extreme behaviours. Finally, this graph shows also the stimulation and relaxation of the network. This is especially visible in correspondence of iteration 30 and 100.

## Conclusion

The analysis assessed the ability to successfully complete the task, and the influence of the configuration parameters on the result. The network creation density shown an influence on the average result of the optimisation, with higher values performing better. The motor load resulted instead not influent on the final performance, for most of the runs. Considering the adaptive strategy, the pre-processing of the input signal shown to be important. The use of a ‘custom range’ shown the best overall results, closely followed by the ‘weighted’ approach.

The experiment also evaluated a successful solution, in order to clearly understand what the optimisation produces. The analysis shown a well organised disposition of the connections, for the evaluated task. This explained the ‘information flow’ in the network. Additionally, the signals’ correlation suggested an ability in facing faults. This has been assessed, showing that the robot maintains the behaviour also when some signals disappear.

# 7 Experiment: area avoidance

The collision avoidance test evaluated the ability of the robot to move in a world with obstacles. Nevertheless, the presence of items in the environment simplifies the task, in that they can physically block the movement. The robot has thus to react quickly to get a good score, but an immediate reflex is not needed. The area avoidance test aims to evaluate the ability of the network to provide a fast response to stimuli, without the help of other entities. Additionally, the task is a way to assess the phenotypical plasticity of the robot when a similar selective pressure (i.e., objective function) is used in a different environment.

## Task and arena

To clearly define the task, it is important to initially describe the arena in which it will take place. This is a flat area, empty of obstacles. The physical items are indeed substituted by coloured zones on the floor (see Figure 32). The white ones are the illegal areas, and represent an intangible obstacle for the robot. Passing over them is indeed a not-allowed action that will penalise the performance. To allow the robot to foresee the illegal areas, those are surrounded by gray borders. No penalty is applied when passing over them: they are only needed as a warning. Finally, the black areas are normal zones on which the robot can wander.

The definition of the arena considered the robot configuration. As already discussed in Chapters 5, the use of normally high or low input signals drastically changes the behaviour of the network. The result may consist in having the device normally excited, or vice versa. At the moment, this problem can be managed only by manually changing the pre-processing logic of the robot (i.e., negating the reading). This is a limit of the optimisation approach. Since this factor cannot be automatically modified, the arena is designed to provide normally low signals. This allows the robot to work in the normally-relaxed state, that is supposed to perform better due to the higher dynamic[45].

[45] This is because the relaxation is slower than the excitement

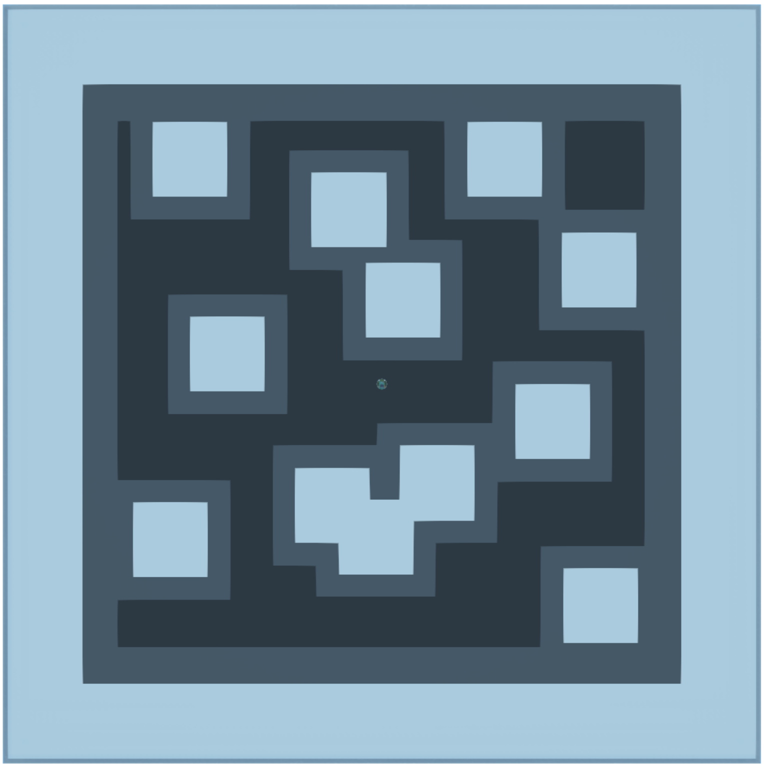

Figure 32: Top view of the arena used for the area avoidance task. The white fields are the 'illegal' places where the robot cannot go. The gray borders are a warning to the robot, saying that it is going toward an illegal area. No penalty is applied on them. Finally, the black areas are normal movement spaces. The outer border is illegal in order to confine the robot and to penalise configurations that only move ahead.

## Objective function

To make the EPuck complete the task, it needs a selective pressure that can guide its adaptation. This is represented by feedback that the robot can autonomously calculate. The function must consider the penalties and the rewards that can be obtained during the run, and should carefully evaluate their impact in the optimisation. High returns or punishments can indeed cut out some interesting adaptive paths[46]. In the specific task, the complexity is not particularly high.

[46] For example, the choice of using relevant penalties may cause the optimisation process to converge on configurations that are expert in avoiding the illegal areas but are less able to

Because of that, the chosen objective function highly penalises wrong actions[47], decreasing the score by 100 at each wrong step. Additionally, we want the robot to mostly move ahead, avoiding circular motion paths. The desired function is then obtained as a modification of the one used in the collision avoidance task, with the proximity penalty substituted by the illegal area one. The performances on each step are then summed and normalised in a $[0,100]$ range:

$$(1-|v_l(n)-v_r(n)|)\cdot\frac{v_l(n)+v_r(n)}{2}-100\cdot\theta(n) \tag{4}$$

where:

- $n$ is the actual step;
- $\theta(n)\in\{0,1\}$ return 1 if the robot is on the illegal area, 0 otherwise;
- $v_l(n), v_r(n)\in[0,1]$ respectively are the normalised left and right velocities of the robot wheels at step $n$.

Note that no explicit reward is given if the robot is on a normal (i.e., gray or black) area. Positive scores depend indeed only on the motion behaviour.

## Configuration

As said, the experiment in the area avoidance maze aims to evaluate the ability of the robot to act in a less favourable environment. The increasing complexity is given by a more punitive objective function. Additionally, the experiment assesses the phenotypical plasticity of the robot. This is done evaluating how the behaviour adapt when the same networks are placed in different environments with similar objective functions. The assessed devices are indeed created with the same seeds used in the collision avoidance task. This means that every network previously used is tested also in the area avoidance maze. The results should give an additional confirmation of the hypothesis according to which the network (i.e., the genotype) is not the only responsible for the emergence of a behaviour (i.e., the phenotype).

wander in interesting ways. The choice thus depends on the goal of the task

[47] For more complex tasks, we may decide for a more indulgent objective function

Most of the robotic and experimental parameters used in this test are already discussed in Chapter 5. Therefore, we will discuss only the ones that differs from the standard. Differently from the collision avoidance task, the epoch duration is 400 steps. This is needed in that the area avoidance is more subject to lucky movement of the robot, that may be able to avoid the illegal areas just due to its direction[48]. An increment in the evaluation time thus increases the validity of the experiment. The sensor provides ground measures, that are adjusted according to the custom range chosen in Chapter 6[49]. This causes a higher response of the network and allows a comparison with the collision avoidance results. Specifically, the Configuration II of Table 1 in Chapter 6 is used for the comparison. Finally, the change in the objective function determines a revision of the fitness adaptive threshold. From the value of 40.0 used in the collision avoidance task, this is changed to a value of 70.0.

## Results

The first step for the analysis of the results quality, consists in selecting some threshold levels for the categorisation between good and bad solutions. The selection has been done discretionary, with the values selected according to mathematical considerations and direct observations. The objective function is calculated applying a penalty of 100 in case the robot override an illegal area. Compared to this value, the prises are almost negligible. Giving an example, a fitness score of 70 would be given to configurations that are able to stay out of the illegal areas $\sim 70\%$ of the time (i.e., 280 steps out of 400). According to these considerations and to visual evaluations, the ranges chosen for the classification of the simulation results are the ones shown in Table 5.

Given the ranges, it is then possible to assess the results in a statistical way, identifying how the configurations performed and according to which metric.

[48] This is due to the greater size of the arena

[49] The range was decreased from [0, 4095] to [0, 1550], causing a multiplication factor of $\sim 2.6$

| Range | Classification |
|---|---|
| $[-\inf, 75)$ | Fail |
| $[75, 90)$ | Good |
| $[90, \inf]$ | Success |

Table 5: Ranges for the classification of the configurations results.

**Density influence**

The analysis of the results considers also the influence of some parameters. The density is thought to be one of the most important, and potentially one of the most influent. Therefore, the evaluation considers the changes in the performance according to the variation of this parameter. The data consider the results of 120 configurations for each value of density[50].

The results highlight a relation between performance and density. Lower values seem to help in achieving better scores. The percentage of *successful* instances differs by 13.34% between the lowest and the highest density (see Table 6). Nevertheless, this value seems to not influence the percentage of failing configurations, that remains $\sim$ 20% for all the three values.

| Density | $[-\inf, 75)$ | $[75, 90)$ | $[90, \inf]$ |
|---|---|---|---|
| 5.00 | 23.33 | 37.50 | 39.17 |
| 7.48 | 23.33 | 45.00 | 31.67 |
| 10.00 | 20.00 | 54.17 | 25.83 |

Table 6: Percentage of configurations that achieved a value of fitness in the specified ranges. The data are differentiated according to the network creation density.

The percentage of successful instances is obviously related to the average score obtained by the density values. Nevertheless, this measure is useful to have an idea of the expected result. Taking all the configurations that use a network of a given density, and calculating the average of the best scores, it is visible that the results differ by some percentage points (see Figure 33). This

[50] 30 replicas * 4 loads = 120 configurations

emerges also by looking at the distribution of the best performances (see Figure 34). The variation is not particularly big, with all the densities able to reach the first threshold, averagely. The second threshold is instead reached by the configurations that belong to the fourth quartile.

