Heriot-Watt University

Masters Thesis

Neurorobotics Models Uncovering Neural Sensory Processing and Muscle Control

Author:
Daria de Tinguy

Supervisor:
Dr. Patricia A. Vargas
Co-Supervisor:
Dr. Renan C. Moioli *

A thesis submitted in fulfilment of the requirements for the degree of MSc.

in the

School of Mathematical and Computer Sciences

and

School of Engineering and Physical Sciences

May 2021

* Federal University of Rio Grande do Norte, Brazil

Owing to the Covid-19 pandemic, the university was closed and access to information required for this project, limited. Please consider this thesis taking into consideration that this paper was written under these circumstances.

Declaration of Authorship

I, Daria DE TINGUY, declare that this thesis titled, 'Neurorobotics Models Uncovering Neural Sensory Processing and Muscle Control' and the work presented in it is my own. I confirm that this work submitted for assessment is my own and is expressed in my own words. Any uses made within it of the works of other authors in any form (e.g., ideas, equations, figures, text, tables, programs) are properly acknowledged at any point of their use. A list of the references employed is included.

Signed: 

Date: 22/04/2020

"Either write something worth reading or do something worth writing."
"Words do not express thoughts very well. They always become a little different immediately after they are expressed, a little distorted, a little foolish."

Benjamin Franklin & Hermann Hesse

Abstract

Parkinson's disease (PD) is the second most common type of dementia after Alzheimer, yet its mechanisms are not well known. PD is characterised by its cognitive and motor symptoms, such as tremor or paucity of movement, and the brain area most affected by this degeneracy is the basal ganglia, an area believed to be implicated in motor behaviour as a motion regulator. Therefore, understanding the brain, and more particularly the basal ganglia, in a PD context, is essential in order to conceive new therapies. There is evidence that PD symptoms emerge from brain-body-environment interactions, therefore experiments to test PD hypotheses should take this trinity into account. Monkeys and rats are commonly used to experiment on PD, with high costs and long studies procedures. Neurorobotics, by embodying computational models to study PD, is seen as an alternative to reproduce this body-environment part of the process. Neurorobotics could allow further understanding of unclear PD mechanisms and isn't yet well present in studying this disease. This work addresses this gap by proposing a humanoid robot, able to reach and grasp, according to the neural activity of a computational model based on rats' motor cortex and basal ganglia. The motion of reach and grasp is generated by a relatively high cortex activity and has been extensively studied in both robotics and neuroscience. Furthermore, it is an everyday action on which symptoms of the disease are easily observable. Therefore, this robot will be able to reach and grasp with and without symptoms according to the neural activity (healthy/PD). This realisation aims to allow researchers to work on reusable models to search for remedies. Hence, animal study could be significantly reduced, thereby, increasing the number of experiments possible and reducing financial costs on long-term studies. This thesis is not a research study but the realisation of a robotic architecture to give physical outputs to an existing computational model.

Acknowledgements

I would like to thank all the researchers, who worked and contributed to the subject matter and shared the result of their respective studies, thereby allowing others to learn too. I wish to thank my supervisors Dr. Patricia A. Vargas and Dr. Renan C. Moioli for their help and supporting me.

Finally, I would like to thank Timothee Freville, Anna Freville, Terry Bateman, Alexandre Lacour, Mael Abgrall, George Corbineau, Simon Lutun for their help in editing my thesis; my parents for all their support during the Covid-19 quarantine, and last, but not least - Lawrence Tesler for his marvellous creation.

Contents

List of Figures

List of Tables

Abbreviations

AIP	**A**ntraparietal area
ANN	**A**rtificial **N**eural **N**etworks
CNS	**C**entral **N**ervous **S**ystem
CTX	Motor **Cor**t**ex**
DBS	**D**eep **B**rain **S**timulation
DH	**D**enavit–**H**artenberg
dMSNs	direct **P**athway **M**edium **S**piny **N**eurons
ECR	**E**xtensor **C**arpi **R**adialis
FCR	**F**lexor **C**arpi **R**adialis
FOV	**F**ield **O**f **V**iew
DOF	**D**egree **O**f **F**reedom
F1	non-human primate primary cortex motor
GPe	**E**xternal segment of the **G**lobus **P**allidus
Gpi	**I**nternal segment of the **G**lobus **P**allidus
ISI	**I**nter **S**pike **I**nterval
iMSNs	indirect **P**atway **M**edium **S**piny **N**eurons
M1	human primary cortex motor
mIP	medial **I**ntraparietal area
MSE	**M**ean **S**quared **E**rror
MSNs	**M**edium **S**piny **N**eurons
MTM	probabilistic **M**ixture of **T**rajectory **M**odels
PD	**P**arkinson's **D**isease
PG	**P**recision **G**rip
PMd	dorsal **P**remotor area
PMv	ventral **P**remotor area

SMA	**S**upplementary **M**otor **A**rea
STD	**St**andard **D**eviation
SNNs	**S**piking **N**eural **N**etworks
STN	**S**ubthalamic **N**ucleus
VL	**V**entral **L**ateral
WHG	**W**hole **H**and **G**rasping

List of Equations

For anyone reading this...

Chapter 1

Introduction

1.1 Motivation

Parkinson's disease (PD) is a neurodegenerative disease affecting 7 to 10 million people in the world, as of 2019. 1% of the population over 60 years of age develop the disease[1, 2]. With the average age of the world population rising, these numbers will increase. It's the second most common type of dementia after Alzheimer's [3]. The key manifestations of PD are cognitive and motor symptoms, such as tremors (currently affecting 70% of patients) [3], slowness or paucity of movements while executing a task[3, 4]. The disease makes everyday motions so much more difficult and there is no stopping it. The disease has no cure, as of now, although therapies exist, such as L-DOPA injection or deep brain stimulation (see section 2.1.3), which helps to palliate the symptoms, but do not stop the progression of the degeneracy [5, 6].

PD deteriorates the dopamine neurotransmitters in the basal ganglia, the region of the brain which is believed to have a key role in the motor loop to control motion (see section 2.1.4.2) [7]. Understanding PD would help developing treatments, yet, PD is not fully explained and neither are the effects of the therapies used. Therefore, many expensive and time consuming studies are done on rats or monkeys. One way to minimise these costs is to create computational models of the disease and test assumptions on a virtual brain. As it is believed that PD is a disease that emerges from an interaction of the brain-body-environment [4], creating a neurorobotics model would make studying the computational models much easier. Modification on the computational brains could be physically observed, reducing the number of experiment to be done on animals and therefore reducing cost and time procedure. The goal of this project is therefore to replicate the animal features of the disease into a neurorobotics model to contribute

to the main aim of gaining further insights into the mechanisms of Parkinson's disease (PD).

In order to observe PD symptoms in a real environment, the motor symptoms are easier to observe than cognitive ones. They can be reproduced in a motion. For example, in the arm and hand. The action of 'reach-and-grasp' is interesting to reproduce on a robot because PD symptoms are well-defined and observable in this motion and because this movement has already been extensively studied in animals and robots. Creating a robot that, is able to express PD symptoms such as resting tremors from a computational model would allow the researchers to work on reusable models to search for remedies. Therefore, animal study could be significantly reduced, thereby, increasing the number of experiments possible and reducing financial costs on long-term studies.

1.2 Project description

My work is part of a wider project-the Neuro4PD project, which aims to develop a Neurorobotics model of Parkinson's Disease. This Neurorobotics Model of Parkinson's Disease is an interdisciplinary project to gain further insights into the mechanisms of Parkinson's disease (PD) by combining the neuroscience expertise of a Brazilian partner with the data mining and robotics excellence of the UK partner. The objective being to contribute to new PD therapies and reduce and/or replace experimentation on live animals. The goal of this project is to replicate the animal features of the disease into a neurorobotics model. Experiments with robots are cheaper and faster than their animal counterparts. My work will be based on a neurorobotics model currently being developed by another team on the project, I will be focusing on how complex motor control can be accomplished in realistic computational neural models embedded in simulated robots. In particular, I will explore how to translate core neural mechanisms into motor signals.

1.3 Objectives

The objective of this project is to embody the implication of the basal ganglia in PD's symptoms while effectuating a specific action (see Section 2.1.4.2 of the literature review for an explanation on the basal ganglia). The objective of my work is to contribute to understanding how a robot motor control can receive inputs from a computational model (reproducing healthy and PD Basal ganglia and cortex) to execute a reach and grasp motion. The scope of the research is to enable a robot to effectuate a reach and grasp motion while incorporating the symptoms of the disease and external sensory stimulus.

1.4 Professional/legal/ethical and social issues

According to the British Computer Society's code of conduct 1, as a future engineer, I have to adhere and am proud of adhering to the following guidelines :

• Public interest: my work shall not interfere with public health, privacy and security, I shall respect the right of third parties and the project shall be conducted without discrimination to ensure equal access to the benefits of IT.

• Professional competence and integrity: I shall not claim any level of competence that I do not have, and I shall only undertake the work within my range of competences. As I do not have all the competences required to do this thesis, I shall work to acquire them.

• Duty to relevant authority (Heriot-Watt University): If some aspects of my work are to be kept undisclosed, I shall not disclose them to any unauthorised person(s) or organisation(s).

• Duty to the profession: I shall act with integrity and respect in my relationships and dealings with my colleagues, and I shall help and encourage them whenever I can do so.

The main legal, Ethical and issues I will face during the project are:

• Intellectual property: Any use of the work of a peer should be referenced and all the work done during this project is the property of Heriot-Watt University.

• Intellectual property 2: I should not publish any work of a peer without his consent.

• All tools used to implement the system are open source

• This work follows the 3Rs guideline and especially aims to the Replacement and Reduction of animal use. Any animal experiment realised by the Neuro4PD project follows this guideline and respect the legislation.

• There is no ethical issues in this particular study as no human or animal are directly implicated in its completion.

• The robot may realise unexpected motions, a NAO robot is used, therefore even if the motion is sudden, its use should not be dangerous to the user, according to Softbank NAO's conception.

• There is no social issue. This robot has not for aim to have any interaction with the population or researchers, it is there to express physically the comportment of a computational model.

Chapter 2

Literature review

This Chapter will explain the different concepts on which this study will be based, as understanding the subject is necessary in order to study it. First of all, the characteristic of PD, its symptoms and the therapies used to palliate PD manifestations will be presented. Next, the basal ganglia and dopamine neurotransmitters are going to be described, as well as the motor loops present in the basal ganglia and how they are disrupted by the lack of dopamine. Then, how computational models try to artificially reproduce the mechanisms of Parkinson's will be explained. Leading subsequently to neurorobotics- what is it, what are the solutions to control motor outputs and an explanation of the sensory motor loop coding/decoding process. Finally for this study Nao, ROS, neuron NetPyNe and the simulation bench webots are, also, going to be briefly explained.

2.1 Parkinson's disease

2.1.1 Introduction

Parkinson's disease (PD) was first described by James Parkinson in 1817 as a neurodegenerative disease (i.e. a progressive loss of structure -or function- of neurons) in the central nervous system (CNS) and more particularly the motor system. It is the second most common neurodegenerative disease in the world after Alzheimer's [3]. It is estimated that there are 7 to 10 million people (in 2019) who are diagnosed Parkinsonian, worldwide. With the median age of the world's population rising, the probability of getting the disease is of 1% when over 60 years old [1, 2, 5]. The life expectancy of a PD patient is about 10 to 20 years after being diagnosed. There is no cure for this disease, only treatments alleviating the symptoms temporarily.

Lifestyle changes such as regular exercise and a healthy diet can contribute to longevity of PD patients. This degenerative disease affects only humans, and men are 1.5 times more likely to get Parkinson's than women, with the reasons not known [1, 2, 4].

The exact causes of PD are still being studied. up to 15% of patients have a family history with Parkinson [2]. However, genetic factors do not fully explain the cause of Parkinson. Age, lifestyle, environment factors are also at play [4]. Imbrication between brain, body and environment should be continually studied to try and understand where Parkinson's Disease is coming from. As an increasing number of people are going to be affected by PD in the next few decades, it has become imperative to better understand the disease to find reliable remedies to this degeneracy. Hence, multiple models, as the one this study relies on, are being developed and tested to explain the human brain's processes in order to develop better treatments and a cure.

2.1.2 Symptoms

PD progresses relatively slowly and the symptoms grow from mild to severe as time goes by. There are many symptoms associated with the Parkinson's disease, however, each person may develop a different set of symptoms with varying severity [3]. Early symptoms are a reduction of olfactory functions (sense of smell), the appearance of rapid-eye movements etc.), before the manifesting of motor symptoms. Then the main signs of this disease are cognitive and motor behaviours. The most common and visible ones being motor symptoms as [3, 4, 8]:

- Tremor (i.e. the shaking of the limbs. This symptom usually begins in a relaxed limb such as an arm or a hand, and is developed by 70% of the patients. A surprising feature of tremor in Parkinson is that it appears in a relaxed state and it may disappear during a voluntary movement.
- Bradykinesia (slowness of movements)
- Rigidity (muscle stiffness, cramps)
- Akinesia (freezing of movement during the execution of one)
- Postural instability (difficulty in maintaining balance) which emerges rather late in the disease's progression.

All the above mentioned cadre of PD symptoms are caused by a low-level of dopamine in the brain, which particularly affects the basal ganglia. Therefore studying these areas is relevant to understanding the disease and developing a treatment.

2.1.3 Current treatments

As far as I'm aware, there is no widely available therapy, today, that can stop the degeneration of the neurons, rather available therapies alleviate the symptoms. Neither the mechanisms that induce Parkinson's nor the mechanisms of the treatments are fully understood yet. The most-used strategy is to enhance the levels of dopamine delivered to the basal ganglia either by boosting the level of dopamine in the brain or by mimicking the effects of dopamine.

The most common pharmaceutical therapy is the L-DOPA administration. The dopa dopamine (L-dihydroxyphenylalanine or levodopa) is a precursor to dopamine. It boosts dopamine synthesis in the cells, which are still active in the substantia nigra, alleviating some of the symptoms [5, 7]. However, this does not stop the disease from continuing to degenerate the substantia nigra neurons (see section 2.1.4.2 for an explanation on the substantia nigra) at its normal pace. Furthermore, injecting dopamine replacement may have no effect on some symptoms [2] and could provoke a dependency, as a side effect, in the long run. Dopamine-dependent cognitive symptoms include deficits in attention, processing speed or affected verbal fluency [9]. Typically, after 5 years of medical therapy, complications develop in most patients [6].

The second most-used pharmaceutical treatment is the use of dopamine agonists, which are often less efficient in diminishing motor symptoms and it is mostly used to retard the L-DOPA usage.

Once pharmaceutic therapies begin to fail to cope with the increase of motor symptoms, surgery is recommended. In the past, ablative surgeries on the thalamus or on the globus pallidus were widely-used but due to the emergence of deep brain stimulation (DBS) (which does not destroy brain tissue like a lesional surgery) this operation is not as common as before [2].

In a DBS approach, an electrode is implanted in the globus pallidus or subthalamic nuclei, and then an electrical current is delivered to the structure [10, 11]. DBS main contribution is to change the firing rate and firing pattern of neurons in the basal ganglia [6]. As it is based on electric stimulation, it has an impact on a wider range of the brain than just the affected basal ganglia. The influence of this electrode on the brain as a whole is unclear, however the benefits of using this therapy in alleviating symptoms of Parkinson has been established empirically thanks to the use of DBS in others medical areas.

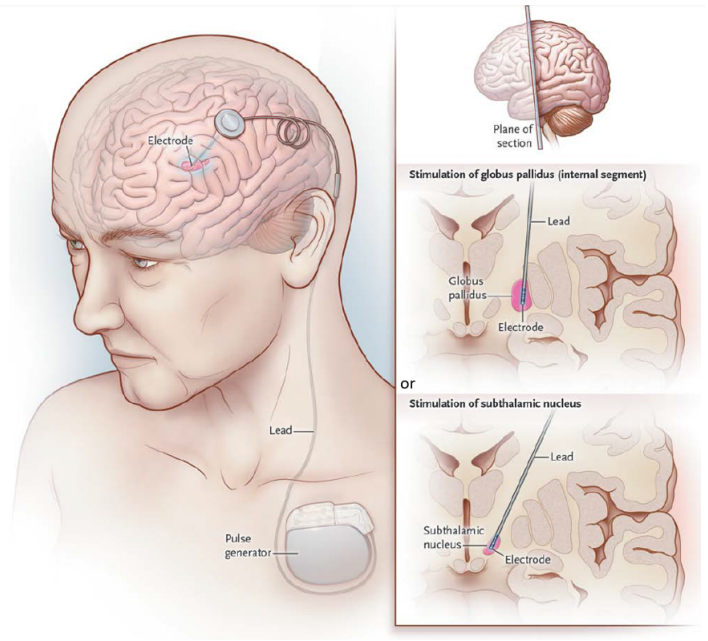


FIGURE 2.1: Electrode Implantation in Deep-Brain Stimulation [12].

Unfortunately, DBS is believed to induce mild depression, among other rare side effects, in some of the operated patients and does not stop PD from progressing. Further, over a course of time (that varies from patient to patient) DBS appears to lose its efficiency[6, 11].

Another element that may help alleviate PD's symptoms, and improve a PD patient's body control and well-being, is regular exercise and a balanced diet. And, other studies say that physiotherapy, kinesitherapy are also relevant treatments to alleviate some of the motor symptoms [4, 13].

There are also some therapies in experimental stage, such as the grafting of non-neural cells, genetically manipulated to produce dopamine into the basal ganglia or grafting human stem cells that have been induced to release dopamine. However, these approaches are tempered by the risk of carcinogenesis (cancer development), the difficulty in executing of such a therapy and to obtain sufficient stem cells [7].

Presently, the mechanism behind Parkinson's development, its symptoms and therapies are yet to be clearly defined. it is not known if motor and non-motor symptoms are shared or are distinct circuit mechanisms or even how major therapies employed for PD work. Hence, these drawbacks hinder the development of new treatments or stopping the disease from degenerating. Understanding the human brain's process in PD is a must to improve treatments.

2.1.4 Basal Ganglia Thalamus Cortex complex

Almost, if not all, of the parkinsonian signs seem to arise from an abnormal basal ganglia activity (see section 2.1.4.2). Therefore, to be able to study the disease and develop a solution, understanding the Basal Ganglia Thalamus Cortex complex is primordial. In PD, it is possible to note a deficit of dopamine with a cell death of dopaminergic neurons in the basal ganglia, and, more specifically, in the substantia nigra. The motor loop is affected, resulting in motor symptoms (as described previously in section 2.1.2). The executive, visuomotor, and motivational loops are affected generating severe acute symptoms to the point where living a normal everyday life for a PD patient becomes impossible.

2.1.4.1 Dopamine

The dopamine is a neurotransmitter released from axon terminals in response to a nervous impulse. The neural impulse being an electrochemical signal travelling along the axon. Dopamine receptors control the excitability of the neurons. In the part of the brain this thesis is interested in, the most common receptors are of D1 and D2 types [14].

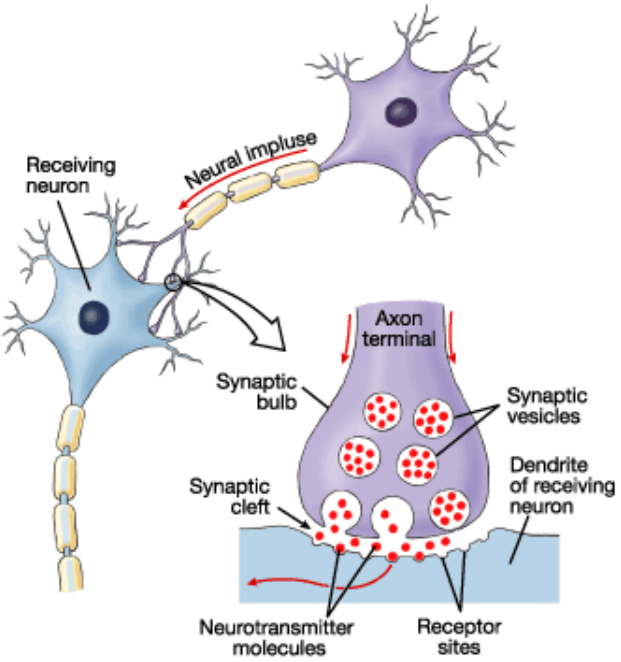


FIGURE 2.2: Representation of a neurotransmitter in a neuron [14].

In the case of Parkinson, the nervous impulses do not release dopamine in the affected terminals, which leads to a lack of transmission of the signal from one neuron to the

next. The exact mechanism of this inhibition is not fully known but some studies try to model the basal ganglia to determine the causes.

2.1.4.2 Basal Ganglia

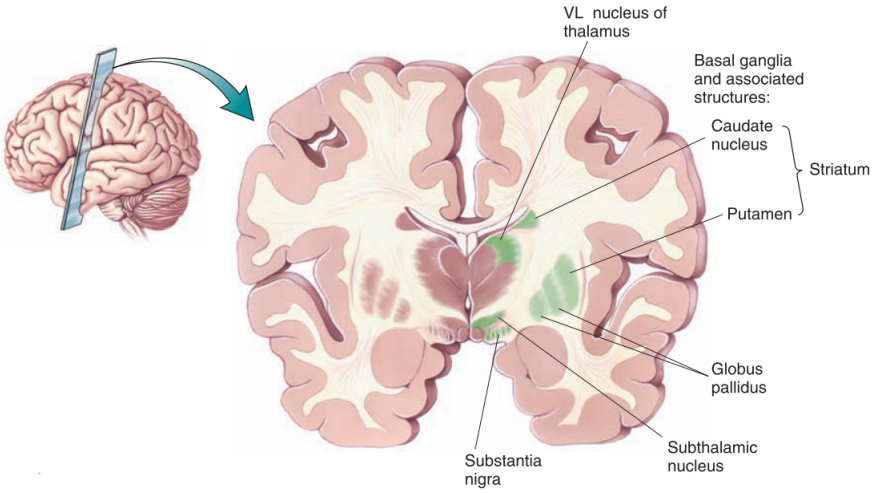


FIGURE 2.3: Presentation of the basal ganglia (or basal nuclei) and associated structures [7].

The basal ganglia is often associated with the control of voluntary and subconscious motor movements (selection and initiation), the procedural learning and habit learning (adjustment to new/unexpected situations and sequential movements requiring an adaptation), eye movements, cognition and emotions [10, 15–17]. Studies show that it is most active during the planning phase of a movement. Therefore, if the basal nuclei is damaged it is possible to observe a disruption of movement and/or significant deficits in other neural functions, such as cognition and perception [10].
It's in this structure the cell death of dopamine terminals, which leads to the degeneration of the substantia nigra inputs to the striatum and especially the putamen.
The striatum controls the motor, cognitive, and reward system and serves as the primary input source to the rest of the basal ganglia.
The Putamen's primary function is to regulate movements and influence learning. It contains the densest expression of dopamine receptors in the brain, consequently it is particularly dependant on this substance to function normally [7, 18].
The basal ganglia treats the signals coming from the cerebral cortex and, after a loop, send them back to the cortex and then to the spinal cord,.

Direct Pathway

the direct motor loop, or "direct pathways" can be defined as being:
Cortex (+) → Striatum (Putamen) (-) → internal segment of the Globus pallidus (GPi) (-) → Thalamus (ventral lateral (VL) nucleus) (+) → Cortex supplementary motor area (SMA)

(+): excitatory synapses
(-): inhibitory synapses

The Cortex is the largest site of neurons in the Central Nervous System (CNS) (influence the whole-body activity). It plays a role in attention, perception, awareness, memory, language, and consciousness[7, 19]. From the cortex, the striatum (and more particularly the putamen) receives excitatory signals which stimulates the inhibitory synapses of the putamen cells, inhibiting, in turn, the Globus pallidus.

The Globus Pallidus, as a whole, is also a major component of the basal ganglia- it receives inputs from the striatum and sends outputs mainly to the thalamus and substantia nigra. It regulates voluntary movements. Its primarily role being inhibitory, it balances the excitatory action of the cerebellum. It is interesting to note, that a lesion in this part of the brain causes tremor, which hypothetically is caused by an imbalance between the signals of this structure and the signals coming from the cerebellum. The GPi, in the direct motor loop, once stimulated makes inhibitory connection with the VL cells directly, as can be seen in Figure 2.4 [15].

Then the VL receives this motor information and sends excitatory signals to diverse structures such as the SMA and Cortex motor. Its function being to help the coordination and planning of movement. It also contributes in the learning of movement.

Finally, the SMA projects the motor signals directly to the spinal cord. This part of the brain is believed to manage the postural stabilisation and coordination between both sides of the body [7].

It is speculated that the signal to start a movement is initiated when activation of the SMA is boosted beyond a threshold, sending a strong enough signal to pass through the basal ganglia "funnel".

The depletion of dopamine in PD closes this funnel that feeds activity to the SMA via the basal ganglia. An increased inhibition of the thalamus in the basal ganglia is often associated with PD symptoms of tremor or paucity of movement.

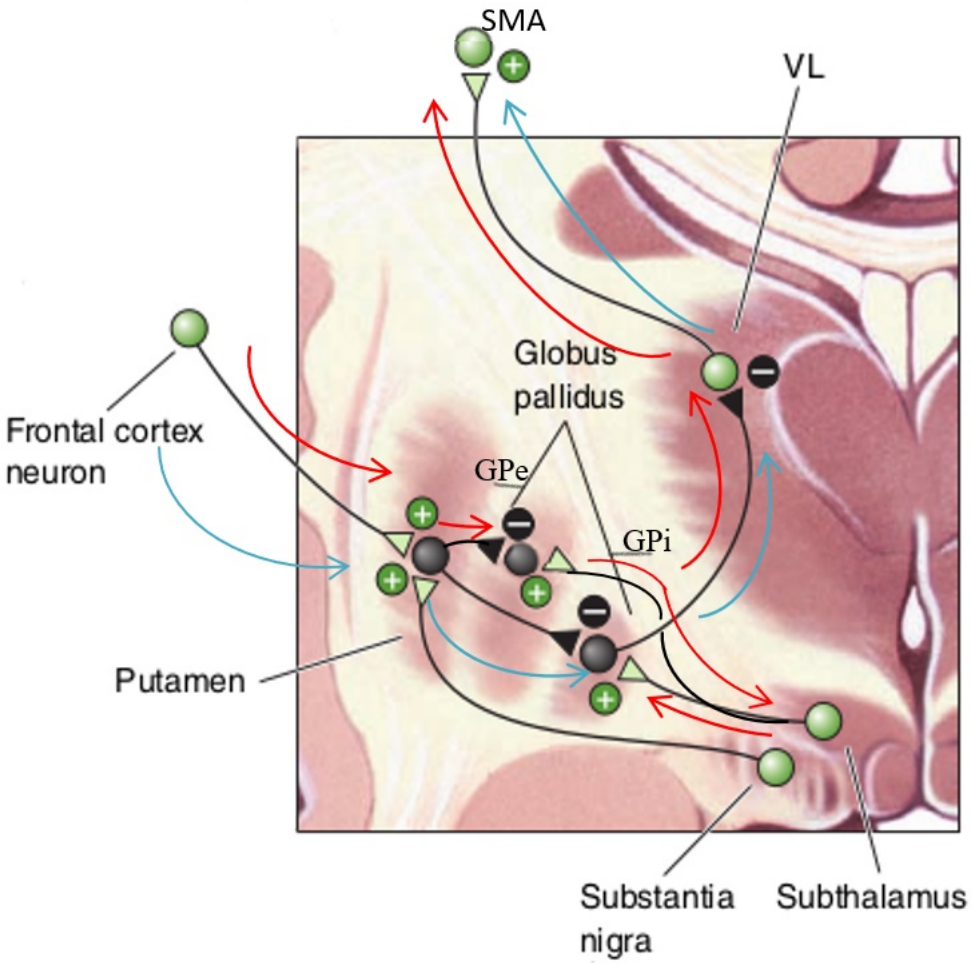


FIGURE 2.4: A wiring diagram of the basal ganglia direct/indirect motor loops. Synapses marked with a plus (+) are excitatory; those with a minus (-) are inhibitory. The blue arrows are the direct loop path, the red arrows the indirect loop path ([7],modified).

Indirect Pathway

Besides the direct pathway, another motor loop goes through the lateral cerebellum, which is also called side loop or indirect pathway. The indirect motor loop can be outlined as:

Cortex (+) → Striatum (Putamen) (-) → external segment of the Globus pallidus (GPe) (-) → Subthalamic nucleus (STN) (+) → internal segment of the Globus pallidus (GPi) (-) → Thalamus (ventral lateral (VL) nucleus) (+) → Cortex supplementary motor area (SMA)

(+): excitatory synapses
(-): inhibitory synapses

The External Globus pallidus is peculiar as it does not send signals out of the basal ganglia. It seems to serve more as the main regulator of the basal ganglia system and can modulate the effects of the direct pathway allowing finely-tuned basal ganglia outputs.

The STN is a part of the subthalamus whose function is unknown. It is supposed to be a component of the control system of the basal ganglia that performs action selection and impulse control (hold muscular responses in check). Most of the time, subthalamic cells are inactive due to the constant inhibition of the GPe. If the basal ganglia was to decrease its activity, as is the case in Parkinson, it would activate the subthalamic cells, which would, in the end, result in a motor deficit [10].

Those pathways are thought to be critical to the execution of planned, voluntary, multi-joint movements. Once the intent to execute a movement has been received by the cerebellum, this structure seems to instruct the primary motor cortex with movement direction, timing, and information on the force to be applied[7].

The first panel (1) of Figure 2.5 outlays all the connections in the basal ganglia, the second (2) highlights the direct and indirect pathway activity and motor output in a healthy human basal ganglia, while the third (3) shows what is happening in a parkinsonian brain (3) [5].

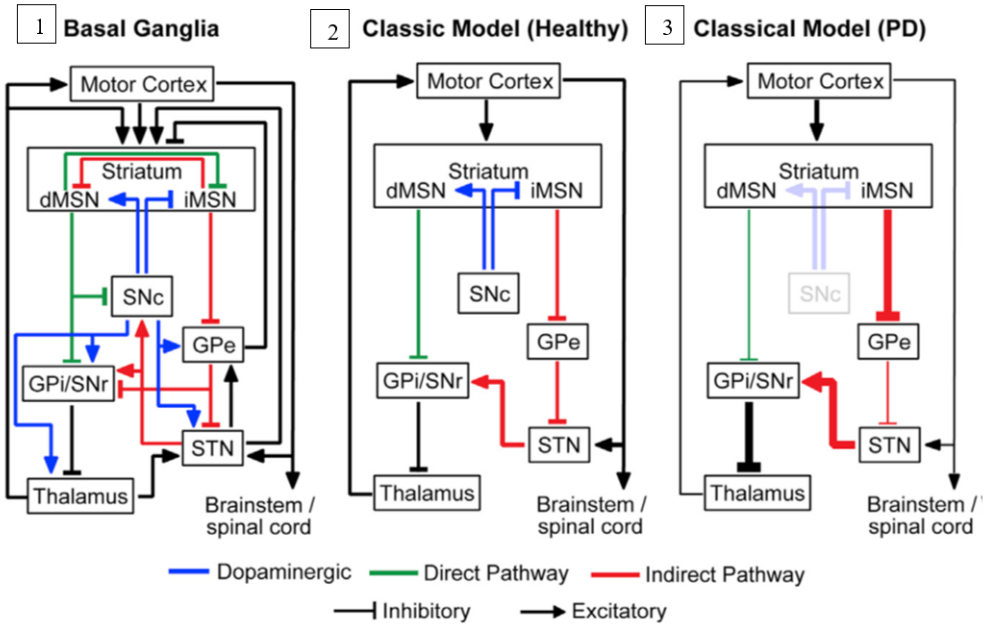


FIGURE 2.5: Classical Models of the Basal Ganglia [5] .

The Dopaminergic transmission, depicted in blue in the first and second panels of Figure 2.5, shows the involvement of dopamine as a neurotransmitter between channels.

The SNc and SNr are parts of the substantia nigra. dMSN and iMSN are two types of Medium spiny neurons (MSNs) (also known as spiny projection neurons or SPNs). They represent 95% of neurons in the striatum and are inhibitory cells. They are composed of two main phenotype: D1 or D2 types of dopamine receptors. dMSN stands for direct pathways medium spiny neurons (composed of D1 type MSN), they excite output structures of the basal ganglia, while iMSN stands for indirect pathways medium spiny neurons (composed of D2 type MSN), they inhibit output structures. D1 is believed to allow the completion of movement while D2 is there to restrain it. A right balance between those two MSNs allows action selection as seen in a healthy individual [5].

According to the classical model, in healthy basal ganglia (panel 2 of figure 2.5) the dopamine circuit activates the direct pathway (green line) and inhibits the indirect pathway (red lines in Figure 2.5) MSNs. It decreases the GPi outputs, inhibiting the thalamus and cortex activity thus promoting movement. In contrast, in the PD model (3), the loss of SNc dopamine causes hypoactivity of the direct pathway, a hyperactivity of the indirect pathway and hyperdirectpathway (STN projections) leading to an excessive GPi output. As a result, the thalamus and cortex are highly inhibited and this leads to a suppression of movement [5].

This phenomenon may partially account for the early development of motor deficits symptoms as bradykinesia [5]. Bradykinesia is the reduction in velocity and amplitude of movements, and there is evidence correlating this symptom to the imbalance between the outflow of the direct and indirect pathways leading to an increase in the activation of the antagonist muscles instead of the intended one. Thus, the lack of movement could be due to an inappropriate activation of the antagonistic muscles instead of a decrease in muscle activity [10].

The basal ganglia channels, how they are affected by Parkinson, and whether their change could be the cause of modification in the cognitive and behavioural symptoms, remains an area in need of further investigation. Most of what has been described above are largely accepted models, which could further evolve as studies and discoveries of new functions and imbrication in the basal ganglia are conducted. In the case of this study, the basal ganglia and motor cortex activity are both essential for a voluntary motion as reach and grasp, as it will be developed in future sections.

2.1.4.3 Motor Cortex

Previous studies show that the control of most voluntary movements involves all the neocortex. The neocortex has to take charge and consider:

- The actual position of the limbs,
- The desired limbs position,
- The formulation and remembering of a plan,
- The transmission to the muscles of the plan's instructions.

Among the anatomical regions involved in motor functions of the body, the motor cortex (CTX) and the basal ganglia are fundamental.
The primary role of the CTX is to generate neural impulses to control the execution of movements in a contralateral fashion. The primary motor cortex (M1) is often shown in a body map (called the motor homunculus) featuring the cortical space required to execute control on different body parts.
The SMA, which is part of the cortex, is responsible for transforming visual information into motor commands and is closely linked to the basal ganglia activity [20, 21].
PD disrupts the motor loop from the SMA to the basal ganglia, which sends the signals back to M1 and SMA, inducing the symptoms presented earlier. A perfect movement to study these disruptions would be an everyday action, such as the reaching and grasping motion. To compare how the basal ganglia, and M1, affect the motion in a healthy

individual vs in a PD patient would allow us to study the involvement of the basal ganglia in inducing these symptoms and to try out therapies to palliate them. To be able to study the motion of reaching and grasping, it is important to understand how it works in an healthy brain.

2.1.5 Motor control of reaching and grasping

A reaching and grasping movement needs the brain to have first evaluated the distance, the shape, weight, and size of the object and then determine the grip strength necessary to grasp it. It requires the ability to process and coordinate multiple configurations of arm and finger movements depending on the object analysed.

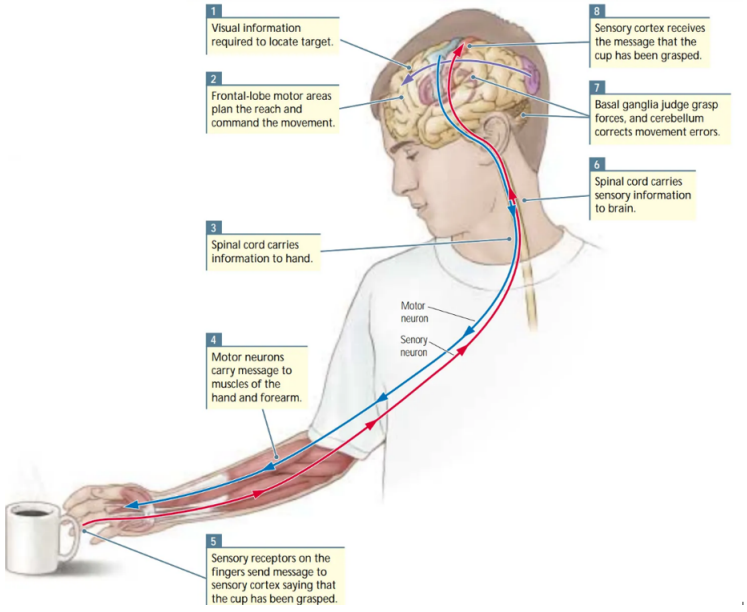


FIGURE 2.6: Description of the steps and assumed brain regions involvement in the action of reaching and grasping [22].

The Figure 2.7 shows the implications of diverse areas of the brain in a motion movement. This study will mainly focus on the "actor critic", "premotor" and "motor" areas (and briefly implicate the "parietal" section) described in the image.

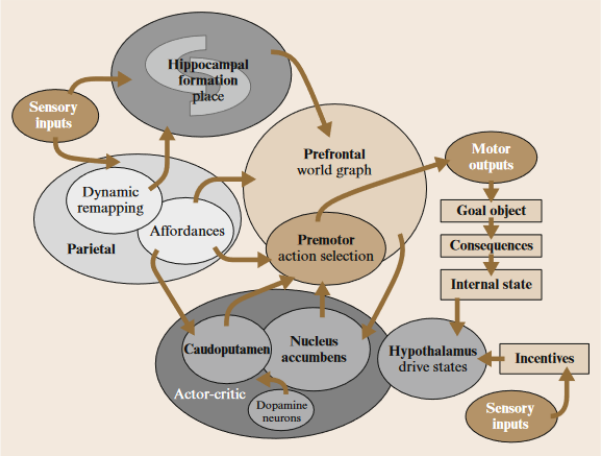


FIGURE 2.7: Primate cognitive map presenting large-scale interaction between areas of the brain. It can be observed how the brain is largely involved in a motion [23].