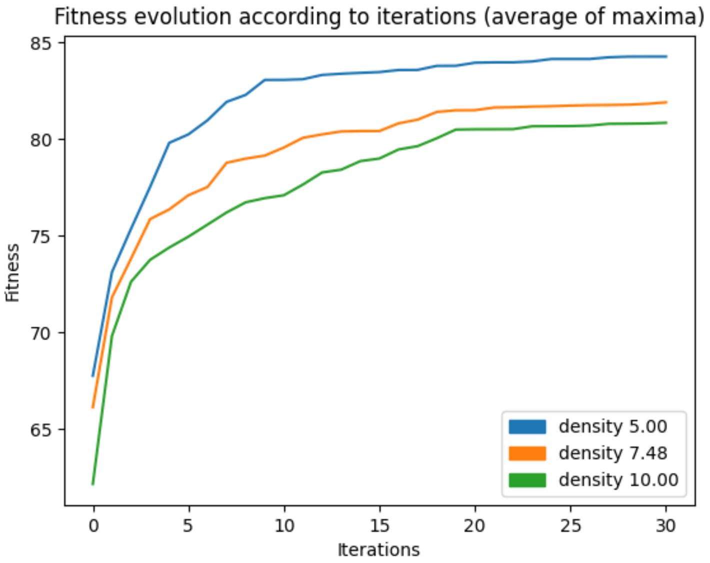

Figure 33: Adaptation of the fitness according to the iterations. The value is calculated by taking the average of the best fitness of each configuration that uses a specific value of density.

### Load influence

The second parameter to assess is the influence of the loads in the computation. The data consider the results of 90 configurations for each value of load[51]. As expected, their impact in the area avoidance task is greater than in the collision avoidance. This is due to the fact that a faster and stronger response is needed in the experiment. Oddly enough, however, the performances increase as the loads get smaller. This is an unexpected result. Due to the simplicity of the task, a simpler controller was expected to perform better. As described in Chapter 3 indeed, increasing the load induces a higher power on the motors and a lower

[51] 30 replicas * 3 densities = 90 configurations

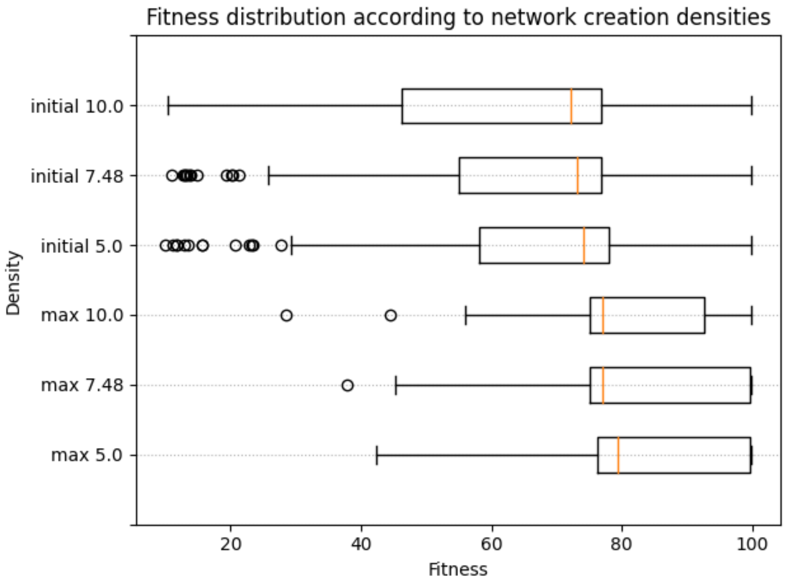

Figure 34: Initial and maximal fitness distributions according to the network creation density.

stimulation on the network. Nevertheless, the results highlight an incredible improvement in the performances when low loads are connected (see Figure 35). The spread of the percentages of *successful* instances between the smaller and bigger load is indeed 75.56% (i.e., more than one quarter of the runs; see Table 7).

| Load | $[-\inf, 75)$ | $[75, 90)$ | $[90, \inf]$ |
|---|---|---|---|
| 1e+03 | 8.89 | 15.56 | 75.56 |
| 1e+04 | 30.00 | 23.33 | 46.67 |
| 1e+05 | 50.00 | 43.33 | 6.67 |
| 1e+06 | 0.00 | 100.00 | 0.00 |

Table 7: Percentage of configurations that achieved a value of fitness in the specified ranges. The data are differentiated according to the motor load.

By looking at the distribution of the fitness at the initial and final step, we can clearly see that higher loads prevent any improvement in the solution (see Figure 36). Instead, as the resistance decreases, the performance increases.

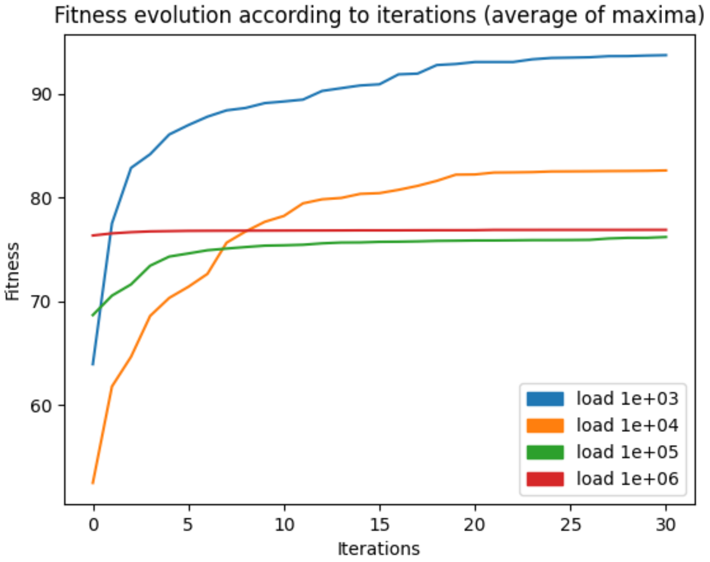

Figure 35: Adaptation of the fitness according to the iterations. The value is calculated taking the average of the best fitness of each configuration that uses a specific load value.

This is extreme in case of the lower load, that have its median almost at 100% of the score (see Figure 36).

### Overall performance

As an additional step, the evaluation considers the overall experiment results. On a total of 360 optimised configurations[52], the percentage of succeeding instances is 32.22%. The ‘good’ solutions are 45.56%. Finally, the failing instances are 22.22%. Therefore, the most common result consists in an instance that occasionally overrides the white (i.e., illegal) area. According to the classification, the ‘at least good’ results are 77.78%. Nevertheless, the differences due to the parametrisations cannot be ignored. It has been shown indeed how the results strongly depend on the densities and the motor loads. Accordingly, it results that the best performing configurations are the ones that use low values of density and load (see Figure 37). Indeed, the best instance, calculated on 30

[52] 30 replicas * 3 densities * 4 loads = 360 configurations

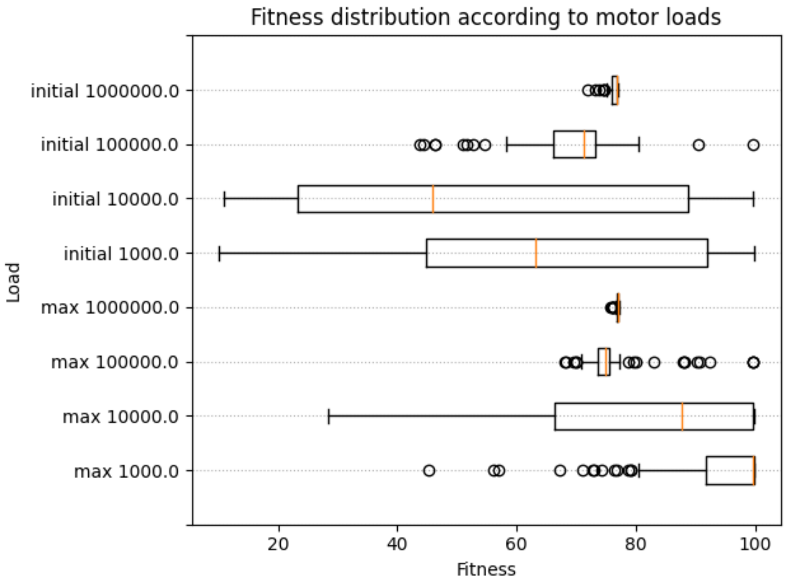

Figure 36: Initial and maximal fitness distributions according to the cascading motor load.

replicas, found most of its scores above the *successful* threshold.

### Phenotypical plasticity

The final step in the evaluation consists in assessing the phenotypical plasticity of the robotic system. As said at the beginning of the chapter, the experiment considered the same networks used in the collision avoidance task. Specifically, the instances are the ones optimised in the *custom range* evaluation (see Configuration II in Table 1). The results confirm the hypothesis that sufficiently complex networks (i.e., the genotypes) are able to express different behaviours (i.e., phenotypes). The same instances were indeed able to perform well both in the collision and area avoidance experiments. Considering the best scoring loads[53], the behaviours allowed the robot to achieve good results in both the tasks, respectively 78.89% and 75.56% of the times for collision and area avoidance.

[53] The selection of the load value determines the complexity of the network behaviour, with high resistances that reduce the internal dynamic. Therefore, we can assume that not-enough stimulated systems are not complex enough to generate a successful phenotype

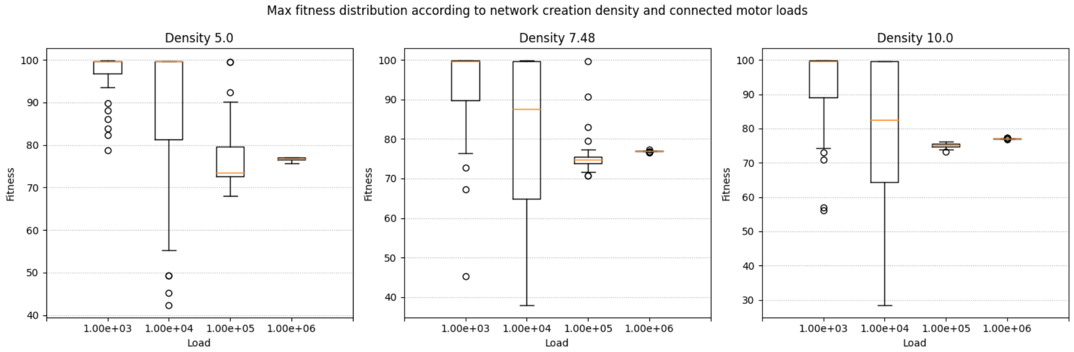

Figure 37: Maximal fitness distributions according to the network creation density and cascading motor load. Note that the fitness range of the rightmost plot is [30, 100] instead of [40, 100] used by the others.

Additionally, the similarities between the objective functions (i.e., the selective pressure) suggests that the characteristics of the environment directly affect the emergence of phenotypes. In the opposite case, we would see the same or similar behaviours emerge from the optimisation. Instead, the phenotype adapt to the specific characteristics of the arena, allowing the robot to score well in both.

## Conclusion

The analysis assessed the ability to successfully complete the experiment and the influence of the configuration parameters on the result. It turned out that both the density and the motor load influence the result. Specifically, this latter one determines a huge impact on the results, with too high values not being able to properly complete the task at all. This consideration suggests that the presence of the network is indeed necessary, and it does not simply act as a relay. In the opposite case, we would see an opposite behaviour[54].

The experiment also assessed the phenotypical plasticity of the system. This was evaluated considering the capability to adapt both to the collision avoidance and the area avoidance tasks. The results shown that the networks are able to

[54] As said, the higher the load, the higher the power on the motors. This would cause a decrement in the voltage falling on the network, and thus a reduced influence of it

express different behaviours (i.e., phenotypes) according to the environment and selective pressure.

# 8 Experiment: T-maze

The previous experiments respectively evaluated the capability of the system to perform a simple task (see Chapter 6) and its reflexes (see Chapter 7). Additionally, they assessed the adaptability of the network to different tasks and environments, showing phenotypical plasticity. Nevertheless, the exploitation of the intrinsic memory of the system was not attempted. Given the importance of this property, it is needed to evaluate if the nanowire network can be used to complete assignments that require an awareness of the past. This is tested through the optimisation of the T-maze task. Additionally, the results of two different adaptive approaches are compared.

## Task and arena

The T-maze task was historically used to assess the memory capacity of rodents. The goal was to show a switching behaviour, choosing the paths at the crossroad in an alternate way [12]. In this research, the task is modified. The aim is to allow an external stimulation to guide the robot choice. This requires an initially presented stimulus to be memorised for a short term. Therefore, the environment directly influences the robot, suppressing or enhancing some of its behaviours [31]. This is done through the use of an initial stimulation path, that may be both white and black (see Figure 38). The direction that the robot has to take is thus dynamically set by the initial colour. The goal is to reach the end-area with the opposite hue of the start. The score is influenced by the time spent outside the target zone, and thus by how quickly the target is reached. The robot shall pass to a gray area that is neutral and positioned at the end of the stimulation path, at the crossroad.

## Objective function

The goal of the task is for the robot to reach the end point of the maze. This indeed influences most of the performance. Nevertheless, also the time needed to reach the goal is important. Therefore, the objective function also considers this factor. This is implicitly done, in that positive feedbacks are obtained

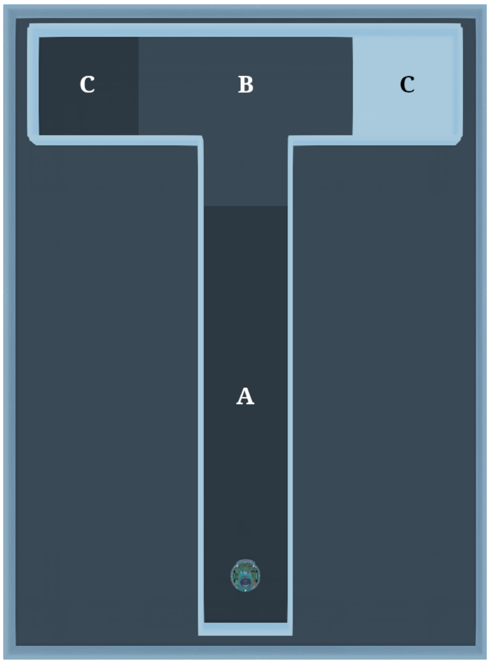

Figure 38: Top view of the arena used for the T-maze task. The gray area (B) is neutral, and no score is gained in it. It corresponds to the crossroad of the maze. The starting/stimulating area (A) is from where the robot gets the information about where to go. Its colour can be white or black depending on the direction. The areas (C) on the sides of the maze (i.e., after the crossroad, on the left and right) are the end-points. Their colour does not change during the execution.

only after the target is reached. Therefore, the more time is used to reach the goal, the lower the final score will be. The performances on each step are then summed and normalised in a $[0, 100]$ range:

$$2 \cdot \overline{\theta} \cdot \overline{\beta} - 1 \cdot \theta \tag{5}$$

where:

- $\theta \in \{0, 1\}$ is 1 if the floor is the same colour as the starting one;
- $\beta \in \{0, 1\}$ is 1 if the floor is gray.

The step objective function gives then a double, positive score if the colour of the floor is different from the starting one (i.e., $\overline{\theta}$) and from gray (i.e., $\overline{\beta}$).

Instead, if the robot is still on the initial area or in the wrong end-point (i.e., $\theta$), it receives a penalty. Finally, staying in the gray area (i.e., $\beta$) is not rewarded and neither penalised.

## Preliminary test

Preliminary tests did not manage to complete the task. The continuation of the experiments required then an intermediate, manual analysis to verify the feasibility of the goal. This has been done by taking a random network and manually selecting the input nodes (i.e., the nanowires), basically creating a custom thalamus. The connections are direct, with only an attenuation applied to the ground sensor[55]. This is needed both to avoid the robot turning on itself, and to avoid an overstimulation of the network[56]. The goal is to make both the motors subject to the proximity sensor, with one of them slightly more sensible to it. This translates in having the input connected with the outputs through two differently resistive paths. The aim of this uneven balance is to allow a standard manoeuvre also in the absence of an initial stimulation[57]. The opposite turning is achieved by making the ground sensor influence the most resistive path. The stimulation increases the conductance, making the output more powered and the connected motor slower[58]. This balance-shift causes the robot to turn on the opposite direction when the obstacle is perceived.

This custom solution shows the ability to complete the task, allowing the robot to turn on one direction when subject to a given stimulus, and vice versa (see Figure 39). The result thus confirms that the endogenic memory of the network can be successfully exploited. The need is thus for the optimisation system to produce good performing solutions. The choice of the input/output

[55] It corresponds to an attenuation of 70% of the signal, with a maximum value of $3V$

[56] An overstimulation may cause the network to remain stimulated by a previous test, influencing the result of a subsequent one. With a custom range, this problem is resolved, allowing only stimulus that the robot can recover from

[57] Note that the turning happens only when the robot perceives the wall. Before of the perception, the output is almost 0 and thus the robot proceeds straight. This behaviour is due to the chosen working ranges of sensors and actuators (see Chapter 5)

[58] This is due to the negation of the output signal of the network

nodes (i.e., the nanowires) does not seem to be a problem. For small networks, the number of nanowires is indeed limited. Nevertheless, the selection of the input range or the multiplier of the signals is more complicated. A naive solution may require the user to specify the parameter according to observations. However, this approach limits the potential and partially violates the concept of the adaptive computation. Moreover, it may lead to worse solutions. Therefore, the idea consists in allowing the optimiser to also automatically balance the sensors influence in the network. This was effectively done after the assessment of this result, with the implementation of a signal weighting.

Finally, as an additional consideration, the sequence of floor colours between multiple runs seems to not influence the ability of the robot to complete the task. A concern was indeed that the end-area may stimulate the network before the kidnapping[59], changing its subsequent behaviour. Nevertheless, the results do not show any related problem. The hypothesis is that the length of the stimulation path allows the network to 'reset' its state. A conclusion is that the influence of old stimulations does not prevent the robot to succeed the task.

## Configuration

Each experiment consists of different configurations and aims to understand behaviours or confirm hypothesis. For the T-maze task, the goal is to explain if the nanowire network can exploit its plasticity to store and use memory information. This is assessed by verifying that the robot can succeed in completing the maze. Additionally, it is required for the adaptive algorithm to be able to successfully optimise most of the configurations. If this last point is not guaranteed, one may argue that the obtained results are due to lucky configurations.

Most of the parameters of the experiments are standard, defined in Chapter 5. Nevertheless, some that are specific to the T-maze task, or were not previously described, exist. These newly defined parameters are here presented. The experiments take place in the T-maze arena and use a combination of one ground and two proximity sensors. The epoch duration is 1000. Every 250

[59] After one run the robot is automatically brought back to the starting point

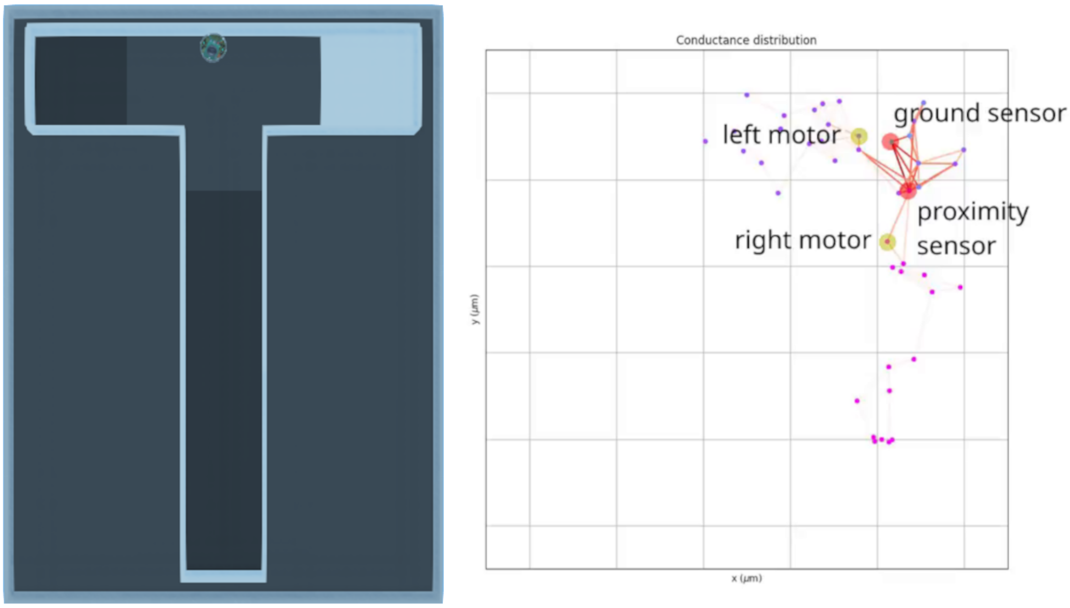

Figure 39: Frame of the manual test of the T-maze task. On the left, the position of the robot in the arena is visible. On the right, the nanowire network state is shown. Although the stimulation seems slightly biased toward the left-motor, the path is still more resistive. This is due to the number of subsequent wires and junctions (i.e., resistors). The result is a lower electrical potential on the left output node. Due to the negated logic, the right-motor results thus slower than the other, causing the robot to turn right. With a higher stimulation from the ground sensor, the voltage distribution flips, causing an opposite behaviour.

steps, a new round starts, and the configuration is tested with a different initial condition (i.e., alternatively, black or white stimulation). The epochs count is 200. This high amount is needed in order to allow the optimisation. The T-maze task resulted indeed harder to complete compared to others. Each combination is replicated 5 times to improve the robustness of the results. Nevertheless, the population is still small for a reliable, statistical analysis. This is due to the length of the computation, that inhibits more run to be performed[60]. With the aim of speeding up the search, the configurations are adapted only if they

[60] A run composed of 5 replicas, 200 epochs, 1000 steps of duration takes more than a week of computation

obtain a minimum fitness of 40.0[61]. In case a lower performance is obtained, the connections are re-created from scratch. As final point, the experiments evaluate two types of input pre-processing: (*i*) weighting; (*ii*) adaptation to a custom range. The first (*i*), applies and adapts a specific multiplier to each sensor signal. The second (*ii*), adapts the inputs to a fixed, hand-selected range, raising their voltage influence[62]. A direct connection is not assessed due to the results obtained in Section 'Preliminary test', that show an inability to complete the task.