It is difficult to really pin-point which areas are most involved in reaching and grasping movement, as the action seems to be coded by different neural systems [24]. The activity occurring in diverse regions of the brain differs, depending on the size of the object to be picked up - for instance, the precision grip (PG) (for little objects) provokes a significant activity in the anterior intraparietal (AIP) region of the brain while a whole hand grasping (WHG) (for bigger objects) do not. Hence, this section will first try to fully understand the brain activity involved in each action and then simplify the system to outline the most important features. This step is essential in order to understand which areas of a model are the most important for this thesis and from where to extract the most significant neural activity.
PG involves small objects and a small hand aperture, so the thumb and the index are the major actors of this motion and the primary motor cortex is active. While WHG, used for grasping bigger objects, requires less accuracy in the movement. Here, the fingers and the palm are the main actors of this motion and do not trigger as much brain activity as PG in the primary motor cortex [21, 24]. In the cortex, multiple studies seem to implicate the AIP, M1 (or F1 depending on the animal studied), ventral premotor (PMv), and dorsal premotor (PMd) in both reaching and grasping operations [24–27].

AIP helps in determining the shape, size, direction and distance to an object to be grasped. It could be described as the brain area receiving the visual inputs and extracting 3D information about the object seen.

If visually triggered, a movement induces a strong response in PMv and PMd, which are involved in dealing with the selection and guidance of movement based on visual information. M1 and PM (PMv and PMd) are related to the execution and preparation of reaching movements and visuomotor learning, while SMA (and the basal ganglia)

are more active during the performance of pre-learned sequences. The basal ganglia is strongly active during a precision grip or any task where you have to adapt to a new situation, the basal ganglia is supposed to have a corrective function.

In a healthy non-human primate, a simplified presentation of a reaching and grasping action could be seen as explained below [16, 25–30].

From a cortex point of view, excluding the basal ganglia involvement, the visual cortex (V3a) information is processed by V6a for reaching and AIP for grasping and then sent to different parts of the premotor area.

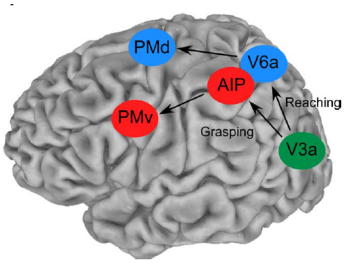


FIGURE 2.8: The classic reaching and grasping network showing cortical regions involved in grasping and reaching. The dorsomedial circuit (blue) connects area V3A in the visual cortex, area V6A in the parietal-occipital sulcus, and the dorsal premotor cortex (PMd). The dorsolateral circuit (red) connects area V3A, the anterior intraparietal area (AIP), and the ventral premotor cortex (PMv) [29].

Considering the basal ganglia involvement in action generation this is what can be added:

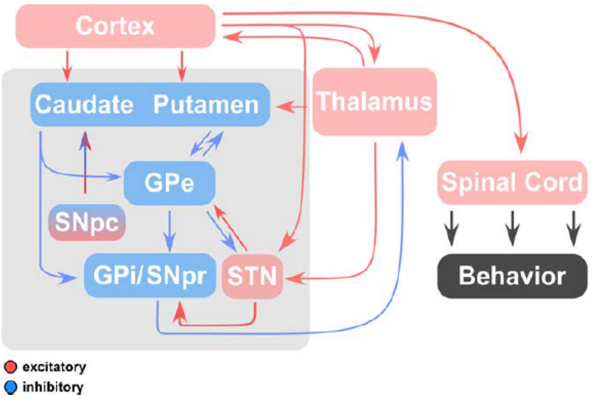
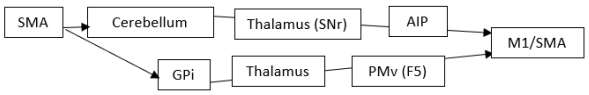


FIGURE 2.9: Interactions between basal ganglia regions, the motor cortex and the muscles. Excitatory connections are red while inhibitory connections are blue [29].

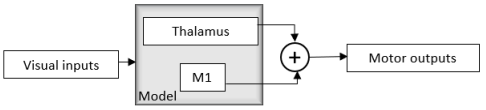
The basal ganglia's direct and indirect pathways are stimulated by willed movements. And some studies have gone even further in trying to analyse the precise involvement

of the basal ganglia to control the grip strength in non-human primate [29]. An overly simplified path could be summarised as below:



The work of J. Prodoehl & all (2009) [29] inscribes his review on the main stream model presented in Figure 2.5 adding an higher scale brain interaction.This paper reviews the studies done on grasping and how the different brain regions are believed to be involved in the process of precision grip force.

Note that the SMA receives less inputs than the M1 overall, the SMA is particularly active while executing unusual actions [31]. Out of those direct/indirect paths a really simplified comportment could be extracted and applied to the model used in this thesis by extracting outputs from the Thalamus and Cortex to define the final information to be sent to the motor control loop: It is important to note that none of these pathways



mentioned above encompass the integrity of the coding of the reaching and grasping action.

Parkinsonian reach and grasp motion

U. Castiello & al (1999) [32] compared the adaptive response of the reach-to-grasp movement of Parkinson's patients vs healthy subjects with regards to a change of size and position of objects. Well defined differences were observed between both groups in all the experiments realised in this study, describing manifestation of PD in adaptive behaviour in reach-and-grasp motion. Since [32] presents the results of a well defined experiment, its findings are serious and the comprehension of the subject matter mentioned below is heavily based on their observations.

It has been observed in Parkinson patients that each movement executed is made of sequential responses and is significantly slower than in normal individuals. As reach and grasp actions require the involvement of many of the brain's regions and a complex exchange between them, it is easy to disturb them. Several observations have been made [29, 32]:

- Firstly, it seems the size of the object does not induce a major difference in PD patients reaching speed.

- Secondly, individuals with PD seem to produce higher peak grip forces when the load of the object is unknown.

- Thirdly, the end of the movement is particularly slowed down (the moment in which precision is most required in term of reaching speed -deceleration- and hand aperture).

- Then a loss of independent finger movement in PD patients is notable while a combined movement of using all fingers at once remains normal.

- Finally, the movements executed in the end by PD patients and normal subjects are the same. Therefore, the ability to execute the movement itself, independent of the duration, does not seem to be affected.

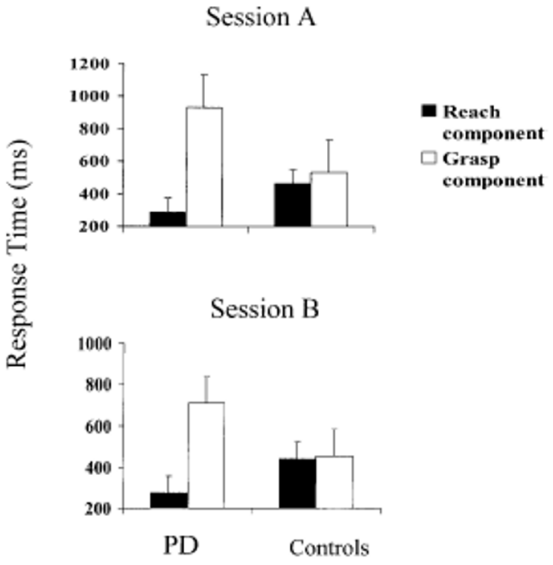


FIGURE 2.10: Average time of reaching and grasping in PD patients and normal patients. Session A is first reaching and grasping small objects then a larger one while Session B aims first at handling a large object then a smaller one. "Reach component" measures the time to reach the peak velocity and the "Grasp component" measures the time to reach the maximum grip aperture. The max reaching speed is attained much faster in PD patients than healthy individuals [32].

In Figure 2.10 (Session A corresponds to reaching and grasping a small object then larger one, while Session B requires the reverse) It can be observed that PD subjects passing from a WHG to a PG requires less time and effort than passing from a PG to a WHG. It is hypothesised that the visuomotor processing, the planning stage of the action, in which the basal ganglia contributes, is being activated later in PD than in normal subjects and could be a cause for this disruption [32].

Some of the studies referenced here were done on humans, others on non-human primates. It is important to note that, here, it is being considered that the cortex of the two species are similar enough to be able study on the monkey and conclude for the human. However they are not believed to be completely identical and the limit of their similarity has yet to be clearly defined.

As Parkinson motor dysfunctions are the most visible phenomenon, reaching to grasp an object provides an ideal experimental behaviour for analysing the symptoms. As it is an everyday action, wherein it can easily be observed that the effects of PD it is interesting to try and model it to understand how acting on the model could impact behaviours. And to do so, there is computational models.

2.1.6 Computational models

Over the years a lot of biological data associated with the basal ganglia has been collected. Interpreting them has become essential to understanding PD. Hence models, and computational models especially, are one of the best ways to test hypothesis on its complex behaviour. To study the mechanisms of the brain in a healthy person, in a Parkinson patient, or to study the treatments, computational models are a way to verify assumptions and anticipate results.

There are two main ways to create models: top-down or bottom-up. Top-down models are created based on hypothesis to verify with the data already accumulated (or will be) and bottom-up models are based on integrating this accumulated data to search a viable model and then test ideas. The former requires a solid hypothesis while the latter requires a significant quantity of data to resemble biological systems [33].

One of the biggest challenges of creating a computational model is to mimic a biological response while at the same time simplifying its process and not omitting important features. If a team was to reproduce each neuron fully, the computation may become very slow, or difficult to analyse (too many details). In most models Spiking Neural networks (SNNs) are used as they are among the artificial neural networks closest to a natural one. They integrate the concept of time and reproduce the behaviour of a neuron and synapse. The "firing" is binary (0-not excited enough, 1-threshold reached) and temporal (the excitation decreases over time) as a real neuron. However the SNNs are hard to integrate and train into systems and aren't really biologically accurate as it is said that a spiking neuron in a computational model can replace hundred of neurons in a real biological process [34], losing complexity in the operation. Still, traditionally it is thought that most of the relevant information can be read in the firing rate of neurons, focusing more on a global output than in individual neurons firing activity. A firing rate

is a temporal average of neurons. There are several procedures to extract the firing rate [35]:

- Average over time (rate as a Spike count, most used)

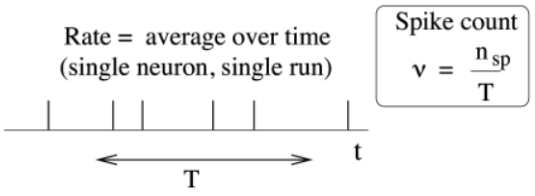


FIGURE 2.11: The spike count measure: Definition of the mean firing rate by temporal average.The firing rate "v" is the spike count "nsp" in an interval of duration "T" divided by this "T" [35].

- Average over several repetitions of the experiment (rate as a Spike Density)

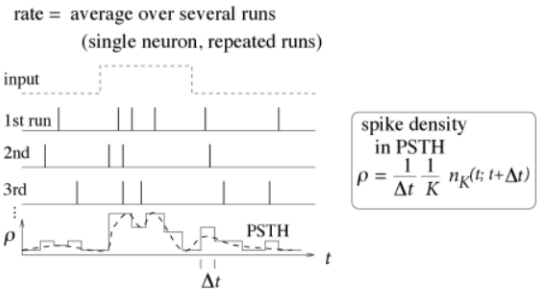


FIGURE 2.12: The Stimulus Time Histogram and the time-dependent firing rate as an average over several runs of the experiment.The time "t" is measured with respect to the start of the stimulation sequence and "Δt" defines the time bin for generating the histogram, it is typically on the order of milliseconds [35].

- Average over a population of neurons (rate as a Population Activity, reassemble neurons with similar properties -same pattern of inputs/outputs-).

The firing rate is often used in models as it has been shown by Adrian (in 1926-1928) that the firing rate of the stretch receptor neurons in the muscle is intimately linked to the force applied to the muscle [34]. So it became a tool to describe sensory and cortical neurons properties, even though some information may be neglected (e.g. exact firing timing). Whilst the use of only firing rate is debated as being too simple to fully explain mechanisms, it is still a powerful tool, which is easy to integrate and gives good results in understanding parts of the cortical neurons comportment.

It is important to note that all models intentionally omit details and simplify behaviours in order to identify causes and effects. The goal of these computational models is not

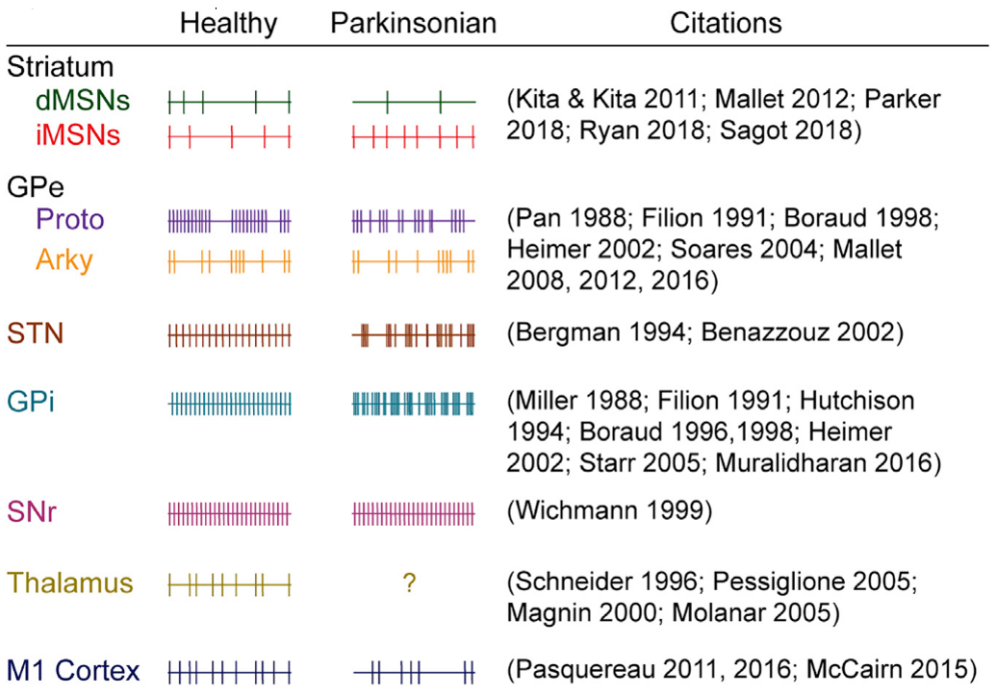


FIGURE 2.13: Changes in Firing between healthy and Parkinson conditions schematic depiction of pattern and rate changes observed across healthy and Parkinson non-human primates in the basal ganglia, thalamus, and cortex [5].

to replicate the brain identically but to test assumptions on simplified model versions of the brain, and hence verify if the hypothesis makes sense. If they do, then they can be tested on a more complex model.

For this study, the models implicated are bottom-up models as they try to mimic real behaviours of healthy as well as Parkinson's affected rats [36] and monkeys [37] as close as possible.

Kumaravelu & al (2016) [36] developed a computational model able to represent the Parkinson state in 6-OHDA (molecule used to destroy dopaminergic neurons) lesioned rats and investigate some therapeutic mechanisms. Diverse parts of the brain were simulated. Notably the GPi and GPe of the basal ganglia and the motor cortex (CTX), and outputs as firing patterns, firing rates and neural synchronisation across different basal ganglia structures are extractible to use for modelling behaviour.

[36] work based itself on a heavy documentation and allowed a good repeatability of its finding, exposing clearly most of the parameters used to conceive this rat model. This computational model was notably reproduced and used in the Neuro4PD project, this thesis computational model will be based on the liability of their research.

Their model demonstrated at least two states i.e. representing a normal brain and a lesioned (PD) brain. To represent a PD brain which is the result of loss of striatal dopamine neurons they made three changes to the normal state they simulated:

- A reduction of M-type potassium current in the direct and indirect pathways MSNs. Which means a disturbed communication between MSNs.
- A reduced sensitivity of direct MSN to cortical stimulation.
- An increase of aberrant GPe firing (due to GPe being overstimulated).

The result shown in the figure 2.21 shows how much this model fits experimental results in mean firing rates.

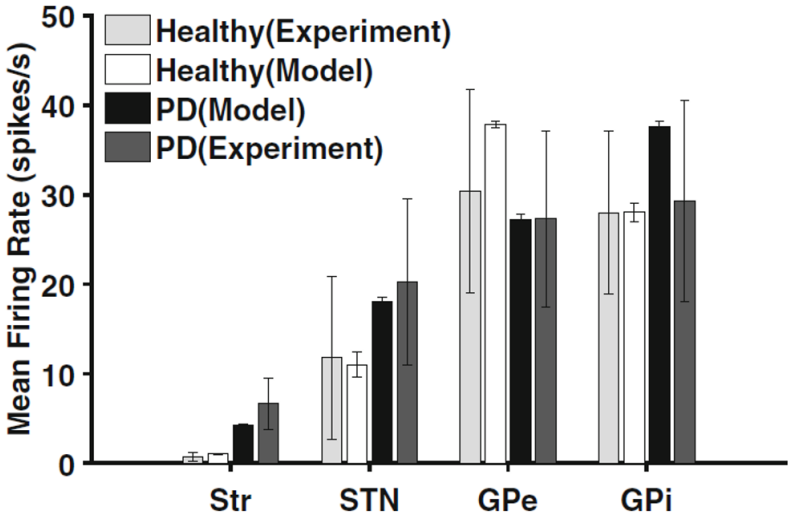


FIGURE 2.14: Firing rates of model and experimental neurons in striatum (Str), subthalamic nucleus (STN), globus pallidus externa (GPe) and globus pallidus interna (GPi) under normal and PD conditions. Standard error bars for model data are shown for 10 second simulations under each condition [36].

It is important to note that the above model was based on rats, whose firing rate of Basal Ganglia neurons differ from non-human primates. In fact, in rats, the firing rates of the GPi, GPe and STN are much lower in both healthy and PD conditions.

By extracting firing rates from the modelled brain parts involved in a motion it should be possible to reproduce a movement with healthy and PD symptoms. Which is what this thesis intend to realise. The computational model is the "brain" part of the brain-body-environment relationship. Therefore having a robot reacts to neural stimulus is the next step to study computational models interaction with the body and its environment.

2.2 Neurorobotics

2.2.1 Introduction

Neurorobotics is at the intersection of neuroscience, robotics, and artificial intelligence. It embodies brain-inspired algorithms and computational models, which enable the creation of testing platforms for motor control and locomotion, reward systems and action selections, and importantly used to study medical cases such as Parkinson's disease.

The term "neurorobotics" is relatively young but the pioneers of this field of science are estimated to have emerged during the end of the '40s, with Grey Walter's Tortoises experiment of a robot moving towards a light by adapting its trajectory [38, 39]. It was a simple robot, mimicking a seemingly natural behaviour. With light detectors and a bumper detector the machine could move in a room, towards a goal, while sensing and avoiding obstacles [38]. From this early pioneering experiment, the pursuit to build "intelligent" robots continued and progressed steadily, where, nowadays, neurorobots can describe the consequences of different neuronal models in a natural environment [39].

To use neurorobotics to study the brain, one needs to bear in mind that neural dynamics and functional neuroanatomy has to be as close as possible to a biological response [39]. In 2015 the resurgence of neuromorphic engineering (using inspiration from the brain to create computer architectures and sensors) allowed the use of spiking neural networks for neurorobotics [39], helping greatly to the realisation of computational models closely resembling the biological ones.