## Results

In this section, the results of the experiments taking place in the T-maze arena are assessed. This involves the evaluation of the influence of various parameters in the performance. Nevertheless, to be able to analyse the results, a categorisation between 'good' and 'bad' solutions is needed. This is done by defining three ranges of results, each of them representing a further step in the task completion (see Table 8).

| Range | Classification |
|---|---|
| $[-\inf, 40)$ | Fail |
| $[40, 50)$ | Good |
| $[50, \inf]$ | Success |

Table 8: Ranges for the classification of the configurations results.

The first range contains the failing configurations. Those are the ones that make 'errors', terminating in the wrong end-point or remaining in the starting area for too long. By observation, a value below 40 is indicative of bad solutions. The second range is from 40 to 50. This represents 'good' solutions. Those are the ones that never end up in the wrong end-point nor remains in the initial area for long. Nevertheless, those configurations typically fail in reaching both the

[61] The choice of the value corresponds to the first threshold of the results' classification (see Section 'Results'), meaning that the robot never goes in the wrong target. Nevertheless, it also means that the robot may block in the middle of the maze

[62] The range is [0, 1550], meaning that values equal or higher will be set to 10V

targets. The common behaviour consists in reaching one of the end-points when a stimulation is presents and wandering in the gray area for the remaining time. Although not successful, these configurations are labelled as ‘good’ in that they show some kind of memory, going on one side if the stimulation meant so, but avoiding it otherwise. Finally, the last group considers the configurations performing more than 50. Those are the successful ones, in that able to reach both the end-points. Indeed, due to the duration of a run, values over 50 are possible only for robots that always succeed. Higher results are indicative of the time needed by the robot to reach the desired areas. These thresholds are chosen with a cautious approach. This means that, although lower values may already present a successful behaviour, the ranges start when almost all the configurations succeed.

### Overall considerations

The T-maze experiments consist in the test of two distinct configurations, differentiated by the adaptive approach. To analyse the single parameter influence, it is needed to firstly present the results. The first (*i*) approach, using the weighting pre-processing, obtained the results in Table 9. Instead, the second (*ii*) approach using an adaptation to a custom range produced the results in Table 10.

As explained in the Section ‘Configuration’, the results were calculated on a reduced amount of executions. Specifically, the statistics of each density are calculated on a total of 20 samples[63], while the load ones used just 15 runs[64]. Due to the reduced population, both the results are not statistically relevant. Nevertheless, the discovery of strong patterns in the data may be symptom of interesting properties. This may lead to future evaluations, and help in predict the influence of some configurations in complex tasks.

To enhance the statistical reliability of the data, it is proposed to perform a combined evaluation of the two runs. Although different in the approach and in the results, the overall behaviour should help in the search of some patterns in

[63] 5 replicas * 4 loads = 20 configurations

[64] 5 replicas * 3 densities = 15 configurations

| Density | $[-inf, 40.0)$ | $[40.0, 50.0)$ | $[50.0, inf)$ |
|---|---|---|---|
| 5.00 | 15.00 | 50.00 | 35.00 |
| 7.48 | 10.00 | 40.00 | 50.00 |
| 10.00 | 5.00 | 35.00 | 60.00 |
| Load | $[-inf, 40.0)$ | $[40.0, 50.0)$ | $[50.0, inf)$ |
| 1e+03 | 6.67 | 40.00 | 53.33 |
| 1e+04 | 6.67 | 46.67 | 46.67 |
| 1e+05 | 13.33 | 40.00 | 46.67 |
| 1e+06 | 13.33 | 40.00 | 46.67 |
| Average | 10.00 | 41.67 | 48.33 |

Table 9: Percentage of configurations that achieved a value of fitness in the specified ranges. The data are differentiated according to the network creation density and load. The results are generated by the 'weighting' (*i*) test configuration.

the parameters choice. The hypothesis is that densities and loads influence the results regardless of the adaptive technique used (i.e., weighting or adaptation to a custom range). In other words, a parameter that positively influences the performance, is expected to do the same with both the approaches.

Both the weighted (*i*) and the custom range (*ii*) adaptive approaches are able to successfully complete the task, almost or more than half of the time. Using a joint evaluation, the result is that 55.83% of the instances are categorised as successful[65]. Conversely, the failing configurations seem to be a negligible part. This highlights a capability of the optimiser to generally allow the robot to exploit some sort of memory. As explained at the beginning of the section, also the ability to stop in the gray area is considered a symptom of awareness. Although the results do not show an overwhelming presence of fully successful configurations (i.e., performing more than 50), the memory of the network seems likely to be exploitable.

[65] The percentage is calculated on 5 replicas * 3 densities * 4 loads * 2 approaches = 120 samples

| Density | $[-inf, 40.0)$ | $[40.0, 50.0)$ | $[50.0, inf)$ |
|---|---|---|---|
| 5.00 | 15.00 | 30.00 | 55.00 |
| 7.48 | 0.00 | 35.00 | 65.00 |
| 10.00 | 5.00 | 25.00 | 70.00 |
| Load | $[-inf, 40.0)$ | $[40.0, 50.0)$ | $[50.0, inf)$ |
| 1e+03 | 0.00 | 26.67 | 73.33 |
| 1e+04 | 0.00 | 40.00 | 60.00 |
| 1e+05 | 20.00 | 20.00 | 60.00 |
| 1e+06 | 6.67 | 33.33 | 60.00 |
| Average | 6.67 | 30.00 | 63.33 |

Table 10: Percentage of configurations that achieved a value of fitness in the specified ranges. The data are differentiated according to the network creation density and load. The results are generated by the 'custom range' (*ii*) test configuration.

### Density influence

One of the parameters that may influence the ability of the configuration to succeed is the creation density of the nanowire network. In the previous experiments, this parameter shown to effectively influence the results (see Chapter 6 and 7). This is thought to be valid also for the T-maze task. The hypothesis is that the higher need of memory requires more complex networks to be used[66].

Differently from the area avoidance tasks, the results show denser networks generating more successful instances (see Table 9 and 10). The percentage increases sharply from low to high values. Depending on the adaptive approach, the spread is of 25 and 15 percentage points, resulting in an average increment of 20%[67]. This suggests that high values of density determine a general improvement in the amount of successful instances. Nevertheless, by looking at the scores' distribution, this result seems less certain (see Figure 40). Although less statistically relevant[68], a clear distinction is visible only for the 'weighted'

[66] The memory is expected to be related to the dimension of the network state

[67] This value is calculated on 40 samples: 5 replicas * 4 loads * 2 approaches

[68] The statistics of each approach use only 20 samples

approach. This result mines the previous one, requiring more tests to be produced. Therefore, a more complete analysis is postponed to future works with a greater population of results.

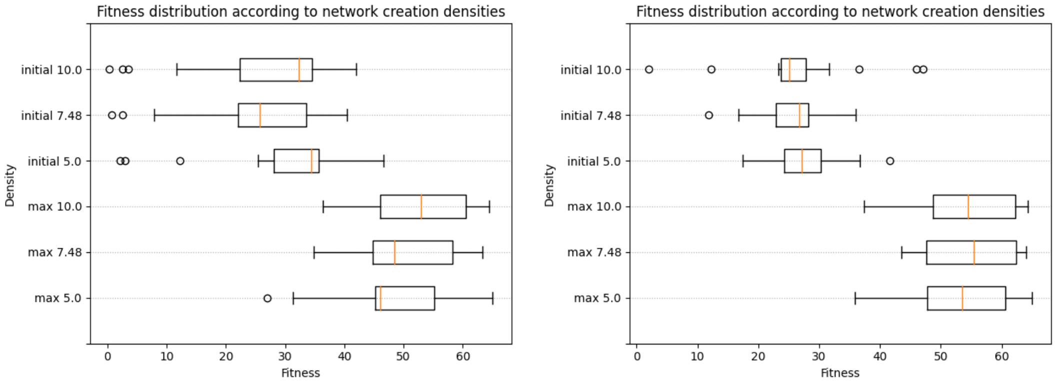

Figure 40: Distribution of the scores according to the network creation density. On the left, data of the ‘weighted’ approach are plotted. On the right, the results of ‘custom range’ strategy are shown. The plots show both the initial and final distribution, in order to allow the comparison of the improvement.

### Load influence

The other parameter to evaluate is the motor load. The choice of a good value, shown to be extremely important in some settings (e.g., in the area avoidance). Its evaluation is thus needed in order to understand which kind of stimulation better drives the robot in the T-maze[69]. The results suggest that low loads score better than higher ones (see Table 9 and 10). This is in line to what was analysed in the area avoidance task. Nevertheless, the drift in performances is reduced. This result is partially unexpected. The formation and storing of memory inside the network is thought to be facilitated by high intensity stimulations. Therefore, tasks that require awareness of the past (e.g., T-maze) are expected to be more sensible to the connected loads, compared to more reactive ones (e.g., area avoidance). This hypothesis is not confirmed by the data, suggesting that also low stimulations are enough for the task completion.

[69] The stimulation of the network is indeed related to the motor load

The previous results have however to be carefully accepted. As for the density analysis, the amount of data is limited. Additionally, if the density had an amount of 40 samples, the load has just 30[70]. This further reduces the statistical relevance of the result, and precludes the consideration of the parameter influence in the single approach[71].