To implement a brain process in a robot and use its inputs to stimulate neural activity, there are three challenges to overcome:

- Identifying how the brain represent external stimulus (e.g. visual information, motion).
- Determining how this information is encoded in a brain model.
- Creating a link between the external representation and encoded information.

Currently, these problems are mainly transformed into linear models. However, the progress of machine learning and statistical modeling should allow the implementation of more complex and non-linear representations [40].

Surprisingly, even though neurorobotics is now an active field of research, to date, no robots can really match the behavioral and cognitive abilities of a rat, much less a primate.

My project is to focus on controlling a robot arm motion through a sensory motor loop and a computational model. Hence, I aim to first present robotic motion solutions already in usage in the field. I will then present the challenges of coding, decoding the sensory motor loop and how internal models are an alternative to treat sensory information. And finally expose the role of neurorobotics in PD studies.

2.2.2 Robot motor control

In robotics, object manipulation has been extensively studied, providing different solutions to perform the reaching and grasping action. Three things need to be determined in order to build a reach action:

- The trajectory (choice of the path)
- The coordinates to reach (from sensory to Cartesian data)
- The result of the motor command (the angle of each joint)

Basically, reaching is the result of an appropriate sequence of motor actions (torque and force for each joint), taking into consideration the current position of the arm and given the desired target point of the reach. The goal is to direct the endpoint of the arm to a desired target position. To achieve a successful reaching movement, some architectures are coded with predefined trajectories and use computer vision to determine the best path to reach the goal. Some other architectures, often including artificial neural networks (and motion data), teach the robot how to move. While, lastly, some other studies use brain machine interface (e.g. use of EEG) to convey the movement intention to a robot.

However, when the machine is supposed to mimic the movement of a natural arm, some recurring issues need to be considered [41, 42]:

- Redundancy of DOF, (in primates arms, makes learning movements difficult in robots because there is several trajectories possible for the same endpoint),
- Non-linearity between joint movement and hand's final position (some joints have more impact on the final position than others),

- The gravity and inertia in a suspended arm (i.e. what would be the most natural way to replicate it in robotics),
- Speed and noise effects in motor control signals (incomplete sensory information and speed reaction).

In order to interpret and map the inputs coming from a model while considering the points exposed above and then control the joints' motor accordingly, many approaches exist. The main ones will be briefly described.

Inverse kinematic

Positioning the hands to pick up an object can be achieved with inverse kinematic. Inverse kinematic is a mathematical process of calculating the joint positions knowing the desired position of the end point in Cartesian coordinates. This problem could be modelled and solved using equations and usually there is more than one way to reach the same position. Therefore, the movement has to be constrained or it is necessary to make a case by case decision on which approach to be used [43]. Through the sensory data (visual information), the end point position (x,y,z) could be extracted and sent to the inverse kinematic process. This solution doesn't allow any margin for adaptation. It's a hard-coded behaviour that will stay static whatever happens. When studying the brain using embodied models it may lower the realism of the behaviour process.

Artificial Neural Networks

To adopt an adaptive behaviour, Artificial Neural Networks (ANN) are more flexible and closer to biological behaviour. These Neural networks have the potential to learn from incomplete data with positive stimulus (correct answer) and negative stimulus (unwanted action) and generalize those behaviors into new situations. Some use learned error correction and predictive controls to simulate muscle memory while others try to recreate a general hierarchical structure of inputs/outputs to send an order to each "muscle" accordingly [41, 44]. Van Der Wal (in 2012) taught a NAO robot to effectuate a reach motion using neural networks. The network was trained with 1000 demonstrations of the reaching motion towards an object placed at different positions. The Network had 4 main parameters: the weights, the learning rate, the activation function and the training method. The results of this experiment, however, were inconclusive as the ANN (which used a back propagation training method) was not able to efficiently retrieve relevant information from the database, the motion wasn't accurate and the reaching failed to move toward the object [43]. This study shows that even if ANN are a powerful tools they are not always conclusive. ANN require a large dataset to be able to train themselves and there is a need of empirical testing in order to tune their parameters appropriately, therefore it's hard to tell whether or not the final model obtained is really

the best it could be or not. Typically, in robotics, ANN is used to learn the correct mapping between the action and the result (joint position, end-point position in the workspace) as an internal model approach would do [42], however, this will be further explained in section 2.2.4.

Deep Neural Networks

Deep Neural Networks, are more complex and entail deeper architectures than ANN but they follow the same basic principles. They have been used for robotic applications with promising results as they can solve highly complex problems and mimic human processes with an impressive adaptability. These networks are successfully used in number of studies to model neuronal responses [37]. One of these studies was the one done by Hossain and Capi (in 2016) [39], they created a deep belief neural network trained to recognize objects in different positions and orientations by extracting features out of images and then conceived a robot able to grasp them in real time. However some limitations with this approach should be noted: This approach is complex to integrate in a model and works in limited domains. Besides, the created model will often require lengthy and specific training and may not be able to express many of the behaviours considered natural, attributed to intelligence [39]. Therefore, from a robotic approach, it may be overly constraining to produce.

Evolutionary robotics

Another approach is to use evolutionary robotics. In this approach a set of "individuals" encoding the controller of the robot are randomly created in the environment. The performance of each individual in resolving problems is estimated, and the fittest robots are allowed to generate new offspring, with some change added to them (mutations, crossover, duplication). This process is then repeated until an optimal solution is found. Usually the body of the robot is defined (sensors and joints) as well as the environment (objects, physical constraints) and the algorithm has to learn how to best transform signal inputs into motor responses [42, 45]. It has the benefit to rapidly lead to adaptive behaviour while starting from a few set of parameters. It is possible to combine ANN (and especially this evolutionary robotics approach) and the inverse kinematic by making the system learn inverse kinematic solutions. Floreano & al.(2000) [45], eminent researchers in the subject, helped structure the field of evolutionary robotic by evolving, with this method, neural networks supporting a range of behaviours. From a robot escaping a maze (early 2000) to robots developing predator-prey strategies (2010)[39]. In their experiments the neural network controller was defined by a genome, taking inputs from sensory information and sending, as output, actuator commands. The genomes parameters were the weights of the neurons, the plasticity and topology rules of the network. The fitness of a genome was determined by how well the robot performed in a task [39]. Their work have been, since then, reproduced and adapted to many other studies.

However this solution may not solve all problematic, it is important to well define what we want to obtain, else, whatever the method used the result will be inconclusive.

The methods to effectuate a motion in a robot are diverse, but in neurorobotics the challenge is not the creation of the motion itself but interpreting the signal to translate into motor control. Moreover, the goal position, which is something extracted from sensory inputs, may greatly impact the computational model output signals. Which brings us to the sensory motor loop control, how to deal with sensory data and the computational model data.

2.2.3 Sensory motor loop control

Most often in neurorobotics, the robot sensory data is sent to the computational model which, in turn, delivers the inputs to the motor control.

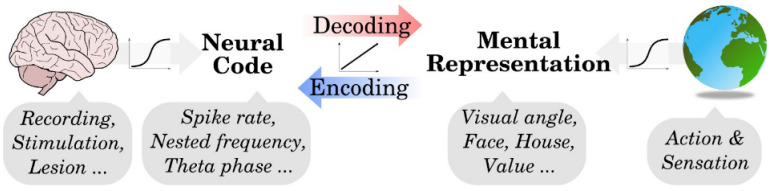


FIGURE 2.15: Schematic representation of the coding/decoding process from external stimulus to a computational model and vice-versa [40].

An important component to modeling a realistic and workable neurorobot is correctly interpreting the data sent from the robot to the model, and the data delivered by the model to the robot. A false assumption could be validated, if the wrong approach is used, as the motor control could palliate what the models do not provide. It could also distort the information being received to fit the intended result.

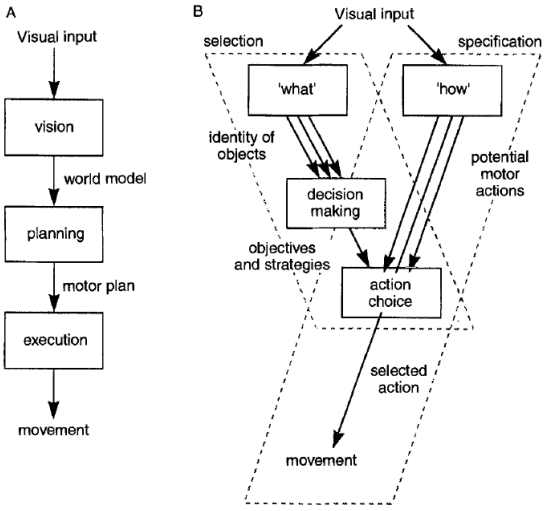


FIGURE 2.16: Example of two models of sensory data interpretation. (A) is the traditional approach with sequential tasks, easier in an engineering perspective. (B) is a schematic representation based on the supposition that the motion is more due to a "reaction" than a planning of visual stimulus input. There is here two really different interpretations of the visual input processing, yet they can both give the same output at the end [42].

2.2.3.1 Encoding sensory inputs

The choice of input features is a key step in the process. It has to be adapted to the computational model and stimulate it adequately. For example, a linear encoding of the sensory inputs will force a linear relationship between the inputs and the neural activity. And a non-linear encoding may express a more complex relationship, but be too complex to interpret, may require more data or may still not be enough to observe non-linear outputs [40, 46].

In the case of visual inputs, interpreting data obtained from cameras isn't trivial. Factors, such as lighting, exposure, resolution, camera parameters etc. can influence the outputs. The computational model could be stimulated by intensity via a linear equation (the more intense is a color, the more stimulated the model will be) or via higher-level features (as reacting to diverse objects detection) [46].

Many feature extractors like SIFT, SURF, FAST can be used to extract features [43] or whole packages as YOLO exist in order to extract features and recognize objects from a pre-learned database. As an example, Van der Wall (2012) work used SURF to extract features out of the robot's video feed. If an object was recognised, the observation and position of the object were sent to launch the grabbing function [43].

2.2.3.2 Neural activity decoding

In neurorobotics, the inputs controlling the motors are coming from simulated neuronal commands. The first step is to decide from which brain region extracting the data. Studies have shown that the basal ganglia could be used to select actions through the motor loop [17, 47]. Yet, how human and non-human primates learn and effectuate reaching and grasping remains controversial. There are two main approaches as to how the brain sends the move command for reaching and grasping [42]:

- **Internal Models** are based on the assumption that the human brain predicts how the limbs will move and the sensory response associated with it, transforms this desired sensory state into corresponding motor commands.

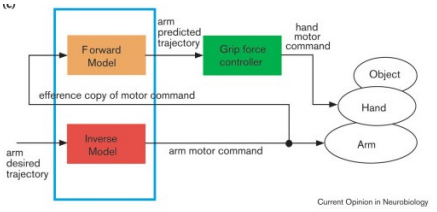


FIGURE 2.17: Coordination of grip force and load force, with a computational model based on internal forward and inverse models. Framework of a grip-force–load-force was achieved with both the inverse and forward models of the arm. The inverse model of the combined dynamics of the arm and hand calculates the necessary motor commands from the desired trajectory of the arm. These commands are sent to the arm muscles and to the forward dynamics model. Then, the forward model can predict an arm trajectory. Given the predicted arm trajectory, the load force is calculated to pick up an object [48].

- **Equilibrium Point Models** are based on the assumption that muscles, and their spinal reflexes, have stable positions, where they settle independently from the previous pose. Then, the central nervous system modifies the current equilibrium point to move.

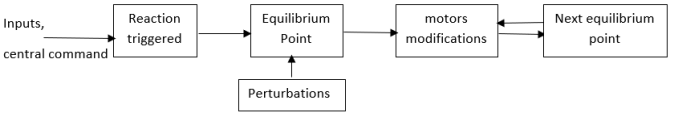


FIGURE 2.18: Causal chain of events during a motor performance. The central commands represents voluntary movements coming from the motor neurons. The events results in a shift of the system's equilibrium point. The motor outputs are the kinematic variables.

These two models (above) offer a different interpretation of the computational model outputs. The motor control could be lead by a desired output or by trying to reach

an equilibrium point. Whichever model is implemented, from a processor perspective (the processor being the robot central processing unit), the outputs sent to the arm's joint will be the same. The robot will, in both cases, effectuate an arm movement. In fact, those outputs have to be read by the robot and then adapted into coherent joint orders, considering the physical limitations of the machine. To do so, among existing solutions, linear-Gaussian decoders, recursive Bayesian decoders, or probabilistic mixture of trajectory models (MTM) are well used.

To briefly present them, the linear-Gaussian decoders assume a linear relationship between the artificial neural activity and the arm state (including position, velocity and acceleration) [49]. The recursive Bayesian decoder is considered more accurate, it's based on two models: the trajectory model (state of the arm from one equilibrium point to the next) and the observation model (relation over time of the neural activity and the arm state). Recursive model has been used, among others, by Brockwell (in 2004) for decoding an arm trajectory in ellipse tracing based on spike count and firing rates of the computational model cortex [50]. This technique largely relies on particle filtering [50], where the decoding method is computationally demanding. Finally the MTM mixes simple trajectory models (based on recursive Bayesian decoding) in a probabilistic approach to generate relatively complex dynamic behaviours. Yu & al consider that this method divides the decoding error by 38 to 48% compared to simpler trajectory models [49] but is much harder to compute. Both articles [40, 46] have been cited in more than hundred of other studies, supporting the seriousness of their work. However they may not be pertinent for this thesis, as their implementation may be too complex to reproduce for this case and require excessive computational power to obtain the expected results.

For this study it will be important to decide how to interpret the computational data sent to the robot. The choice to make depends on whether it is considered that the neural network activity describes the desired end point or real time motion, or, whether a linear decoder or a more sophisticated decoding model should be used. This will affect the realism of the realisation even if the motion is well executed through any of these methods.

2.2.4 Internal models

At first, implementing an internal model seemed interesting to treat sensory data, as an alternative to ANN, for instance.
Moving the hand in order to reach an object involves a series of sensorimotor transformations, coming from visual and motor information about the location of the object and the position of the limbs. That is, into motor commands that will bring the hand

to the desired goal. The general term "internal model" refers to the general notion that the central nervous system has knowledge about the body and environment in which it evolves. An internal model should be able to predict the consequence of a motion.

From a processor perspective, an inverse model could be used directly to build a controller, taking as input the desired behaviour and giving as output the resulting action. When a new desired behaviour is given, the controller asks the model to predict the action needed, as would any learner prediction model.

Cerebellar cortex is one of the few places successfully modeled by Internal models in experimental results. However, acquiring an inverse internal model through motor learning is computationally difficult. In this type of model, forward models are used a lot for motor control, as it can predict the trajectory resulting from a given motor command. Thus, providing the information that a feedback controller needs to generate a desired trajectory. The sensory feedback is always significantly delayed, and those delays can cause a feedback circuit to fall into large oscillations. A prediction of feedback by a forward model essentially provides the circuit with zero-lag feedback, allowing to control the oscillations due to delays [31, 51, 52].

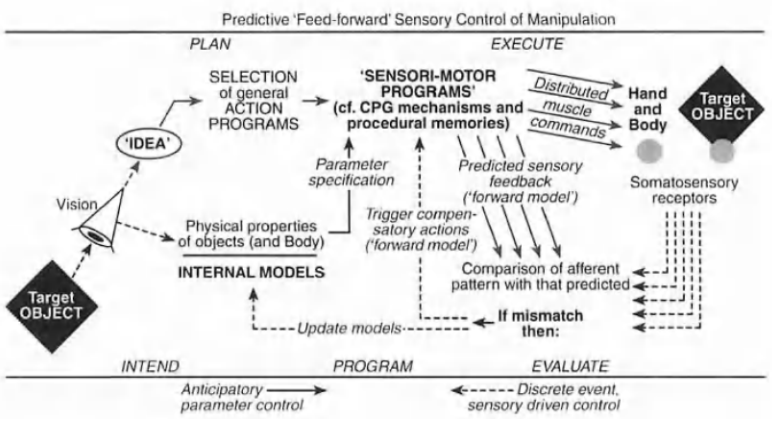


FIGURE 2.19: Predictive feed-forward sensory control for hand manipulation. Here internal models are used to extract parameters out of visual sensory inputs [51] .

To embed an internal model in a robot to test a computational model implying several areas may be overly complicated and not adequate for encoding/decoding sensory data or motor commands.

2.2.5 Neurorobotics models of Parkinson's Disease

Models have been embedded in robots to show the implications of the cortex in reaching [31, 44]. Prescott & al (in 2006) coded a list of actions to do according to the basal

ganglia outputs and virtual "needs" of the robots (hunger, fear) [17] . Bahuguna & al (in 2018) used D1 or D2 signals (of the basal ganglia) to determine the speed and direction of a vehicle [47]. These neurorobots showed how the Basal Ganglia, or more specifically D1 or D2 MSN, could function as action selectors [17, 47]. Both works have well detailed the process of their experiment and allowed a reproducible work. Prescott & al (in 2006) is an established reference proving that implementing a brain computational model into a robot to make him express a comportment is possible and relevant. His work is considered pioneering in the field. It is interesting to note that he used evaluation metrics such as the structure of the actions over time. As his work is comparable to what this thesis will be doing, these metrics may be considered for the work to plan in chapter 3.

But among all the models implemented in neurorobotics, there aren't many studies covering PD specifically. In PD cases, seeing the execution of an arm motion with and without PD symptoms according to the basal ganglia and cortex outputs is wanted. In order to do that, defining what is controlled and how to control it is necessary, as well as the model to be used and signals to interpret.
The motion of reaching and grasping can be modelled with the control of the end effector, the hand. Defining the decoder to interpret neuronal signals and its inverse kinematic, goal coordinates and speed while reaching this goal is also necessary.
In PD, the movement is slowed down and irregularly sequenced. Therefore, the outputs of the model selected as "go" signals are the most important factors to consider. The model presented by Kumaravelu & al [36], on which this work will be based, simulates healthy and parkinsonian brain areas such as M1, the putamen, the GPe and GPi regions. This model simulates the regions dynamics based on healthy/PD rat. According to [17, 29] the thalamus and M1 (and more specifically F5) are the most relevant regions to record, in order to analyse symptoms of the hand motion. Therefore, the model used in this thesis is relevant because the necessary regions of the motor loop are simulated. The question that remains is how to extract motor commands out of the difference of pattern and firing rates of those structures. While an inverse kinematic model is the easiest solution to implement, most studies present ANN as the most adaptable solution [42] to create a mapping between the inputs and the desired output (symptoms/no symptoms according to observed behaviour). The requirements and the necessary steps to be taken, in order to create a robot embodying such a model, are to be further developed in the methodology of this thesis.

2.3 Tools presentation

2.3.1 Nao

Nao is a humanoid robot (by SoftBank) designed to reproduce main human actuation. The robot is 57cm high and weight 4.5kg which makes it lightweight and compact.

FIGURE 2.20: Nao's picture [53] .

A lightweight robot means not only smaller and less powerful motors but also less thermal dissipation. The plus points are that owing to its light-weight, it has a larger acceleration range and better dynamic capabilities. A lightweight robot, such as NAO, is less likely to breakdown and is less dangerous than heavy-weight robots [54].

NAO is designed to perform smooth movements when changing speed and direction.