### Adaptation strategy

One of the points raised in the Section 'Preliminary test' revolves around the need of a pre-processing of the input signal, in order to successfully perform the task. The experiments explored the possibility of using both weighted links (*i*) and custom input ranges (*ii*) to tackle the problem (see Section 'Configuration'). Therefore, the analysis consists in the comparison of the performances obtained with the two approaches. The goal is to identify differences, in order to comprehend how a methodology impacts on the result. Strong variations may suggest indeed an efficiency of one of the two approaches, or vice versa. The amount of samples for the evaluation of each strategy are 60[72].

A primary point in the evaluation of these methodologies, is that both are able to complete the task. This result is important, in that it resolves the problem of the unprocessed connections toward the system, that were almost never able to allow the robot to complete the maze. Therefore, the two novel strategies provide better results, and represent valid approaches for the task optimisation.

The results present averagely higher performances when the 'custom range adaptation' (*ii*) is used. The 'weighted link' (*i*) approach generates indeed lower scores. Nevertheless, the greater adaptability of this latter one has to be acknowledged. The ranged solution requires indeed a handmade work, needing to select the parameter discretionary. This approach is feasible for a limited amount of sensors types, but may become overwhelming as the number increases. Additionally, this value is hardcoded and cannot be modified by the robot itself. From this point of view, the 'weighted-links' approach gives

[70] 5 replicas * 3 densities * 2 approaches = 30 samples

[71] For each load, there would be a total of only 15 samples

[72] 5 replicas * 3 densities * 4 loads = 60 samples

more possibilities to adapt. Firstly, it allows the system to automatically select the weight. This is especially useful in that we are discussing online training, that allows a continuous adaptation. The hypothesis is that the system will eventually converge to a good solution, remaining able to adapt to possibly extreme changes in the environment. The second point, is the possibility to optimise the signal of the single sensor. This is not particularly useful in the T-maze, but may become needed in more complex tasks.

Assessing the results quantitatively (see Table 9 and 10), we can see that the amount of successful configuration passes from 48.33% for the weighted approach, to 63.33% for the 'custom range' one. The spread of the results is thus 15%. The difference between the percentages of failing instances is instead low (i.e., 3.33%), meaning that both the approaches are sufficiently good to exploit the network memory. Indeed, the major difference consists in the passage from the second to the third success range. Nevertheless, by assessing the fitness distribution of the two strategies, it is possible to see that the 'weighted' pre-processing generates an effectively worse set of solutions (see Figure 41). This result is contrary to the ones obtained in the collision avoidance task, where the approaches closely matched each other.

An additional point to discuss is the improvement of the performances in the run. The 'custom range approach' causes the fitness to improve rapidly, stabilising after some iterations. Conversely, the weighted strategy grows slowly, reaching similar results after much longer time (see Figure 42). From the point of view of the speed, the first approach is thus better. Nevertheless, no decay is visible in the improvement rate of the fitness for the 'weighted' approach. This may be due to a slower optimisation towards the same values obtained by the custom range[73], as well as a higher suitability to continuous adaptation. This latter aspect would allow the weighted approach to perform better in the long run. Nevertheless, no test considered this option, mostly due to the computational complexity of the simulation. Therefore, this possibility is actually just a speculation.

[73] This implies that the 'custom range' approach already reached the upper limit of the fitness

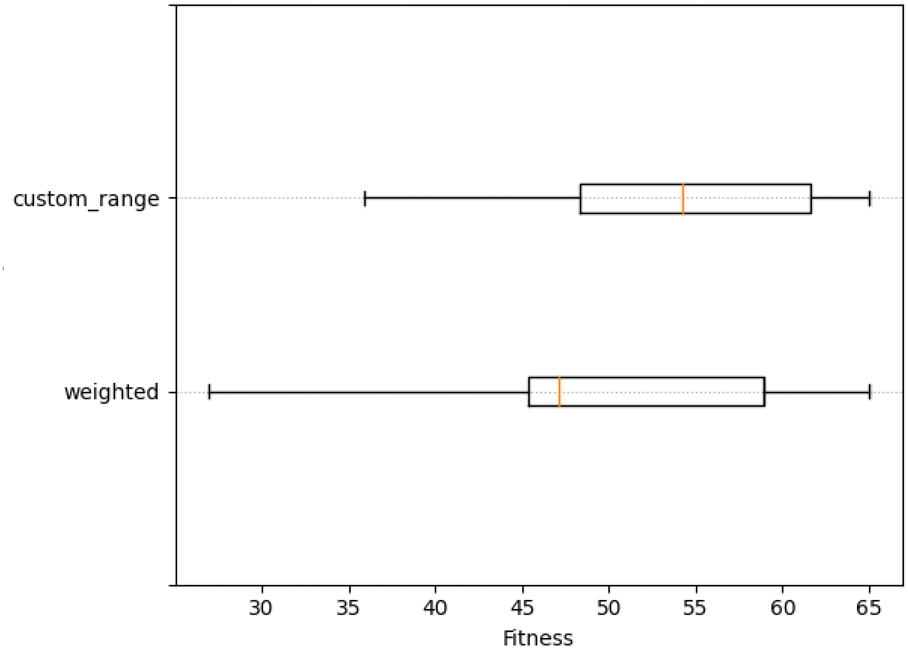

Figure 41: Fitness distribution according to the pre-processing strategy of the input.

## Conclusion

The experiment assessed the feasibility of the goal. The results show a general capability of the system to complete the task, at least when a sufficient number of executions is considered. This thus confirms the presence of an endogenic memory arising from the network dynamics.

Also, the influence of some strategies and parameters was assessed. Specifically, two adaptive approaches were tested, each of them providing a different pre-processing of the input signals. The results highlighted a better performance of the methodology adapting the signal to a custom range, compared to the one using weighted (i.e., multiplied or attenuated) links. The tested parameters were instead the density and the load, both suggesting some kind of influence. Nevertheless, their reduced amount of samples causes a reduced reliability of the results. The analysis of those parameters has thus to be considered a proof of concept for future works and studies.

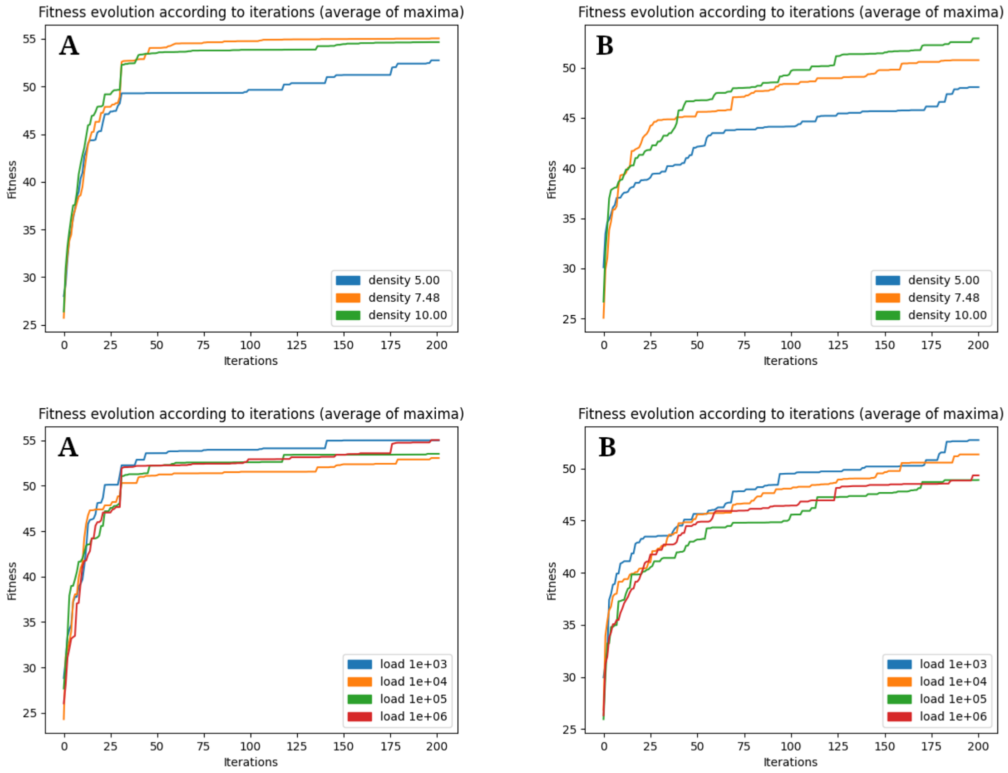

Figure 42: Adaptation of the fitness according to the iterations. The value is calculated taking the average of the best fitness of each configuration that uses a specific value of density (top) and load (bottom). On the left (A), the results from the custom range approach are shown. On the right (B), the results from the weighted link approach are shown.

# 9 Future works

This thesis paved the way for future works, relating robotics and nanowire networks. Because of the novelty of the approach, the amount of possible development is massive. In this chapter, a brief list of possible research paths is presented. Most of them emerged during the development of this dissertation, and represent alternative goals that it could have aimed at.

## Adaptation

One of the initial proposals, considered the use of the nanowire networks to create failure resistant and recoverable robots. Although the first aspect was considered during the work (see Chapter 6), no effort was made to maximise its effectiveness. Indeed, the resistance to some sensor failure was assessed only on already optimised configurations, and not for specifically trained robots. Regarding the recoverability instead, this was not evaluated at all. Therefore, a possible development would consider an explicit training of the network driven system, in order to resists and recover from failures.

Another aspect to assess, is the effective improvement of the performance when the nanowire network is exploited. Indeed, the experiments considered the success of the solutions, but a comparison with static circuit is missing. Therefore, an analysis without a dynamic network (i.e., without the junction plasticity) may be interesting. This can be easily achieved through the use of the simulator. Alternatively, a comparison with other types of networks or neuromorphic systems may be performed (e.g., Boolean or Neural networks).

The use of alternative, adaptive strategies may be a valid continuation of this work. The proposal is the creation of an algorithm that adapts the mutation impact according to the quality of the solution. The goal is to change extremely when the fitness is low, and moderately when a good performance is approached. For the weights, this may be done through the use of a variable sigma in a Gaussian mutation.

## Technical

As said (see Chapter 1), the nanowire networks have the potential for efficient and fast computations. Nevertheless, an effective measure of their impact is missing. Therefore, a proposed area of work relates to the comparison of consumption and computation capabilities with other types of systems. Some valid competitors may be Boolean and Neural networks. Additionally, the analog functioning of the nanowire devices is not limited by ‘clock cycles’, like for the computers. This allows their evaluations as real-time computing systems.

Considering the reduced size of the nanowires, an interesting possibility would be their use in the control of micro and nanobots. One of the problem is indeed finding a control system that can fit these entities. Some approaches exploited biology for this task [21]. These networks represent instead an artificial alternative. Nevertheless, to achieve this goal, a greater research effort is needed. Additionally, due to the limited impact of these robots, this idea necessarily have to mix with studies of swarm intelligence.

## Experimental

Considering future works related to the tasks, an initial point consists in evaluating the T-maze run in a more statistically reliable way. Indeed, in the current test, the number of replicas is limited by the time required for the computation. Nevertheless, with more computing power or with a physical device, this possibility should be assessed. Additionally, an evaluation on longer mazes should be performed, in order to confirm the persistence of the successful behaviour in time.

Still related to the tasks, the test of the system in a dynamical environment would provide information about the behavioural stability and adaptability. Therefore, this may also be an interesting path.

In case the tests start to become too hard to succeed through the use of a simple reconnection, a possibility may evaluate the use of a more complex readout. In this case however, a solution to the online adaptation problem described in Chapter 2 has to be found.

## Analytics

This thesis did not statistical analysed the characteristics of the successful configurations. Nevertheless, the search of some patterns in their organisation may help in understanding how the adaptation works and possibly converges. In future works, a similar evaluation should be considered.

## Nanowire networks

Until now, the presented possible future works mostly considered the robotic area. Nevertheless, some researches may deeply assess the nanowire networks performances and characteristics. An example is the evaluation of the density influence on the computation. In this dissertation, this was evaluated through the performances in the tasks. Nevertheless, formal approaches may apply some constraints to this analysis. An example is to force the input and output nodes to maintain a constant distance[74]. This would allow evaluating if the wider branching of denser networks effectively improve or reduce the performance.

A different analysis may consider alternative ways to induce a critical state in nanowire networks. According to the hypothesis described in Chapter 3, the performances are expected to greatly improve if a phase transition results exploitable.

Finally, a possible study may assess the use of external components to stabilise the signal or improve the memory of the device. Indeed, it is common in the robotic area the use of delayed feedback loops, for the stabilisation of the output and the increment of the storable data. This may extend the effectiveness of the nanowire network to more demanding fields.

[74] It may be both physical or resistive

# Conclusion

The dissertation evaluated the usage of nanowire networks for the control of a robotic system. This is trained by an online, adaptive strategy to complete tasks in specific environments.

The work articulated on multiple steps. The first (see Chapter 1), considered the motivations for the use of the nanowire networks as control systems. The resulting strength points are many. Its neuromorphic computation is suitable for the creation of biologically inspired systems. The similarities with the brain resulted also by: a reduced power consumption; potentially high scalability; the computing capabilities. Indeed, the device is analog and not limited by clock speeds, allowing for theoretically fast and efficient computations. This is enhanced also by the possibility of storing information directly in the system, approaching the problem of the overhead present in classical Von Neumann architectures. This list of benefits has been followed by a brief discussion about the networks' creation, and by the presentation of the simulator used in the project.

The next step consisted in the explanation of the aimed phenotypical plasticity. According to the work vision, the adaptation of the behaviour to tasks and environments is indeed a central point. This has been achieved through the use of an online, adaptive strategy. Additionally, the limitations concerning the use of Reservoir Computing in continuous adaptation have been presented, terminating in the proposal for a custom, mixed architecture (see Chapter 2). Part of the work has been indeed later devoted to the definition of a biologically plausible and architecturally feasible design (see Chapter 4). This resulted in the association between entities of the two domains. The main identified modules are the brain (i.e., the controller), the thalamus (i.e., the sensory connections), the pyramid (i.e., the control connections), the sensors and the muscles (i.e., the transducers).

The design of the system required however a previous step: an analysis of the network behaviour and characteristics (see Chapter 3). This helped in understanding how to properly define the architecture. The evaluated aspects concerned many parameters and design choices. For example, the distribution of the stimulation shown a relation with the density. Nevertheless, also the connected components shown to influence the device computation. Specifically, low external loads induced an increase in the network sensitivity. Finally, the most interesting analyses related to the memory, the separation property and the update frequency of the system. Specifically, the latter one shown to determine the presence of chaotic behaviours.

With the definition of the architecture complete, the description of the experiments begun. The approach consisted in the formalisation of the standard choices. This included the definition of the adaptive strategies, and the parameters used by the robot and in the experiments (see Chapter 5). Specifically, the chosen adaptive approach consisted in the re-wiring and re-weighing of the input connections from the sensors to the network.

The robot was then tested in distinct environments, with different tasks. The goal was to verify some behavioural capabilities. Specifically, the collision avoidance evaluated and confirmed the ability to complete a simple task. It also compared some adaptive strategies, showing that a pre-processing of the input signals is needed. Additionally, it studied a successful instance to understand how the optimisation converged to a good solution. This consisted in the evaluation of the connections' disposition, as well as the fault resistance of the system. Specifically, the latter one shown high resilience and mixing capabilities. The second task was instead the area avoidance. It assessed and confirmed the phenotypical plasticity of the network, testing the instances from the previous arena in a new setting. Also, it evaluated a more reactive behaviour. The final test was the T-maze. It considered the use of an intrinsic memory capacity, showing the ability to exploit it. Additionally, it also assessed two adaptive strategies.

The final step of the thesis, presented possible future works. It analysed various aspects, proposing changes in (*i*) the adaptation of the system; (*ii*) the technical evaluations; (*iii*) the experiments; (*iv*) the analysis of the results; (*v*) the assessment of the nanowire networks behaviour.

Overall, the system presented good capabilities in optimising according to a given objective function and environment. The solutions shown to be robust to failures and flexible to the type of task to complete. They also confirmed the possibility to exploit the intrinsic memory of the system. This is extremely important for a possible, future, wide use of this technology.

## Acknowledgement

The development of this thesis would have not been possible without the assistance of some people. The first thank goes to my supervisor, Andrea Roli, from the University of Bologna. He overviewed the work and helped in searching a viable approach in a wide sea of unexplored possibilities. Secondly, my gratitude goes to Carlo Ricciardi and Gianluca Milano, from the Polytechnic of Turin. They proposed the nanowire network here used and provided the first version of the simulator. Additionally, they gave suggestions on possible, future works.

Finally, my thanks go to my family, girlfriend and friends. Their help have been both economical and moral, supporting my studies until now.

# Appendix

## Experiment: collision avoidance - Density influence calculations

Configuration I in Table 1:

$$\begin{cases} \Delta D_{5.0} = 3*21.67/(21.67+27.5+25.0) = 0.8764999325872995 \\ \Delta D_{7.48} = 3*27.5/(21.67+27.5+25.0) = 1.1123095591209384 \\ \Delta D_{10.0} = 3*25.0/(21.67+27.5+25.0) = 1.011190508291762 \end{cases}$$

Configuration II in Table 1:

$$\begin{cases} \Delta D_{5.0} = 3*69.17/(69.17+79.17+80.83) = 0.9054850111271108 \\ \Delta D_{7.48} = 3*79.17/(69.17+79.17+80.83) = 1.0363921979316664 \\ \Delta D_{10.0} = 3*80.83/(69.17+79.17+80.83) = 1.0581227909412227 \end{cases}$$

Configuration III in Table 1:

$$\begin{cases} \Delta D_{5.0} = 3*30.83/(30.83+36.67+38.33) = 0.8739487857885287 \\ \Delta D_{7.48} = 3*36.67/(30.83+36.67+38.33) = 1.0394973070017954 \\ \Delta D_{10.0} = 3*38.33/(30.83+36.67+38.33) = 1.086553907209676 \end{cases}$$

Configuration IV in Table 1:

$$\begin{cases} \Delta D_{5.0} = 3*73.33/(73.33+81.67+78.33) = 0.9428277546822098 \\ \Delta D_{7.48} = 3*81.67/(73.33+81.67+78.33) = 1.0500578579693995 \\ \Delta D_{10.0} = 3*78.33/(73.33+81.67+78.33) = 1.0071143873483908 \end{cases}$$

Configuration V in Table 1:

$$\begin{cases} \Delta D_{5.0} = 3*55.00/(55.0+69.17+62.5) = 0.8839127872716558 \\ \Delta D_{7.48} = 3*69.17/(55.0+69.17+62.5) = 1.1116408635560078 \\ \Delta D_{10.0} = 3*62.50/(55.0+69.17+62.5) = 1.004446349172336 \end{cases}$$

$\Delta D$ for 5.0: $\frac{0.88+0.91+0.87+0.94+0.88}{5} = 0.8960000000000001$
$\Delta D$ for 7.48: $\frac{1.11+1.04+1.04+1.05+1.11}{5} = 1.07$

$\Delta D$ for 10.0: $\frac{1.01+1.06+1.09+1.01+1.00}{5} = 1.034$

Range of density influence for each Configuration in Table 1:

$$\begin{cases} I \longrightarrow max(\Delta D) - min(\Delta D) = 1.11 - 0.88 = 0.23 \\ II \longrightarrow max(\Delta D) - min(\Delta D) = 1.06 - 0.91 = 0.15 \\ III \longrightarrow max(\Delta D) - min(\Delta D) = 1.09 - 0.87 = 0.22 \\ IV \longrightarrow max(\Delta D) - min(\Delta D) = 1.05 - 0.94 = 0.11 \\ V \longrightarrow max(\Delta D) - min(\Delta D) = 1.11 - 0.88 = 0.23 \end{cases}$$