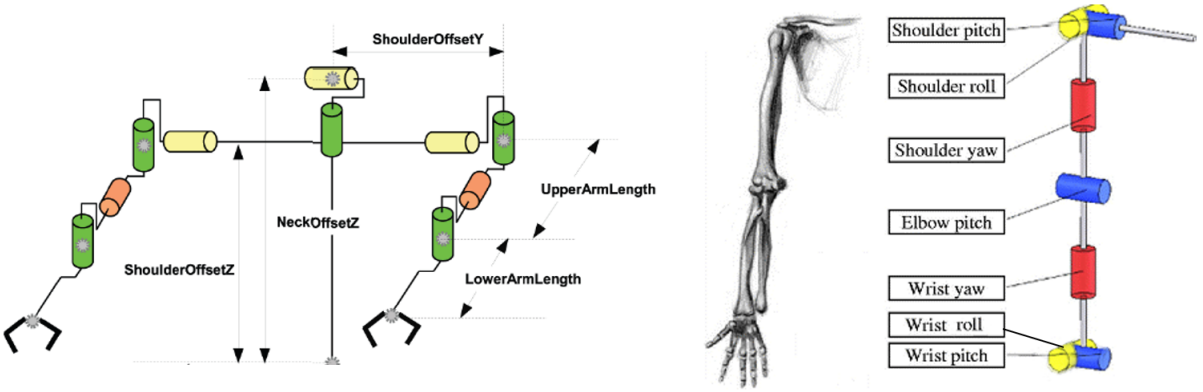


FIGURE 2.21: Comparison between NAO's kinematic upper body and a human arm kinematic. Nao wrist joint not represented. Colour representation: in Nao yellow is pitch, green is roll and orange is yaw [54].

Each arm has 5 Degrees of Freedom (DOF). Two rotational DOF for the shoulder, two for the elbow, 1 for the wrist and an additional one for the hand's grasping motion.

In comparison, a human arm has 7 DOF: Three for the shoulder, one for the elbow and three for the wrist. In Nao's architecture, the additional DOF in the elbow (a yaw motion) is there to palliate the absence of the third DOF of the shoulder and still imitate a human-like motion [53].

Motion	NAO's Arm (left) Range(degree)	Human's Arm Range(degree)	NAO's actuator type
Shoulder roll	-18 to 76	-90 to 90	M2R22
Shoulder pitch	-119.5 to 119.5	-90 to 180	M2R21
Shoulder yaw	X	-90 to 90	
Elbow roll	-88.5 to -2	X	M2R22
Elbow yaw	-119.5 to 119.5	X	M2R21
Elbow pitch	X	0 to 140	
Wrist roll	X	-60 to 60	
Wrist pitch	X	-30 to 20	
Wrist yaw	-105 to 105	-90 to 90	N/A
Hand	Open and Close		N/A

TABLE 2.1: Comparison of NAO's arms and human arms in Joint type, Range and actuators.

Nao's hand is limited with only the torque being modifiable through a stiffness parameter and an open and close action. The aperture of the hand is modifiable but limited to approximately the size of a 3cm radius ball [53, 54] .

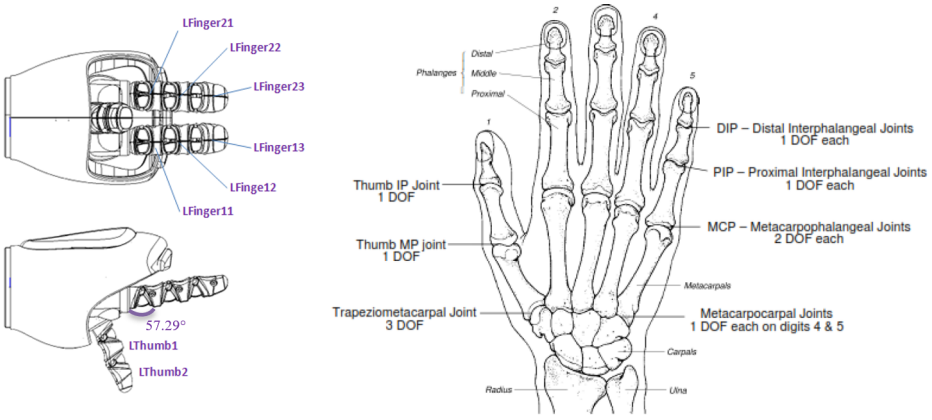


FIGURE 2.22: Comparison between NAO's hand and a human hand. Nao possesses 1DOF while a human hand possesses 27DOF [53].

The main sensory modules of NAO are press buttons on top of NAO's head, sonars, tactile sensors, joint position sensors, a video camera and a microphone. NAO has 2 cameras on its head, which enable him to see the forefront and under side of an object, with a focus of 300mm minimum. Its camera angle of vision is 61° from the centre of its head [53].

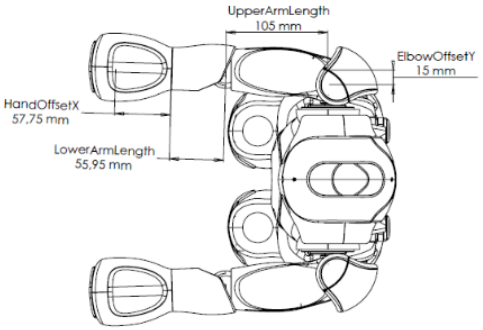


FIGURE 2.23: Nao H25 V3.3 top down view blueprint [53].

However, all these difference between NAO and a primate's body doesn't make the embodiment of a monkey brain model in this humanoid robot irrelevant. For one, because the system already created can be adapted to another robot, since the body of the robot has little impact on the computer's architecture (how the inputs are treated etc.). And two, because we should be able to still observe PD symptoms in the arm movement even if it has less DOF than a human arm.

2.3.2 Neuron NetPyNE

NetPyNE (Networks using Python and NEURON) is an open source Python package, which facilitates the development, simulation, parallelization, and analysis of biological neuronal networks using the NEURON simulator. It allows for the conversion of a set of high-level specifications into a NEURON network model defined simply. The specifications are provided in a simple, standardized, declarative Python-based format for an easier use [55].

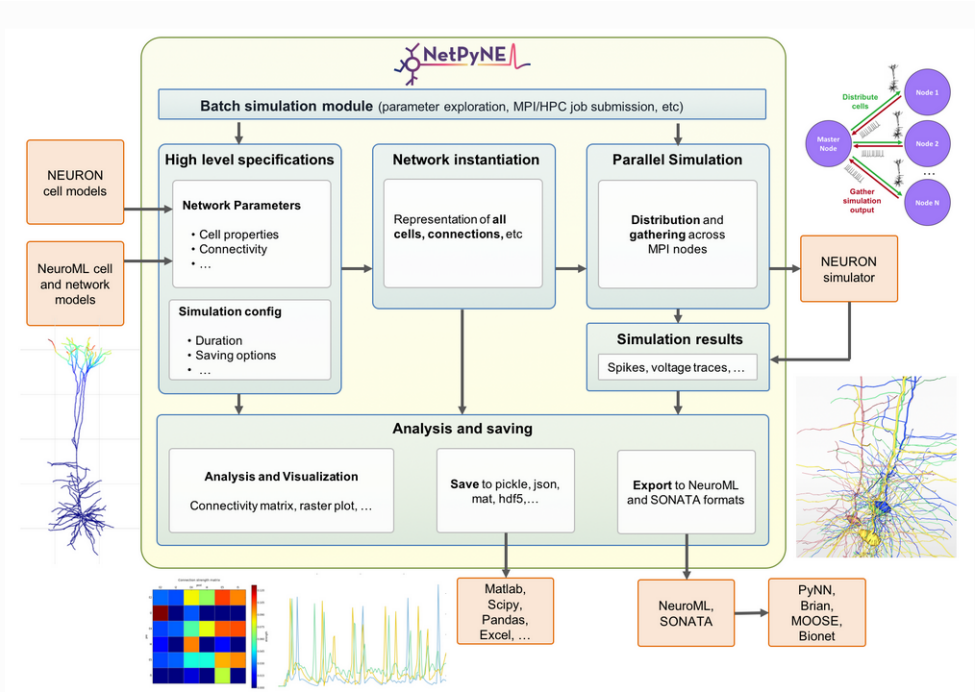


FIGURE 2.24: condensed overview of NetPyNe functionalities [55].

2.3.3 ROS

The Robot Operating System (ROS) is a set of open source software libraries and tools that help you build robot applications - from drivers to state-of-the-art algorithms, and with powerful developer tools. Created in 2007, by the company Willow Garage, it is now developed by Open Robotics. It offers an operating system for any robotic machine in Python or C++ alike and it enables communication between sensors, motors. ROS imitates the computers operating systems to facilitate robots programming [56].

2.3.4 Robot simulation on Webots

Webots is a professional mobile robot simulation software package, on open source since 2018. It offers a rapid prototyping environment that allows the user to create 3D virtual worlds with physics properties such as mass, joints, friction coefficients, etc.

Many mobile robotics projects have relied on Webots for adaptive behaviour research (genetic algorithm, neural networks, AI, etc.) Webots was a commercial project at the origin, therefore it was build with neophytes in mind, which makes the usage easier. Furthermore, it allows a large variety of programming languages (to code the robot

behaviour). such as C, C++, Java, Python or MATLAB, and in VRML97 (Virtual Reality Modelling Language) [57].

Webots is physically realistic, however in a coding perspective it is not. Webots expect the user to use his libraries in order to command the robot and do not broadcast the robot's topics. Making any conception in webots only usable in webots and not directly transferable in a real robot.

Chapter 3

Organisation

Chapter three aims to provide more details about the work I intend to do for this master thesis. The first section is a presentation and explanation of my objective; the second shall outline the requirements to meet that objective, followed by a presentation of the prototypes to be deliver in order to assess the work achieved at each step of the thesis. Additionally, the risks that could occur are taken into consideration. A Gantt chart of the project plan is included.

3.1 Project task analysis

3.1.1 Objective

The objective of my work is to contribute to the understanding of Parkinson's Disease's (PD) motor symptoms by embedding a computational model of PD on a humanoid robot (e.g. NAO robot) capable of performing a reach and grasp motion according to a visual stimulation and neural activity[1]. To rephrase it, the goal of this thesis is to realise a robot architecture to display healthy/PD symptoms. This neurorobotics model of PD could help to facilitate further PD studies on the brain, body, environment interactions. The implementation of a mapping of NAO's arm-joint based on the neural activity mapping will be inspired from studies of motor cortex signalling patterns and linear decoding models in order to:

- Reach and grasp an object.
- Mimic a "healthy" robot reach and grasp motor dynamics
- Investigate the consequence of PD motor neural commands on the same motion.

[1] All the tasks will be realised only in simulation due to the Covid 19 Pandemic.

3.2 Requirement analysis

To realise the objectives, the work has been divided into four tasks to complete. They are introduced in Table 3.1. Task 1 is the pillar of the whole project, it is meant to assure that I have a good understanding of the tools that need to be used. Tasks 2, 3 and 4 relate to the objective. Task 2, the sensory mapping, has a medium priority and task 4, the grasp implementation, a low priority. Therefore, the most important task of the project is task 3, mapping the outputs of the computational model into motor commands.

No.	Requirement	Priority	Predecessor
1	Understanding how to best implement a motion in NAO according to our objective	High	None
1.1	Understanding Webots	High	None
1.2	Understanding NAO's sensory outputs (includes visual outputs)	High	1.1
1.3	Understanding how to control a motion in NAO	High	1.1
1.4	Deciding how the motion will be executed	High	1.3
1.5	Implementing a reach motion in NAO	High	1.4
2	Implementing a sensory mapping to send to the computational model	Medium	1.2
2.1	Selecting sensory data extracted from NAO	Medium	1.2
2.2	Designing a mapping of the sensory data	Medium	2.1
2.3	Deciding where to send this data in the computational model	Medium	2.2
3	Implementing a motor control map to send to NAO based on healthy/PD neural activity	High	2
3.1	Deciding from which area of the computational model will the outputs be extracted	High	2
3.2	Decoding computational model outputs	High	3.1
3.3	Designing a mapping of the data	High	3.2
3.4	Implementing the motor control in NAO based on healthy neural activity	High	3.3
3.5	Implementing the motor control in NAO based on PD neural activity	Medium	3.4
4	Implementing the grasp motion in NAO based on healthy/PD neural activity	Low	1,2,3
4.1	Implementing the grasp motion in NAO based on healthy neural activity	Low	1,2,3
4.2	Implementing the grasp motion in NAO based on PD neural activity	Low	4.1

Table 3.1: Table of tasks

All the tasks will be realised in simulation due to the Covid-19 outbreak and will make full use of the webots, simulated NAO, ROS, Python and C,C++ .

3.2.1 Task 1: Understanding how to best implement a motion in NAO according to our objective

At the onset, it's important to understand the different ways a motion can be implemented in a humanoid robot (in simulation) and get acquainted with the tools (Webots, NAO) that are to be used in this project. ROS, python and Cpp will be used for the implementation.
As this will be my first exposure to working with Webots and NAO, the first step would be to familiarise myself with Webots usage, and then to NAO's sensory outputs (touch, camera, etc.), in order to be able to determine appropriate sensory data to be used. That is: the format, the accuracy, and the outcomes. These will provide an understanding that will help to map and encode the most appropriate sensory information.
Then, I plan to focus on understanding how the joint control works on NAO, so that I can understand how to create a bridge between the neural activity of the model and the arm/hand joints of the robot, with the help of ROS.

This task also includes the visual processing that may be required in order to determine the position of the object.

3.2.2 Task 2: Implementing a sensory mapping to send to the computational model

The second task consists of deciding on the nature of the data to map and how to send it to the computational model, using the following set of sub-tasks:

- Deciding which sensory data to analyse;
- Deciding how to treat this data, what will be the main features that will trigger stimulation (e.g.will it be the intensity of stimulation or another specific feature of the sensory outputs);
- Mapping this data into a value compatible with the computational model;
- Determining which region of the model to send this stimulation.

This task also includes calibrating the visual processing to extract the appropriate sensory data.

3.2.3 Task 3. Implementing a motor control map to send to NAO, based on healthy/PD neural activity

I anticipate that task 3 will be challenging as it involves the extraction and decoding of the model's neural activity to send to NAO's arm. I will be seeking clarity on the following aspects of the task:

- How will NAO's arm be controlled (e.g. by endpoint velocity, or position).
- From which brain region to extract the data.
- How to treat this data, which strategy to use (e.g. linear or recursive method).
- How to map this data into a coding value understandable by NAO .

Once the mapping is conceived based on fixed inputs, it will be connected with the sensory mapping and the healthy/PD neural activity. Therefore, this phase will be divided into 2 stages. In the first stage, the mapping will be done based on healthy neural activity data. Then, in the second stage, the results will be tested against the PD's neural activity data-set and adjusted accordingly.

3.2.4 Task 4. Implementing the grasp motion in based on healthy/PD neural activity

Finally, the grasp motion will be added to the reaching motion implemented in task 4 by analysing the motion state of the arm. Arm and hand take different inputs from the brain to operate but considering that NAO's hand has only an open/close mechanism, it can be realised by estimating the state of the reaching completion and how close it is from the object, that it has to grasp.

3.3 Performance assessment

To complete the objective defined in my Section 3.1, five prototypes are planned, as defined in Table 3.2. These prototypes will be assessed by comparing motion graphs between one prototype to the next, by verifying the performance of the system (i.e. are the results repeatable?) and the usefulness of the metrics to study Parkinson. Finally, these prototypes will be evaluated by Dr.P. A. Vargas, Dr.R. C. Moioli, and myself.

Prototype	Due date	Assessment
Prototype 1: NAO's motion control	19/05/20	NAO able to reach, grasp and send sensory feedback
Prototype 2: Sensory mapping	02/06/20	A mapping design of NAO sensory feedback able to produce an activity in the computational model
Prototype 3: motor control implementation with a healthy neural activity	30/06/20	A mapping design of the computational model outputs capable of moving NAO's arm for a healthy neural activity
Prototype 4: motor control implementation with a PD neural activity	10/07/20	A mapping design of the computational model outputs capable of moving NAO's arm for a PD neural activity
Prototype 5: Implementation of the grasp motion with and without PD	17/07/20	NAO can react to a stimulus by reaching toward an object and try to grasp it

TABLE 3.2: Table of prototypes

3.3.1 Prototype 1: NAO's motion control

The first prototype will be based on the implementation of a simple reaching motion on NAO according to a visual input. The humanoid robot should be able to analyse sensory feedback and perform the reach and grasp action towards an object.

The goal of this prototype is to familiarise myself with NAO and determine the sensory data and motor control data to be used on prototypes 2 and 3.

The assessment of Prototype 1 will consist of a plotted-on-graph display of the humanoid robot joints' arm position and hand opening/closing over time, while it performs a reach and grasp movement. The criteria being:

- The graphs have to be pertinent to evaluate the next prototypes.
- The position of the object toward which NAO will reach must be correctly evaluated, with a margin of a few centimetres (if the object is a ball, the maximum allowed error is half of its radius).

- In order to prove the accuracy of the object position estimation, the object must be displaced by NAO's hand.

3.3.2 Prototype 2: Sensory mapping

The second prototype will be a mapping of sensory inputs translated into signals readable by the computational model.

NAO's chosen sensors have to react to a sensory stimulus and send a signal to the computational model. The model has to be triggered and respond to some signal intensity. In this prototype, both PD and healthy models have to somewhat react to the stimulation.

Here a discrimination of the sensory stimulus must be displayed. For instance, if the sensory stimulus is chosen to be a visual stimulus, then the visual stimulus must be able to generate at least two different answers. This is necessary in order to send more than one input to the computational model.

This prototype will show us how different stimulus will create different computational model activities.

To summarise, this prototype must show that:

- the sensory stimulus must be able to generate two distinct answers (to send to the rat model).
- based on the inputs, the computational model outputs must be comparable to the classical model of the brain's basal ganglia. Thus, a healthy and PD neural activity must show different behavioural characteristics.
- Moreover, unrelated stimulations must result in dissimilar brain outputs.

These condition are necessary in order for the study to be relevant and for the other prototypes to be generated.

3.3.3 Prototype 3: motor control implementation for a healthy neural activity

The third prototype will be a mapping of the motor control command based on the computational model's signals.

NAO's arm will reach towards a goal under the direction of the computational model data. It must display a natural reaching motion for a healthy neural activity. The

assessment will hence be a demonstration on Webots of NAO reaching towards a goal with a healthy motion. The criteria for assessment, will include the following:

- This prototype will be able to display a plot representation of the arm joints over time depending on the sensory inputs.
- The plotted path should be comparable to the realisation of prototype 1 and show no sign of Parkinson symptoms.
- The arm motion has to be as natural as possible, fluid, without abrupt motions of the joints.

The metric used here should be relevant for further studies on the model.

3.3.4 Prototype 4: motor control implementation for a PD neural activity

The fourth prototype will demonstrate NAO reacting to a sensory stimulus and reaching towards it, as in prototype 3, with the difference that it will take on PD neural activity. Therefore the model is expected to display PD symptoms, or at least display motor or task performance differences from the healthy model. Here, the aim would be to observe if there are any noteworthy changes- like is the model displaying any PD symptoms or any differences in motor or task performance when compared to a healthy model.

This implementation will be assessed with the same metrics as the previous prototype with graphs comparable to prototype 3.