## Experiment: collision avoidance - Distribution of scores according to the load

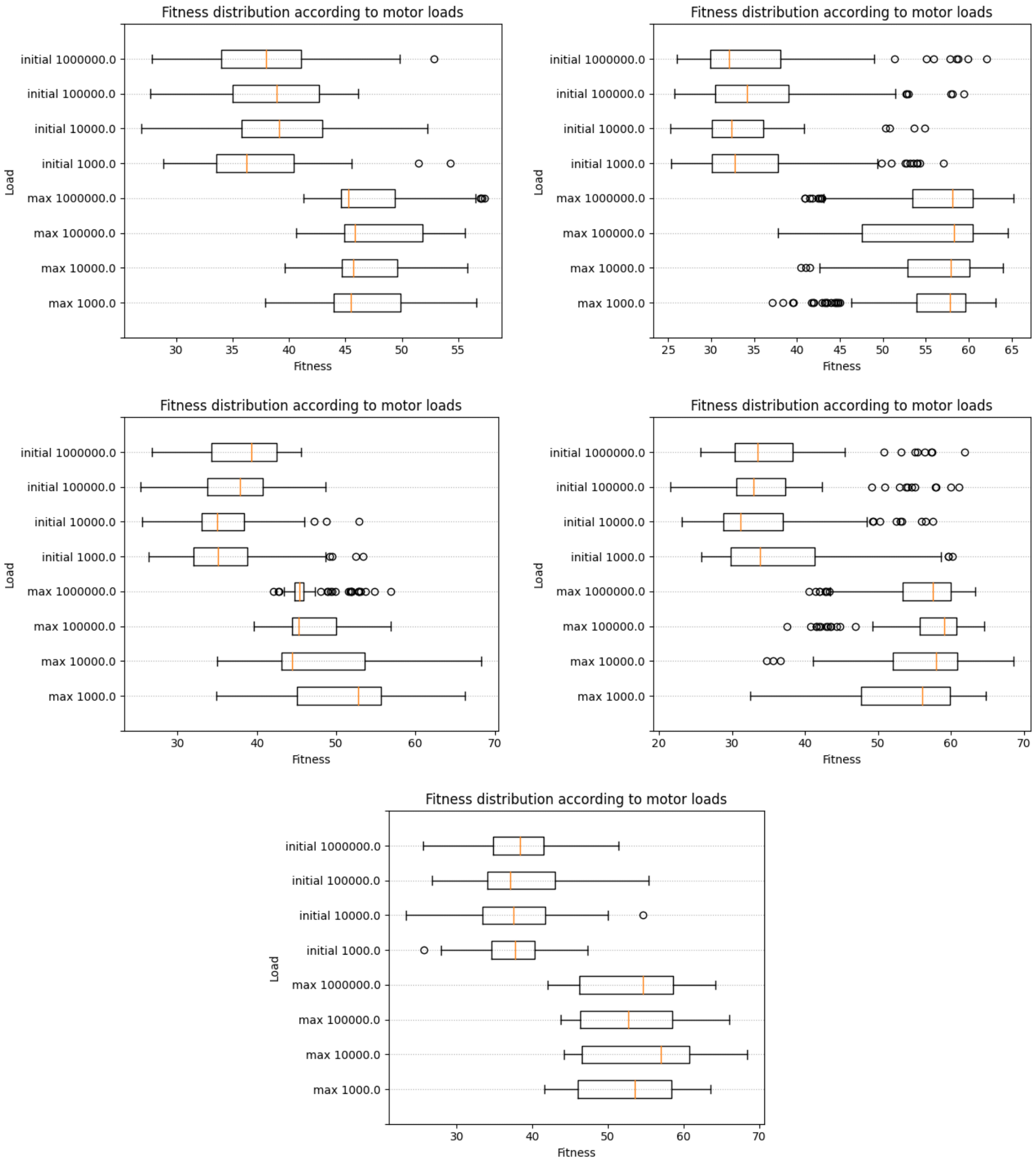

Figure 43: Fitness distribution according to the motor load. From left to right and from top to bottom, the data are the representatives of the configurations I, II, III, IV, V from Table 1.

## Experiment: collision avoidance - Analysis of distance sensing configuration

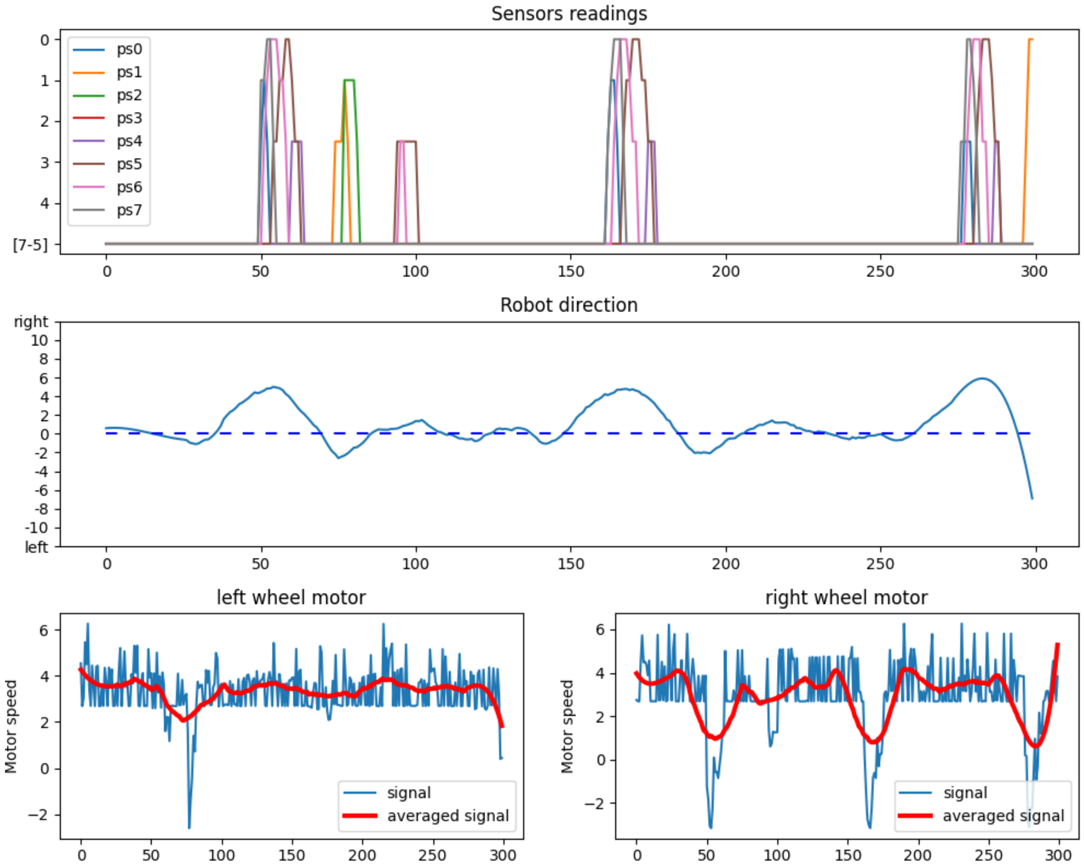

Figure 44: Analysis of input/output signals and consequent robot movement.

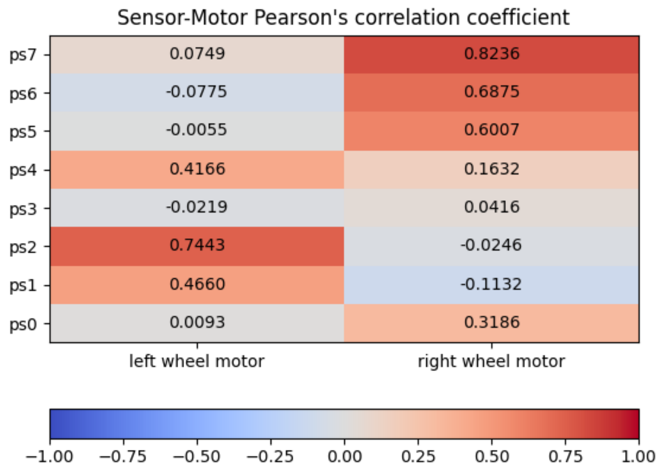

Figure 45: Analysis of input/output signals' correlation.

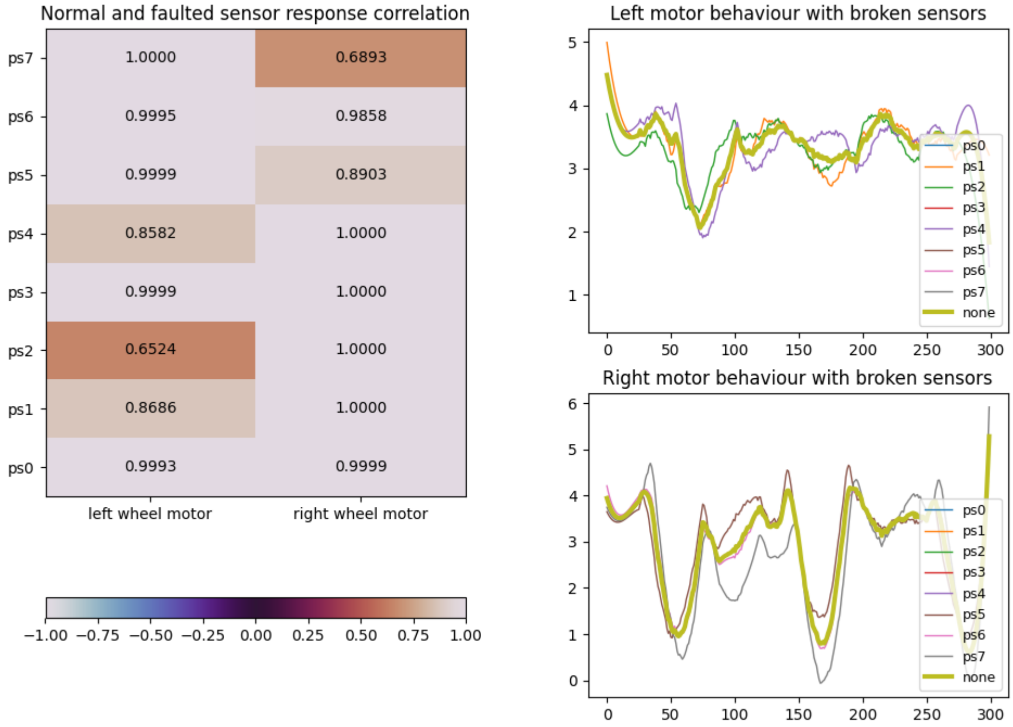

Figure 46: Analysis of output signals' correlation when subject to faults.

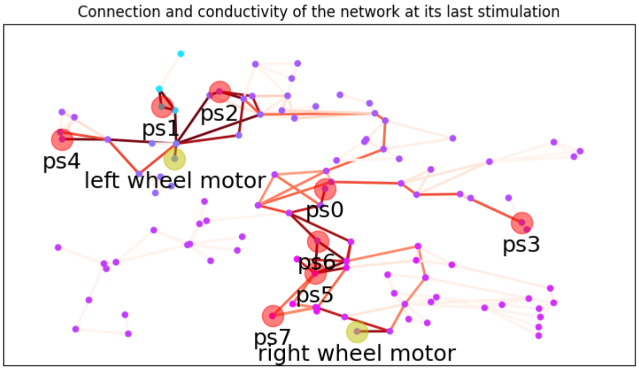
Connection and conductivity of the network at its last stimulation
ps1
ps2
ps4
left wheel motor
ps0
ps3
ps6
ps5
ps7
right wheel motor

Figure 47: Analysis of transducers connections to the network.

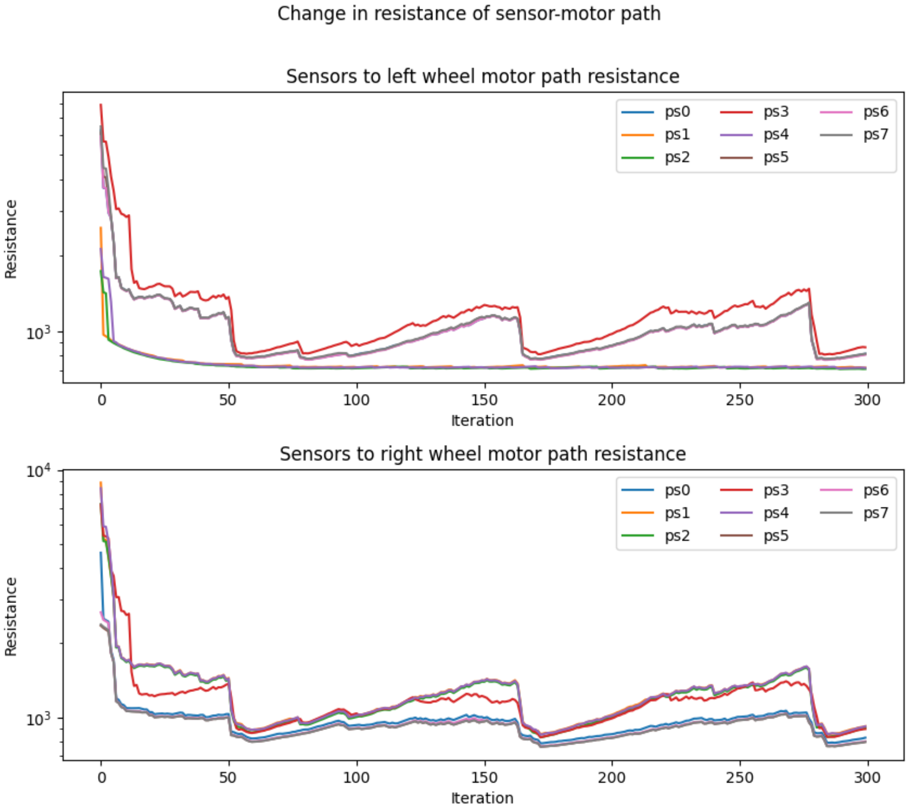

Figure 48: Analysis of paths resistances from sensors to motors.

# References

[1] Edoardo Barbieri. Adattamento online di robot controllati da reti booleane. Master's thesis, University of Bologna, 2021.

[2] Michele A Basso, Daniel Uhlrich, and Martha E Bickford. Cortical function: a view from the thalamus. *Neuron*, 45(4):485–488, 2005.

[3] Mauro Birattari, Antoine Ligot, Darko Bozhinoski, Manuele Brambilla, Gianpiero Francesca, Lorenzo Garattoni, David Garzón Ramos, Ken Hasselmann, Miquel Kegeleirs, Jonas Kuckling, Federico Pagnozzi, Andrea Roli, Muhammad Salman, and Thomas Stützleand. Automatic off-line design of robot swarms: a manifesto. *Frontiers in Robotics and AI*, 6:59, 2019.

[4] Michele Braccini, Andrea Roli, and Stuart A Kauffman. Online adaptation in robots as biological development provides phenotypic plasticity. *arXiv preprint arXiv:2006.02367*, 2020.

[5] Valentino Braitenberg. *Vehicles: Experiments in synthetic psychology*. MIT press, 1986.

[6] Jens Burger, Alireza Goudarzi, Darko Stefanovic, and Christof Teuscher. Computational capacity and energy consumption of complex resistive switch networks. *arXiv preprint arXiv:1507.03716*, 2015.

[7] Henry Vandyke Carter. The motor tract. (modified from poirier.), before 1858. Source: Anatomy of the Human Body; Public domain.

[8] Henry Vandyke Carter. Ways conducting sensitivity, before 1858. Source: Anatomy of the Human Body; Public domain.

[9] Dennis Valbjørn Christensen, Regina Dittmann, Bernabé Linares-Barranco, Abu Sebastian, Manuel Le Gallo, Andrea Redaelli, Stefan Slesazeck, Thomas Mikolajick, Sabina Spiga, Stephan Menzel, et al. 2022 roadmap on neuromorphic computing and engineering. *Neuromorphic Computing and Engineering*, 2022.

[10] Steven H Collins, Martijn Wisse, and Andy Ruina. A three-dimensional passive-dynamic walking robot with two legs and knees. *The International Journal of Robotics Research*, 20(7):607–615, 2001.

[11] William M Connelly, Michael Laing, Adam C Errington, and Vincenzo Crunelli. The thalamus as a low pass filter: filtering at the cellular level does not equate with filtering at the network level. *Frontiers in neural circuits*, 9:89, 2016.

[12] William N Dember and Harry Fowler. Spontaneous alternation behavior. *Psychological bulletin*, 55(6):412, 1958.

[13] Dario Floreano and Laurent Keller. Evolution of adaptive behaviour in robots by means of darwinian selection. *PLoS biology*, 8(1):e1000292, 2010.

[14] Gianpiero Francesca and Mauro Birattari. Automatic design of robot swarms: achievements and challenges. *Frontiers in Robotics and AI*, 3:29, 2016.

[15] Tsuyoshi Hasegawa, Kazuya Terabe, Toshitsugu Sakamoto, and Masakazu Aono. Nanoionics switching devices:“atomic switches”. *MRS bulletin*, 34(12):929–934, 2009.

[16] Walter Hendelman. *Atlas of functional neuroanatomy*. CRC press, 2005.

[17] Chung-Wen Ho, Albert Ruehli, and Pierce Brennan. The modified nodal approach to network analysis. *IEEE Transactions on circuits and systems*, 22(6):504–509, 1975.

[18] Giacomo Indiveri, Bernabé Linares-Barranco, Tara Julia Hamilton, André Van Schaik, Ralph Etienne-Cummings, Tobi Delbruck, Shih-Chii Liu, Piotr Dudek, Philipp Häfliger, Sylvie Renaud, et al. Neuromorphic silicon neuron circuits. *Frontiers in neuroscience*, 5:73, 2011.

[19] Herbert Jaeger. The “echo state” approach to analysing and training recurrent neural networks-with an erratum note. *Bonn, Germany: German*

*National Research Center for Information Technology GMD Technical Report*, 148(34):13, 2001.

[20] Eric R Kandel, James H Schwartz, Thomas M Jessell, Steven Siegelbaum, A James Hudspeth, and Sarah Mack. *Principles of neural science*, volume 4. McGraw-hill New York, 2000.

[21] Sam Kriegman, Douglas Blackiston, Michael Levin, and Josh Bongard. A scalable pipeline for designing reconfigurable organisms. *Proceedings of the National Academy of Sciences*, 117(4):1853–1859, 2020.

[22] Chris G Langton. Computation at the edge of chaos: Phase transitions and emergent computation. *Physica D: nonlinear phenomena*, 42(1-3):12–37, 1990.

[23] Giulia Lucchi. Meccanismi evolutivi per la progettazione automatica online di reti booleane per sciami di robot. Master's thesis, University of Bologna, 2020.

[24] Wolfgang Maass, Thomas Natschläger, and Henry Markram. Real-time computing without stable states: A new framework for neural computation based on perturbations. *Neural computation*, 14(11):2531–2560, 2002.

[25] Stéphane Magnenat. Photo of the e-puck mobile robot, 7 October 2008. CC BY-SA 3.0 <`https://creativecommons.org/licenses/by-sa/3.0`>, via Wikimedia Commons; resource link `https://upload.wikimedia.org/wikipedia/commons/4/46/E-puck-mobile-robot-photo.jpg`.

[26] JB Mallinson, S Shirai, SK Acharya, SK Bose, E Galli, and SA Brown. Avalanches and criticality in self-organized nanoscale networks. *Science advances*, 5(11):eaaw8438, 2019.

[27] Robert M May. Simple mathematical models with very complicated dynamics. *The Theory of Chaotic Attractors*, pages 85–93, 2004.

[28] Carver Mead. Neuromorphic electronic systems. *Proceedings of the IEEE*, 78(10):1629–1636, 1990.

[29] Gianluca Milano, Giacomo Pedretti, Matteo Fretto, Luca Boarino, Fabio Benfenati, Daniele Ielmini, Ilia Valov, and Carlo Ricciardi. Brain-inspired structural plasticity through reweighting and rewiring in multi-terminal self-organizing memristive nanowire networks. *Advanced Intelligent Systems*, 2(8):2000096, 2020. Used images are protected by CC BY 4.0 https://creativecommons.org/licenses/by/4.0/legalcode; resource link https://onlinelibrary.wiley.com/doi/full/10.1002/aisy.202000096.

[30] Kohei Nakajima and Ingo Fischer. *Reservoir Computing*. Springer, 2021.

[31] David S Olton. Mazes, maps, and memory. *American psychologist*, 34(7):583, 1979.

[32] Andrea Roli. Lecture notes in intelligent robotic systems, 2021.

[33] Karl Sims. Evolving virtual creatures. In *Proceedings of the 21st annual conference on Computer graphics and interactive techniques*, pages 15–22, 1994.

[34] Marc A Sommer. The role of the thalamus in motor control. *Current opinion in neurobiology*, 13(6):663–670, 2003.

[35] Wikipedia contributors. Jaquet-droz automata — Wikipedia, the free encyclopedia. https://en.wikipedia.org/w/index.php?title=Karakuri_puppet&oldid=1071605485, 2021. [Online; accessed 26-February-2022].

[36] Wikipedia contributors. Karakuri puppet — Wikipedia, the free encyclopedia. https://en.wikipedia.org/w/index.php?title=Karakuri_puppet&oldid=1071605485, 2022. [Online; accessed 26-February-2022].

[37] Tom Ziemke. Are robots embodied? In *First international workshop on epigenetic robotics Modeling Cognitive Development in Robotic Systems*, volume 85, pages 701–746. Citeseer, 2001.