- The display of the symptom must have a physical meaning in a Parkinsonian case,
- show a clear, different motion from the Healthy one.
- The graph of the motion over time must be relevant to study Parkinson, in accordance to what is known of the disease from the literature review.

The aim is to create a functioning architecture according to the objective.

3.3.5 Prototype 5: Implementation of the grasp motion with and without PD

The fifth prototype will observe reacting to a sensory stimulus, reaching towards an object, and trying to grab it. It should be able to display the same motion as in prototype 3 and 4 with a healthy neural activity and a PD neural activity.

The metrics to assess this prototype will be:

- Plots displaying the arm and hand opening and closing over time with and without PD neural activity;
- The healthy and PD arm motion should be comparable to the ones obtained in prototype 3 and 4;
- The difference between the PD and Healthy neural activity must be visible on the arm and hand graphs;
- The robot must visually show that the object is grasped, by lifting it.

The goal of this prototype is to create a fully functioning architecture according to the objective.

3.4 Risk analysis

In this section, I would like to outline the reasons that may delay my work or prevent me from achieving the objectives presented in Table 3.3.

No.	Risk	Probability	Response
1	Difficulty in manipulating NAO	Low	Ask people who have worked on NAO for help
2	Difficulty in implementing the mapping on the robot	Low	Ask people who have worked on NAO for help
3	Laptop failure	Medium	Change the faulty part, restore backup, borrow a laptop
4	Difficulty in understanding the computational model outputs	Medium	Ask the team who created the computational model for help
5	Difficulty in creating the mapping of the computational model data	High	Ask my supervisors for help
6	Difficulty of access resources, materials or data in terms of delay due to the covid-19	High	Ask my colleagues or supervisors for help

TABLE 3.3: Table of risks

3.4.1 Risk 1

There might be a remote possibility that I might find manipulating and/or fine-tuning NAO challenging. It is unlikely, considering I have a mechanical and robotic background. However, if this happens, I have plenty of external resources and support who can help me overcome this - be it via internet research, online help or I could solicit advise from colleagues, who have worked with NAO.

3.4.2 Risk 2

I may be surprised by unexpected difficulty in implementing my work on NAO . If I can't seem to understand the motor control commands or if the robot doesn't react as intended. If this happens, I will breakdown the problem to understand where the problem comes from, search online resources or ask colleagues who previously worked on NAO for help.

3.4.3 Risk 3

A laptop failure can happen any time, especially when we use it extensively over a long period of time. Multiple reasons can cause such a failure: old computer, hazardous test of a new driver or kernel, viruses (quite unlikely), rain (quite likely). Therefore, it is important to regularly backup my work on an external device or on an on-line platforms (Github/Google Drive).

3.4.4 Risk 4

I may have difficulty in understanding the computational model outputs and how to use them. As I have no prior knowledge about it, there is a medium likelihood of this happening. In the case, I don't understand how to manipulate the model outputs, I can reference tutorials online or directly ask the designers of the model to help me with the how-tos.

3.4.5 Risk 5

The mapping of the computational model outputs will likely be the longest part of the thesis. The creation could easily be hindered by difficulties to comprehend certain aspects. If the model signals are not well interpreted, and the mapping parameters are

not well designed etc. In such a case, I could research further via online resources or ask help from my supervisors.

3.4.6 Risk 6

Due to the Covid-19, some access to resources such as documentation and support, that would be usually easily available from the University may be a bit more difficult to obtain, or may be delayed. In such a case, contacting my supervisors or colleagues about it may be the best solution.

3.5 Project Plan

My project plan is presented in the form of a Gantt chart in Figure 3.2. It presents the tasks (in Orange) and sub-tasks (in blue, on the gantt chart) that are needed to be done in order to complete the thesis. It includes the completion and assessment time to realise each prototype (in green). It also provides the times allotted to complete each task. Although, these are only estimates and some sub-tasks are superposed, wherever there is an overlap in tasks.

	Task Name	Dur	Start	Finish	Predece
1	Writing thesis report & litterature review	128 day	Sat 29/02/20	Fri 24/07/20	
2	Understanding how to best implement a motion in NAO according to our objective	14 days	Mon 04/05/20	Tue 19/05/20	
3	Understanding Webots	2 days	Mon 04/05/20	Tue 05/05/20	
4	Understanding NAO's sensory outputs	4 days	Wed 06/05/20	Sat 09/05/20	3
5	Understanding how to control a motion in NAO	3 days	Sat 09/05/20	Tue 12/05/20	3
6	Deciding how the motion will be executed	3 days	Mon 11/05/20	Wed 13/05/20	5
7	Implementing a reach motion in NAO	5 days	Wed 13/05/20	Mon 18/05/20	6
8	Implementing a sensory mapping to send to the computational model	12 days	Sat 09/05/20	Sat 23/05/20	4
9	Selecting sensory data extracted from NAO	3 days	Wed 20/05/20	Fri 22/05/20	4
10	Designing a mapping of the sensory data	7 days	Sat 23/05/20	Sat 30/05/20	9
11	Deciding where to send this data in the computational model	3 days	Sat 30/05/20	Tue 02/06/20	10
12	Implementing a motor control map to send to NAO	33 days	Wed 03/06/20	Fri 10/07/20	8
13	Deciding from which area of the computational model will the outputs be extracted	2 days	Wed 03/06/20	Thu 04/06/20	8
14	Decoding computational model outputs	6 days	Fri 05/06/20	Thu 11/06/20	13
15	Designing a mapping of the data	13 days	Thu 11/06/20	Thu 25/06/20	14
16	Implementing the motor control in NAO based on healthy neural activity	8 days	Mon 22/06/20	Tue 30/06/20	15
17	Implementing the motor control in NAO based on PD neural activity	8 days	Wed 01/07/20	Thu 09/07/20	16
18	Implementing the grasp motion in NAO	6 days	Sat 11/07/20	Fri 17/07/20	2;8;12
19	Implementing the grasp motion in NAO based on healthy neural activity	3 days	Sat 11/07/20	Tue 14/07/20	2;8;16
20	Implementing the grasp motion in NAO based on PD neural activity	3 days	Wed 15/07/20	Fri 17/07/20	2;8;17

FIGURE 3.1: zoom on the Gantt chart, allocation time of each task and sub task detailed

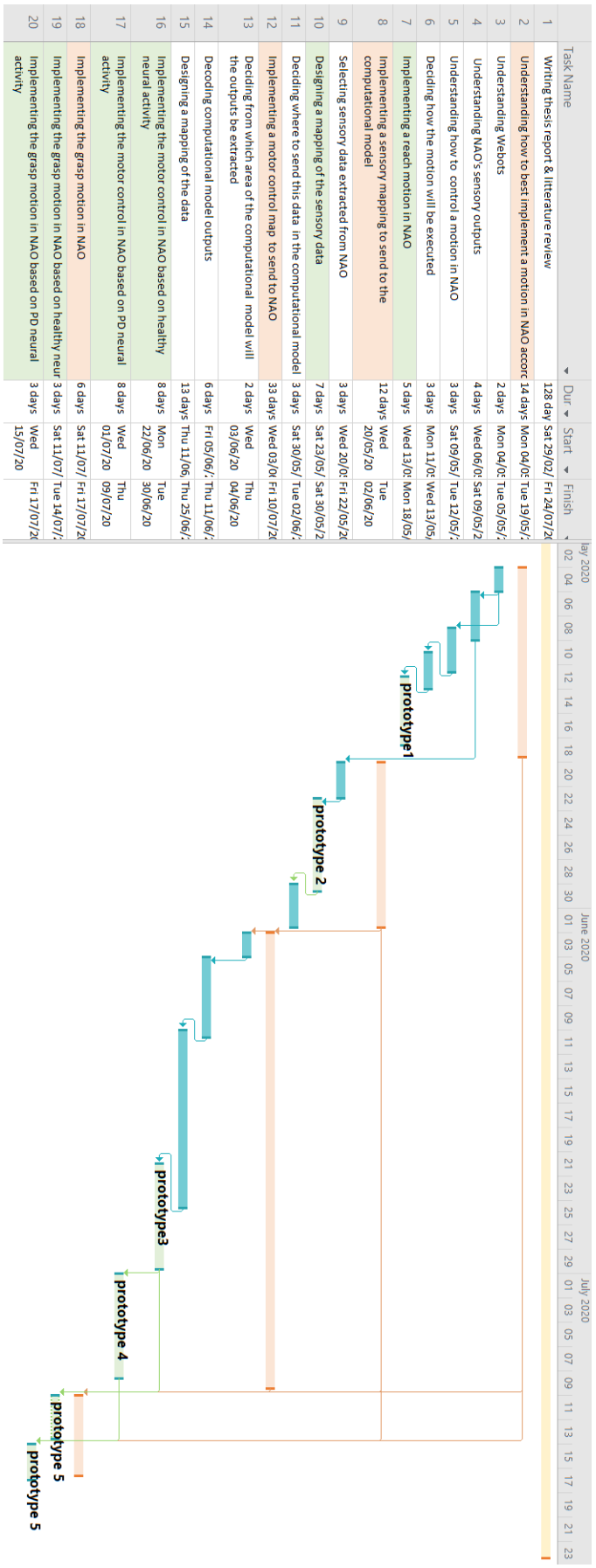


FIGURE 3.2: Gantt chart

Chapter 4

Methods and Results

This chapter presents the diverse prototypes developed in order to complete the project as introduced in section 3.3. The design requirements of each prototype and how they were met are mentioned in a sub-section titled "Design", followed by "Implementation", where the technical details will be provided and finally, the "Results and analysis" of each prototype will be presented.

It should be noted, here, that the following work has been realised in a Webots simulator, on Ubuntu 18. The simulated NAO can't be fully compared to a real NAO robot as NAO's functionalities, NAOqi, could not be used in the simulated stage (this will be explained in the following section). Therefore, some of the solutions proposed in this simulation have to be replaced by NAO's functionalities on a real NAO. When a significant difference between a real NAO and the simulation implementation arises, it will be noted and an alternative to adapt this work to NAO will be briefly described to be further investigated as a future scope of work.

It was determined for all the calculus of the following sections that the NAO used in simulation is a H25 V3.3.

4.1 Prototype 1: NAO's motion control

4.1.1 Design

As defined in section 3.3.1 the goal of the first prototype was to, primarily, simulate a reaching and grasping motion on NAO, towards an object. To meet those requirements the prototype had to:

- detect and recognise an object,
- assert its position,
- calculate the joints position in order to reach this end point,
- execute the motion.

Secondly, different sensory feedback had to be extracted from the robot as:

- joint position sensors,
- NAO's camera data,

The prototype will be presented, step by step, from object detection to the sensory feedback choice.

Everything was implemented in ROS, Python and CPP.

4.1.2 Implementation

4.1.2.1 Visual processing

Selection of camera

As the whole project is to be conducted in a simulator, a choice had to be made between using an external , as a computer webcam, or the simulated NAO's bottom camera.

Real cameras are more realistic than the simulated ones in terms of depth, image resolution and are more adapted to the use of object recognition packages such as "find 2D object" or NAO's integrated object recognition. However, the object detected by a real camera would be out of the simulation bench and the real distance between the object and camera, will have to be manually measured, and consequentially, may be wrongly asserted, making the ground truth untrustworthy.

Therefore, the first prototype has been tested with the simulated NAO's camera, to be able to really verify the results with the given position of the object in simulation (and obtain a trustworthy ground truth). A second version was realised with a webcam and provide a prototype adaptable to real situations. The second version will not be used in the rest of the thesis and is therefore in Appendix 7.1.

From now on, "NAO" or "simulated NAO" will refer to the simulated NAO and the real NAO will be referred as "real NAO".

NAO and the simulated bottom camera

NAO has two camera, a top one on the front and a bottom one in the mouth. Only the bottom one will be used as we want to see objects close to the robot. It is important to note, that Webots NAO's simulated camera has different properties than the real NAO's camera. All the thesis and following tables were realised with the bottom still camera of the simulated NAO.

	Real NAO H25 V3.3	NAO Simulated Camera
Camera Resolution	640x480px	160x120px
Camera Horizontal Field of View (FOV)	47.8°	45.00°
Camera Vertical Field of View	36.8°	33.75°
Object recognition	Yes	None
Camera Focal Length	300mm	None

TABLE 4.1: Comparison of NAO and the simulated NAO's bottom camera

The simulated camera outputs an image in both RGB and in grayscale, in this project both colorscales are used.

Defining Parameters

A ball was set on the table in front of NAO in its observable range. The table height is right below NAO torso, 35cm high, 1.8 metres long and 1.5 metres large. NAO has his arms straight up before him and its head looks straight in front of him (See Setup Figure 4.6a).

Henceforth, the object which will be used from now on in the simulation is a ball with the characteristics described below.

Object Ball
Mass 0.16kg
Radius 32.5mm

Since the simulated camera, or "NAO's camera" as it will be written from now on, has no focal length, the value was chosen according to what would be most ideal (considering that there is a 50.71 mm offset on the x axis between the camera and the centre of NAO's

torso). Most ideal means here that there will be no shifting needed between the object position as determined by the visual processing unit and the kinematic unit.

A realistic way to calculate the focal length of a camera in pixels with the available data is:

$$Focal\_Length = \frac{image\_width}{2 * tan(\frac{FOV}{2})} \tag{4.1}$$

In the case of this thesis, the Focal length was chosen in order not to have the X offset in the calculation. Hence, after tuning it through trials, the Focal length was chosen to be: 182px.

Method of feature extraction selection

With the selected camera, the object parameters and focal point defined, an image was extracted from the simulation, sent as a ROS topic and transformed into an image readable by the python library OpenCV.

OpenCv (which is designed to solve computer vision problems) was used for the following steps (see Figure 4.1):

Transform the grayscale image into a canny picture → extract circles/ellipses.

In this simulation, NAO was able to estimate the correct distance with the specific simulated ball described above.

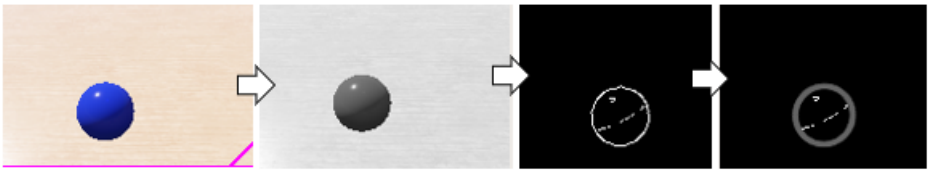


FIGURE 4.1: Visual processing steps to extract the radius of the ball. Grayscale image/ canny filter/ellipse extraction

To calculate the distance, a circle or ellipse extraction method was to implement in order to have the radius of the ball and be able to estimate its distance.

As a choice had to be made between extracting an ellipse or a circle from the image, a full comparison between both methods is explained in the Result and analysis section of this prototype. However, for now, in brief, the ellipse method was chosen to be implemented because it is more precise in the reachable zone of NAO and is less permissive than the circle detection.

Limitations: this feature extraction method may require a cleared field of vision and can manage only one object - a ball- in order to give the best results.

Position extraction

An ellipse has a width and height radius. It was observed that both height and width gradually decrease the further the object is placed, therefore a radius could be calculated (considering, the object is somewhat circular) as follows:

$$object\_radius\_in\_px = \sqrt{\frac{\frac{Ellipse\_width}{2} + \frac{Ellipse\_height}{2}}{2}} \tag{4.2}$$

The calculations used to extract X and Y from the image are (Reference frame depicted in Figure4.3):

$$X = camera\_focal\_length \times \frac{real\_object\_radius\_in\_mm}{object\_radius\_in\_px} \tag{4.3}$$

$$Y = -1 \times (object\_position\_px\_horizontal\_axis - 80) \tag{4.4}$$

In the equation above: 80px is half the width of the image, used in order to centre the Y axis in the middle of the image.

Since everything is realised in simulation there is no need to adjust the position according to the pixel resolution of the camera, In this ideal case 1mm = 1px. This would be completely inaccurate with a real camera (see Appendix 7.1).

The minus 1 is used to have the Y axis adapted to NAO's reference frame (the ball maybe at y=-20mm in the simulated world, but from NAO perspective it's at y=20mm on its left).

The Z element is fixed and can be freely changed in the code. The ball is placed on a table. If the table's height changes, the Z parameter has to be adapted. Finally, in order for NAO to view a broader area an up-and-down head motion was implemented, so that the ball can be seen at any distance (along the x axis). NAO's head moved from its minimum down position and back up, far enough to see the length of the table.

If a ball is identified by NAO at some point, then the head movement will be interrupted and the arm motion towards the ball will be launched.

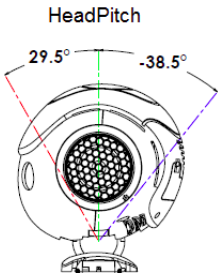


FIGURE 4.2: NAO's head pitch max angles in degree [53].

4.1.2.2 Kinematic implementation

On the real NAO, a NAOqi library does the inverse kinematic calculation. However, as the NAOqi library was not adapted to Ubuntu 18, (the computer system used in this project), and the last Python SDK supported system version was 16.04 [58], the inverse kinematic of NAO was achieved by relying on Kofinas Thesis [59] and Kofinas and noobwestand[1] codes.

With the real NAO, there would be no need for this part of the implementation, once the object position retrieved, it should be directly fed to the following NAOqi function: "motionProxy.positionInterpolation" [60].

The base frame of the kinematic is the centre of NAO's torso (see Figure 4.4).

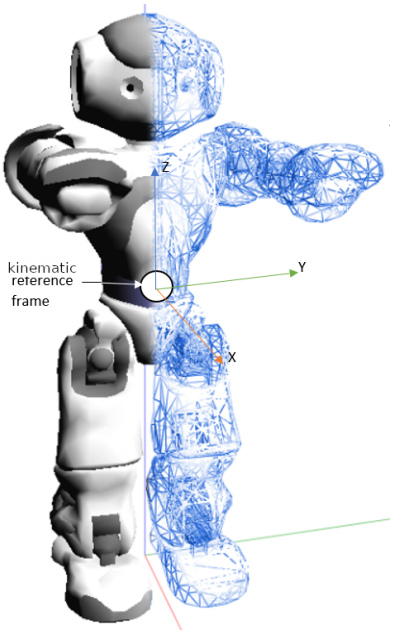


FIGURE 4.3: NAO's kinematic reference frame is at the centre of the torso.

[1] https://github.com/noobswestand/NaoPythonIK

Frame (Joint)	**a**	**α**	**d**	**θ**
Base	$A(0, \text{ShoulderOffsetY}, \text{ShoulderOffsetZ})$			
LShoulderPitch	0	$-\frac{\pi}{2}$	0	θ_1
LShoulderRoll	0	$\frac{\pi}{2}$	0	$\theta_2 + \frac{\pi}{2}$
LElbowYaw	ElbowOffsetY	$\frac{\pi}{2}$	UpperArmLength	θ_3
LElbowRoll	0	$-\frac{\pi}{2}$	0	θ_4
Rotation	$R_z(-\frac{\pi}{2})$			
End effector	$A(\text{HandOffsetX+LowerArmLength}, 0, 0)$			

FIGURE 4.4: Parameters of the modified DH of NAO's left arm [59].

To simplify the calculations, only the Shoulder and
elbow components of NAO's left arm composed the forward kinematic (see red square Figure 4.4). A modified DH method was used (click on hyperlink to obtain an explanation on what is the classical and modified DH methods and on joint tables T_{i-1}^{i}).

The following forward kinematic is obtained with it: $T_0^4 = T_0^1 T_1^2 T_2^3 T_3^4$.

The forward kinematic can determine the cartesian and angular position of the end point (px,py,pz and ax, ay, az) knowing the joint angles. The Inverse kinematic can determine the joint angles when knowing the end point position. One method to obtain the angles is to compare the forward kinematic to the end point matrix (see Literature review section 2.2.2). In this case, the end point matrix is as follow:

EP =

$$\begin{bmatrix} C(ax)C(ay) & S(ax)S(az) - C(ax)S(ay)C(az) & C(ax)S(ay)S(az) + S(ax)C(az) & px \\ S(ay) & C(ay)C(az) & -C(ay)C(az) & py \\ -C(ay)S(ax) & S(ax)S(ay)C(az) + C(ax)S(az) & -S(ax)S(ay)S(az) + C(ax)S(az) & pz \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

With EP being the End Point matrix, C(a_i) being Cos(a_i) and S(a_i) being Sin(a_i).
The final forward kinematic, on which the calculus is based is explained in appendix 7.1.

With a 5 DOF arm, the end point can be reached through several configurations, not all of them possible as limited by the joints maximum movement range. The forward kinematic can be used to verify if the joint configuration obtained is correct or not. In this implementation the joint angles obtained were not verified but constrained to stay in their possible range. Consequently, some positions presented aberrant motions because the kinematic angles had aberrant values.

With regards to the hand motion, it is important to recall that there is a notable difference between the simulated NAO and the real robot. In the simulation NAO has

3 fingers composed of 8 actuators, while the real NAO's hand only has 1 actuator (open and close). The code to be used in both cases is greatly different as the actuators, in simulation, don't have the same name as in the real NAO.

The joints angles were sent to NAO via ROS with a fixed velocity and acceleration speed. The velocity being constrained on Webots between 0 and 10 rad/s and the acceleration between 0 and 100 rad/s^2.

The joint poses were sampled at regular intervals, thanks to a parallel thread. Note that the time slots may vary on the tests (from a sampling of 120ms to a 20ms). If a sleep function is used the interval won't be respected.

Limitation: the whole thesis has been realised with the left arm of the simulated NAO has the right arm is situated in the negative area of the reference frame. Which means, with this kinematic, it is subject to errors of trajectory: some joints results being aberrant (above maximum joint angle).

4.1.3 Results and Analysis

Several tests were done and a final version of the prototype 1 was developed to work in two situations: one to work in the simulation, and an additional one meant for extended use in a real situation using a real camera (Appendix 7.1).

The tests conducted to perform the visual processing will be first presented, and the liability of the distance assessed. This will be followed by tests to verify the kinematic reliability and assess its limits. Finally, the full prototype, using the simulated camera to extract an object's position in space and reach toward it will be presented.

Visual processing - Selecting the Method for feature extraction

The distance was always a bit off while using the circle extraction method, as can be seen in table 4.2 and Figure 4.5, therefore an ellipse extraction was implemented and tested. The results of these two methods were compared: both methods were quite similar, the circle method directly extracts circle like forms from the picture and estimates its radius, while the ellipse method extracts the elliptic closed contour on the picture and estimates its width and height.

These two methods were implemented and compared because the radius extracted from this detection determines the distance estimation, therefore an accurate radius is important.

To ensure that neither the Focal length nor the calculated ball size were the reason for this distance error, a simple check was performed.

It is known that:

$$\frac{Radius\_px}{Focallength} = \frac{Radius\_mm}{Object\_Distance\_mm} \tag{4.5}$$

This formula allows to extract two measurements from the Table 4.2 below and compare them.

Both sides of this equation should be, but are not, balanced. Thus, since the distance is given by the simulation and the ball size in mm is known, the problem comes from the variable radius in pixel.

$$\frac{Radius\_px_1 * Object\_Distance\_mm_1}{Radius\_mm} = \frac{Radius\_px_2 * Object\_Distance\_mm_2}{Radius\_mm} \tag{4.6}$$

Real D(mm)	$D_C(mm)$	$D_E(mm)$	$C_R(px)$	$E_R(px)$	MSE C (mm)	MSE E(mm)
110	157.36		38		47.36	
120	152.94	139.23	39.1	42.95	32.94	19.23
130	149.87	139.23	39.9	42.95	19.87	9.23
140	148.02	146.96	40.4	40.69	8.02	6.96
150	156.54	154.84	38.2	38.62	6.54	4.84
160	163.84	161.49	36.5	37.03	3.83	1.49
170	163.84	170.068	36.5	35.16	6.16	0.07
180	180.66	178.40	33.1	33.52	0.66	1.59
190	186.29	186.00	32.1	32.15	3.70	3.99
200	192.90	193.34	31	30.93	7.09	6.66
210	204.79	200.80	29.2	29.78	5.20	9.19
220	207.64	207.49	28.8	28.82	12.36	12.50
230	213.57	214.64	28	27.86	16.43	15.35
240	220.666	221.81	27.1	26.96	19.33	18.18

TABLE 4.2: Comparison of 2 techniques to extract the distance from the picture, D means Distance, C circle, E ellipse and R radius. The last two columns are the absolute position error. The ellipse method is less permissive than the circle one and truncated balls are not considered by this function, MSE stands for Mean squared error

The table above shows no missing data with the circle method, which is more permissive, and truncated balls are still registered as circles and its radius calculated (wrongly). However, the ellipse method considers only the closed contour of a form, this is why the first line has no Ellipse radius and distance.

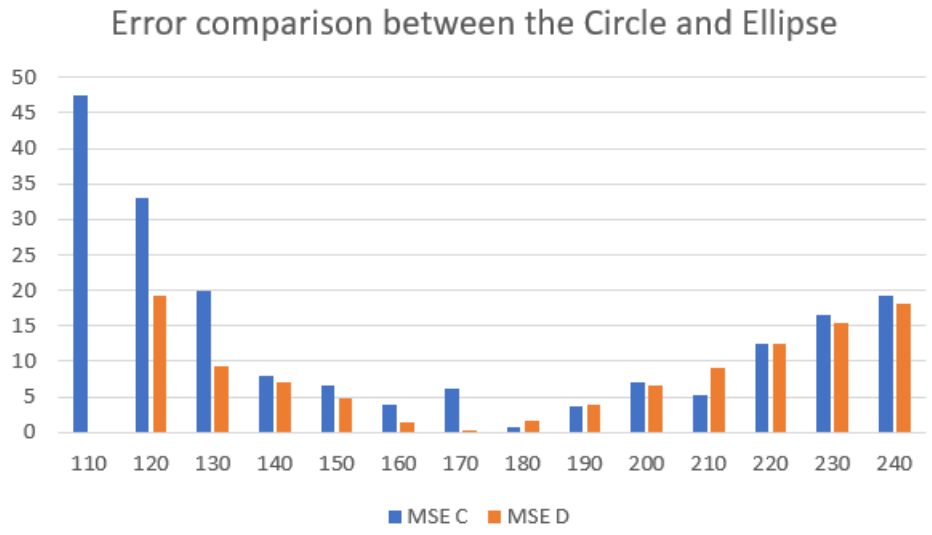


FIGURE 4.5: Error comparison of the Circle and Ellipse methods of radius extraction.

The graph 4.5 highlights the two last columns (MSE C and MSE E of the Table), where the circle method has, in most cases, more errors than the ellipse method.

	Circle	Ellipse
Radius liability (is the radius size always constant when the ball is at a set distance)	Constant	Constant
Radius liability (mean MSE) (is the distance correctly estimated)	13.54mm	8.41mm
Biggest position MSE (in NAO reaching area)	47mm	19mm
Missing data (is the detection permissive -account for truncated ball-)	No	Yes

TABLE 4.3: Observation on Circle/Ellipse methods of radius extraction

Having the least amount of errors and the most reliable distances possible in NAO reachable zone (110-218 mm) is important to reach the correct position. The ellipse method fits those requirements better and was therefore chosen.

This covers the error in the X axis, in the Y axis, the pose error didn't exceed 2.5cm in the reachable zone (with a mean error of 1.4mm over 20 runs).

The Z position being fixed, all the objects are considered to be on the table at a defined height (see setup Figure 4.6a).

Limitations: both methods of feature extraction are a bit unreliable where gauging the ball's distance at the limits of the reachable zone. This means that NAO may try to reach unreachable balls.

Kinematic test: Reaching the object position with the kinematic

All the following experiments were conducted with the same speed and acceleration (speed = 5 rad/s and acceleration = 50 rad/s^2) in order to have a substantial feedback from the joint sensory movements over a period of time, the reaching motion is executed in less than a second

The robot starts up at the same home position at each run (as presented Figure 4.6a). Then, as soon as it receives the endpoint sequence, it effectuates a reach motion with its left arm towards the object. This test shows the robot setup and how the arm moves.

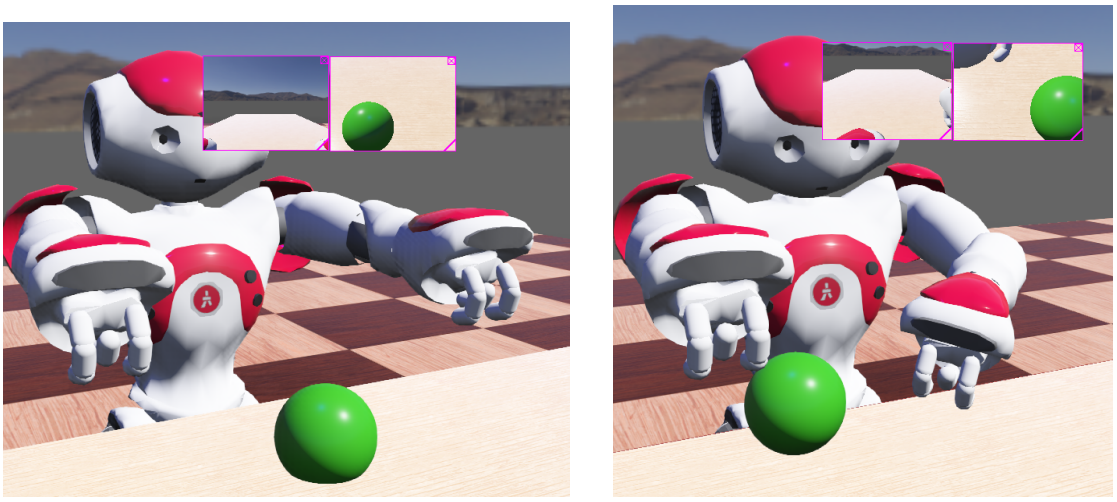
All positions of ground truth simulations to estimated co-ordinates are in NAO's torso reference frame. The starting pose will be the same during the whole thesis, the real object position in the simulated world is the ground truth and the estimated coordinates correspond to the visual processing estimated object position.

	Object position (Ground truth)	**Object position (Estimated coordinates)**	**NAO left arm starting pose coordinates**
Position x (mm)	180	192	218
Position y (mm)	31	30.6	113
Position z (mm)	1	1	87

MSE position:

$$\sqrt{(180-192)^2+(31-30.6)^2} = 12.01mm \tag{4.7}$$

The joints are all set to zero, 0 (with an error of 0.04rad) at the starting pose. In the figure 4.6a, NAO is in its starting pose and the end pose is depicted in figure 4.6b.



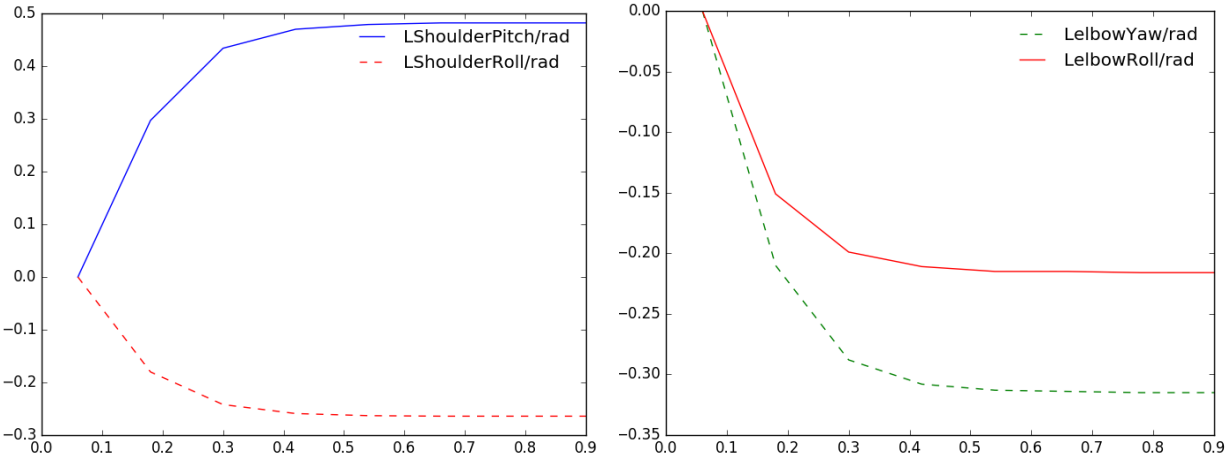
(A) NAO's position before reaching

(B) NAO's position after reaching

FIGURE 4.6: NAO's motion in prototype 1, before and after reaching toward the object

The hand position in figure 4.6b is almost spot on with an MSE error of 12.01 mm (less than half the ball radius) and manages to displace the object, which proves it had made contact.

On figure 4.7, the motion of the shoulder and elbow joints (in radiant) is reported according to time (in s). The movement is fluid and executed in less than a second. This metric is relevant to use for the rest of the thesis as the healthy/PD motion can be compared by observing the joints radiant.



(A) Graphic of the shoulder Pitch and Roll in rad/s

(B) Graphic of the elbow Yaw and Roll in rad/s

FIGURE 4.7: Graphic of the shoulder and elbow joints in radiant according to an object position in the simulator. the axis are in rad/s.

Figure 4.8 shows the result of a grasping motion in NAO. The robot opens and closes its hand 5 times: all the joints of the hand followed the exact same pattern. 3 joints are represented here.

It can be observed that there are not many points forming this plot. This is because to open and close the hand, a delay of execution of several milliseconds was implemented. During those delay gaps, no data was extracted and the code was paused. This plot has to be improved and figure more data to be useful.

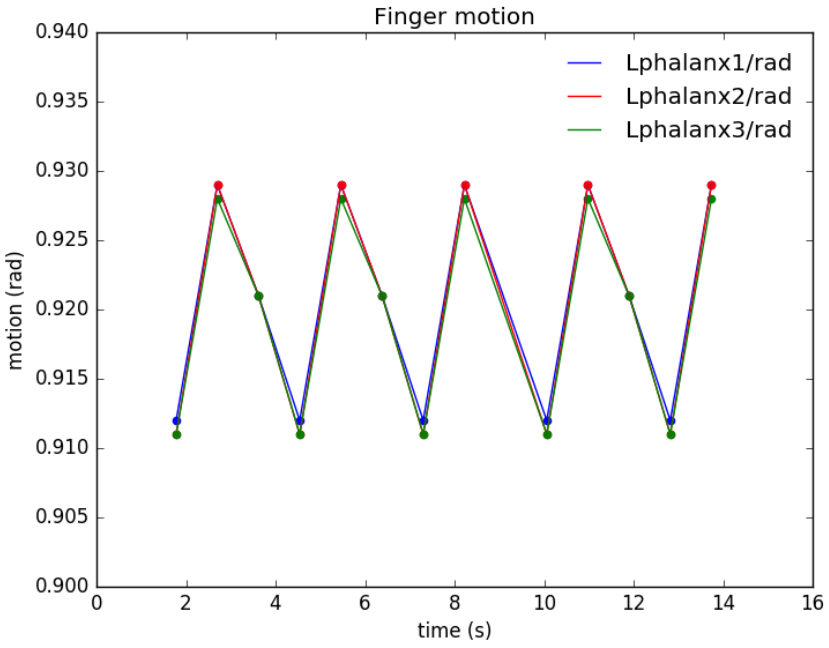


FIGURE 4.8: Graphic of one finger joints opening and closing hand 5times.

The kinematic displays some limitations, it was pointed out that the motion is aberrant when the end point position is out of Nao's left arm workspace. Some motions are also off path when the y or z axis are negative (see Results and analysis of the Appendix 7.1 to visualise those errors). These configuration errors did not appear when reaching for objects on the table within the left arm positive zone.

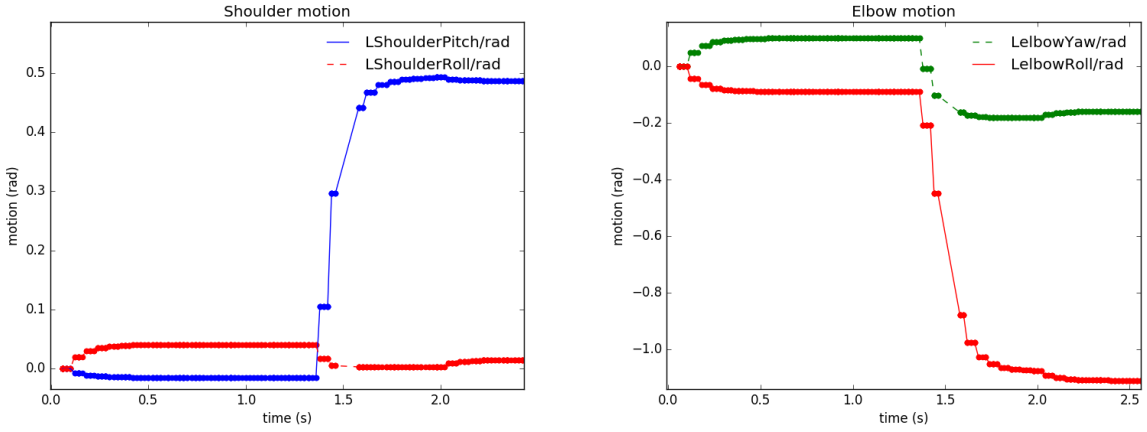
Full test of prototype 1

	Simulation (Ground truth)	**Estimated coordinates**
Position x (mm)	150	145
Position y (mm)	10	12
Position z (mm)	12	12

Calculating MSE of the position:

$$\sqrt{(150-145)^2+(10-12)^2} = 5.0mm \tag{4.8}$$

The ball was placed at the distance specified above to test the reliability of the head motion ability to detect the ball and correctly estimate its position. As in the previous test, the MSE error shows that the pose estimation is highly accurate. The final position displaced the object, attesting a contact.



(A) Graphic of the shoulder Pitch and Roll test 3

(B) Graphic of the elbow Yaw and Roll test 3

FIGURE 4.9: Graphic of the shoulder and elbow joints according to an object seen by the simulated camera with an head motion.

When the set of graphs 4.7 are compared with the set of graphs 4.9, the second set shows a supplementary movement between 0.06 and 1.36 sec. The reason being a processing delay between the moment the camera image is taken, sent, processed and the moment the joints receive and execute the motion in the simulation. The first pose observed is the home position (also called the starting pose), executed because of the delay between the launching of the simulation and the reception of a new position.

The joints' angles corresponding to the ball's position (presented above) was registered by NAO at 1.38sec and finished its execution at 2.02sec. The time of execution (around 0.7-0.8sec) is the same as in the previous test. The program was made extremely reactive to new visual stimulus and a rolling ball can result in a new motion. Here a ball position was registered and sent at 2s, the arm moving slightly.

In the last two tests with the camera, the support of the ball was given a clear aspect, however this isn't necessary in order to extract the main features from the image.

Limitation: in this prototype, once the hand has reached the object, the ellipse detection method extracts the elliptic form of the back of NAO's hand as a point of interest and tries to reach a new non-existing ball. This could be avoided by using colour filtering which excludes any reaction to red forms.

The following prototype is intended to ameliorate the shortcoming of prototype 1 and complexify the system implemented here.

4.2 Prototype 2: Sensory mapping

4.2.1 Design

As defined in section 3.3.2 the goal of the second prototype was to determine which stimulus NAO should react to, and which stimulation intensities should be sent to the rat brain computational model.

The constraint of stimulation have been chosen to be :

- A part of NAO workspace,
- The colour of a ball (in simulation),

If the ball is blue or green and in NAO's left arm workspace then the model will be considered highly stimulated, else it will be poorly stimulated in a binary fashion. The stimulation levels were set at a low of 1.2mA and a high of 3.5 mA respectively and sent to the rat brain thalamus. The whole brain reacts to both stimulation in the healthy/PD case.

The prototype, and its requirements, will be presented here from the workspace implementation to the selection of the relevant regions to stimulate in the brain model.

4.2.2 Implementation

4.2.2.1 Workspace definition

The real workspace (x,y) of NAO is depicted on figure 4.10 by the blue area covering the left arm. This prototype workspace was a simplified version of the real one's.

FIGURE 4.10: top down view of the left arm workspace (x,y) (blue) [61].

In figure 4.11 and 4.12 the dotted blue lines represent NAO real workspace (schematically) while the black solid lines are the ideal workspace, that was implemented. The blue dotted outlined area is not a perfect arc because, while a human can move the left arm from the extreme-left-to-right without bending the elbow, NAO can't. NAO achieves this area of workspace by bending its elbow, and that too, only once the shoulder is at its maximum movement point.

NAO doesn't turn its head left to right (horizontally), only up-and-down (vertically), therefore only seeing the workspace in between its arm. In Figure 4.11 the ideal workspace encompasses an area that is not part of the real one, however as it is outside the visual range of NAO, it is not a problem.

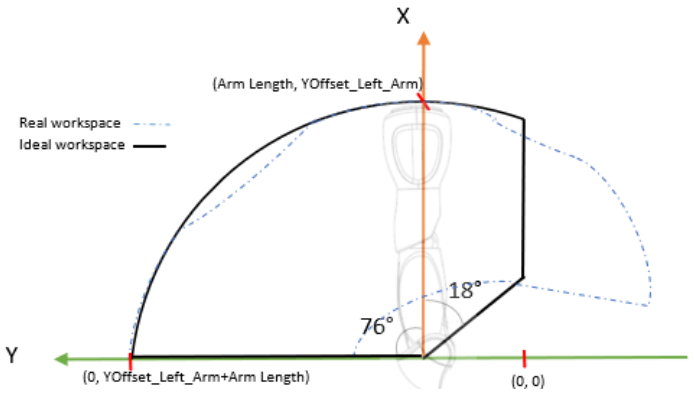


FIGURE 4.11: Ideal and real schematic workspace in 2D (x,y) of the left arm.

In the figure below and in the simulation, the ideal workspace has been limited to 180 degrees, as it is a given that NAO won't try to reach for something behind its line of vision An offset Y and Z can be observed between the frame of reference (centre of the torso) and the shoulder, represented on the figures by the (0,0) position on the axis Y and Z.

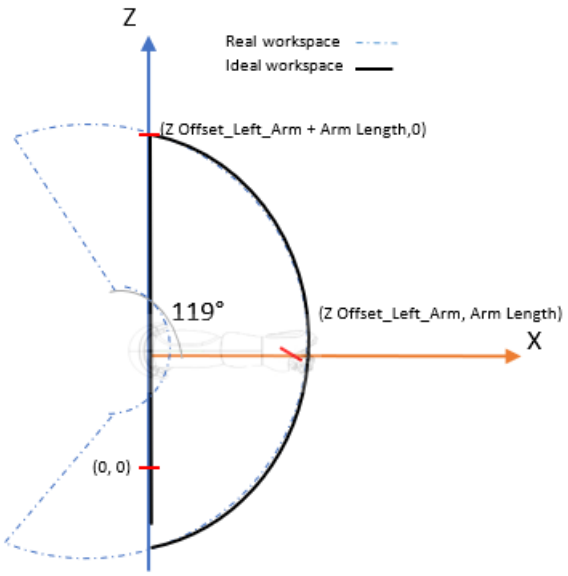


FIGURE 4.12: Ideal and real schematic workspace in 2D (z,x) of the left arm.

In the table below, the left arm measurements of a NAO H25V3.3 model are given.

Name	Size (mm)
Arm length	218.65
Left shoulder OffsetLY	113
Left shoulder OffsetLZ	87.31

TABLE 4.4: NAO's left arm length.

To define the ideal workspace, a distance limitation was implemented, such that the position of the object in X and Y and X and Z must be in arm length.

$$\sqrt{(X^2 + (Y - OffsetLY)^2)} <= Arm\_Length + OffsetLY \tag{4.9}$$

and

$$\sqrt{(X^2 + (Z - OffsetLZ)^2)} <= Arm\_Length + OffsetLZ \tag{4.10}$$

Moreover the object must be placed in the workspace arc defined with the following equations, based on Figures 4.11 and 4.12 and the further reachable position calculated with NAO's maximum joint angles (see table 2.1) with simple trigonometry.

$$asin(\frac{-67 - OffsetLY}{Arm\_Length}) <= asin(\frac{Y - OffsetLY}{Arm\_Length}) <= asin(\frac{331 - OffsetLY}{Arm\_Length}) \tag{4.11}$$

$$0 <= acos(\frac{X}{Arm\_Length}) <= acos(\frac{0}{Arm\_Length}) \tag{4.12}$$

and

$$0 <= acos(\frac{Z - OffsetLZ}{Arm\_Length}) <= acos(\frac{-Arm\_Length}{Arm\_Length}) \tag{4.13}$$

$$asin(\frac{0}{Arm\_Length}) <= asin(\frac{X}{Arm\_Length}) \tag{4.14}$$

Also, X alone must be inferior to the arm size and in order to avoid any minimum values Y must not be inferior to 0. Let's note that Z can still reach negative values in the ideal workspace.

4.2.2.2 Colour filtering

In Prototype 1, a shape extractor to determine the shape of the object was implemented. However having a shape extractor instead of an object recognition capability leads to an issue: the back of NAO's hand was detected as elliptical and its position was sent to the kinematic. To solve this a colour filtering (with 4 colours recognition) was implemented.

The three-colour channels (RGB) were extracted from the simulation camera using a Ros topic, then, using the elliptic coordinates given by the elliptic form recognition, the image is resized to focus on the upper half of the ball and its average colour is determined. This colour is then compared to the palette below.

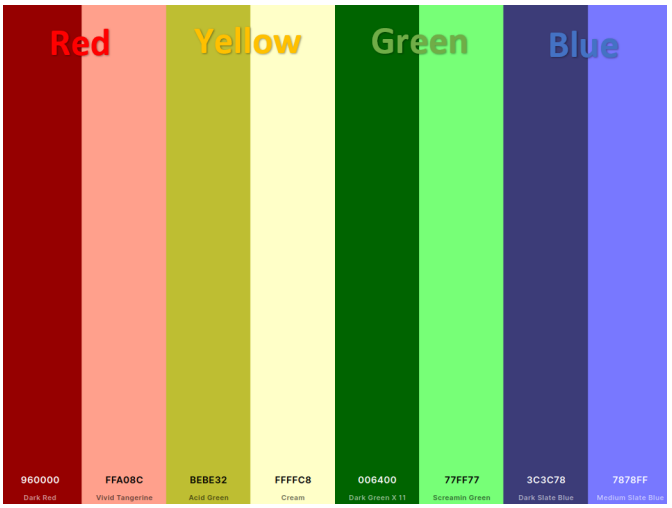


FIGURE 4.13: Colour palette (max and min shades) on 4 colours for ball detection.

Each maximum and minimum values were selected considering any possible overlay, see figure 4.14 . A colour can be defined by three channels Red-Green-Blue (RGB), each channel value defines the intensity of each colour in the final observable one.

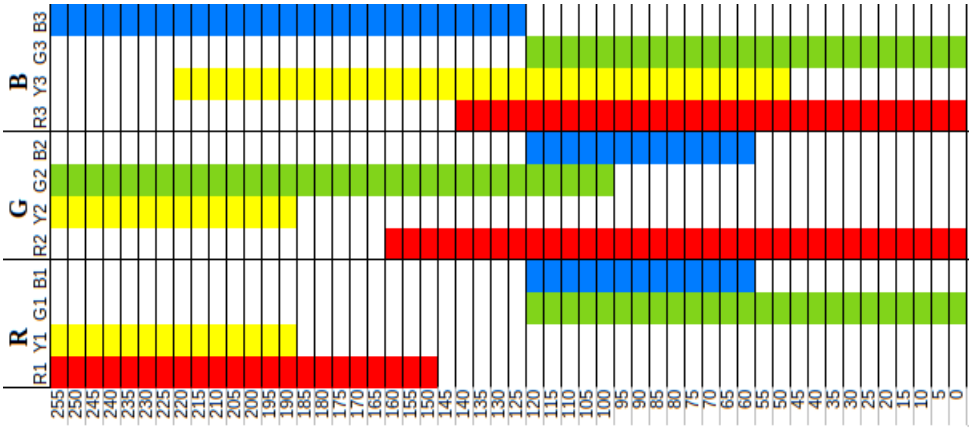


FIGURE 4.14: Colour palette range in each channel RGB.

If there is no overlay between the colour in, at least, one of those channels, then there will be no conflict between them. The maximum and minimum shades can be observed in Figure 4.13, the values of their RGB channel and their range is depicted in the Figure 4.14 .

If the colour detected is red, yellow or not identified, the robot won't be stimulated. If it's green or blue, it will be stimulated and try and reach the ball.

Limitation: the lighting of the environment is important, the image must be well lit: in case of low-light the ball's colour may be wrongly identified.

4.2.2.3 Rat model stimulation

The question now is which brain area should be stimulated and which signal has to be sent to stimulate the rat's brain. In the literature review section 2.1.5 page 18 the thalamus and primary motor cortex seemed to be the most relevant choice where to send the stimulation in a computational model. The model used for this thesis was based on Kumaravelu & al (2016) [36] work and realised by Jhielson M. Pimentel with Neuron NetPyNE [62]. The rat model realism will not be questioned in this thesis.

An additional feature was added to the computational model to assure the repeatability of the results: saving the seed. A random seed regulates the comportment of the rat model brain, at each run a new one is generated. In this thesis the seeds used for each test have been saved.

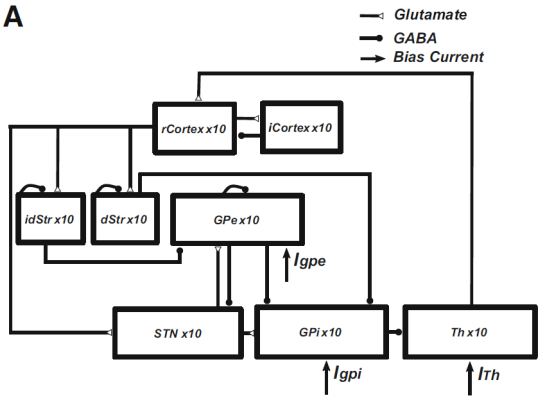


FIGURE 4.15: Cortical-basal ganglia-thalamus network schematic model showing connections within the network. The Glutamate are excitatory connections and GABA are inhibitory [36].

In this model, the thalamus and r-cortex (Regular Spiking cortex motor region) may be appropriate regions to send an input to (see the literature review section 2.1.5). However, only the thalamus receives a binary stimulation in this work. Due to time restriction all the possible configurations could not be tested.

Based on the work of Jhielson M. Pimentel, the optimal stimulation to have ideal motor cortex and other brain regions answers were defined to be 1.2 mA (low stimulation input) and 3.5 mA (high stimulation input). These inputs will be referred to as non stimulated/stimulated cases.

According to Jhielson M. Pimentel research [62] the r-cortex has the following response depending on the stimulation sent to the thalamus:

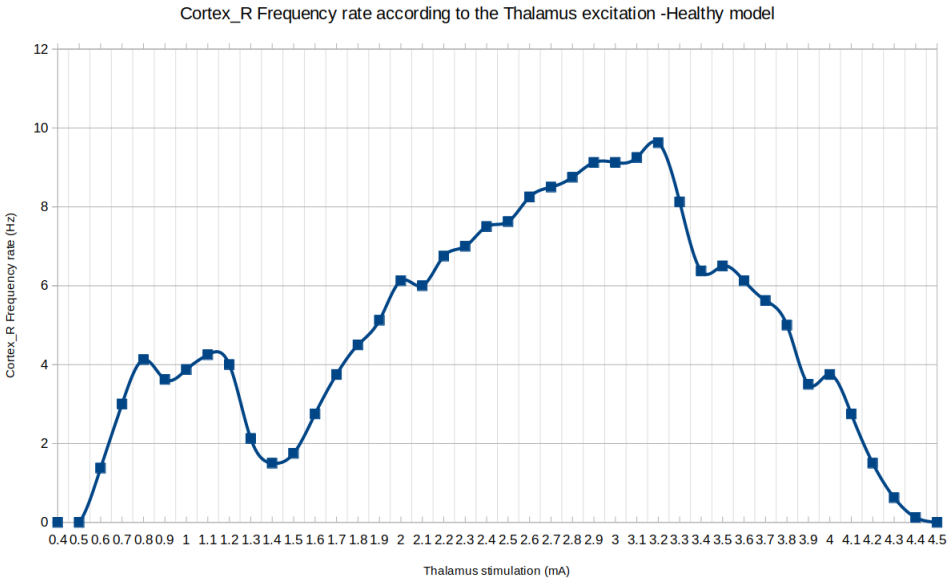


FIGURE 4.16: Cortex-R answer to Thalamus stimulation with an healthy model, graph based on M. Pimentel's work and random seeds.

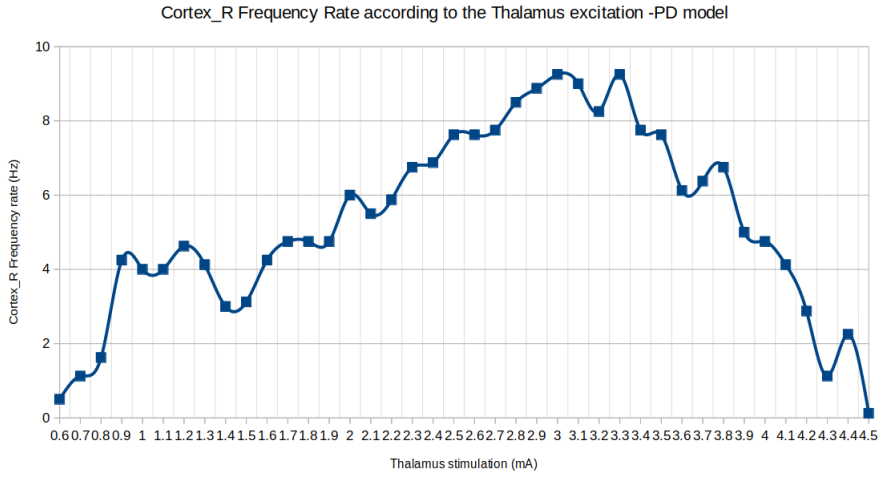


FIGURE 4.17: Cortex-R answer to Thalamus stimulation with a PD model, graph based on M. Pimentel's work and random seeds.

It can be observed that the cortex-r as an increased excitation until around 2.9 to 3.5 mA in both a healthy case and a PD case. However in a PD case the growth pattern is more jagged than that of the curve of a healthy case. As they are either higher or lower than the general tendency, it can be surmised that there is a higher variance and instability in the PD computational model answer to the same thalamus stimulation.

The brain answer is coherent with results observed in a real rat and the stimulation answer is significantly greater with a 3.0 or 3.5 mA input than a 1.2 mA input. Both states (stimulated/non stimulated) have distinct brain response.

4.2.3 Results and Analysis

It is necessary to confirm that:

- Prototype 2 can differentiate balls and send a different signal according to the situation.
- Different inputs generate different outputs exploitable in the next set of prototypes.

Workspace and colour discrimination implementation

To verify the results of this simulation, each position associated with a high stimulation (interesting ball) and the angles sent to the robot joints were saved in a separate file. This allows to verify the accuracy of the ball's position, and the resulting joints' command.

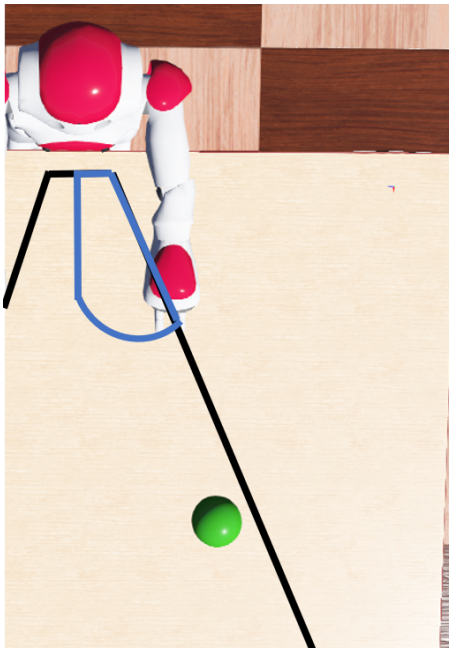


FIGURE 4.18: Simulated NAO visual range (in black) and schematic final reaching area (blue).

In the figure 4.18 the area outlined in a solid black line is the zone within which NAO can see by moving its head up and down. The area outlined with a blue line is the actual reaching zone of the robot's arm (resulting from the ideal workspace implementation and visual limit of the robot).

After several tests, no discrepancy zones were observed on the outskirts of the workspace.

	Wrong estimation probability	Real position (mm)	Estimated position (mm)	Object correctly reached
Pose Error in the Y axis	10% (10runs)	-10	$\simeq 0$	True
Pose Error in the X axis	80% (10runs)	$\simeq 219 - 230$	$\simeq 207 - 215$	False
Pose Error in the X axis	50% (4runs)	$\simeq 230 - 240$	$\simeq 215 - 220$	False
Pose Error in the X/ Y axis	0% (20runs)	$\simeq 110 - 219$/ $\simeq 0 - 60$	$\simeq 110 - 219$/ $\simeq 0 - 60$	True

TABLE 4.5: Results of the ball pose estimation within and outside the workspace

Balls in between 219-240 are outside the workspace but may still be considered as within the workspace. However, no aberrant angles were observed: the robot reaches for, without touching, the ball as can be seen in the Figure below. The given ball pose is in the workspace, even though the ball is actually beyond the reach of NAO's arm. No ball pose were incorrectly estimated within the workspace.

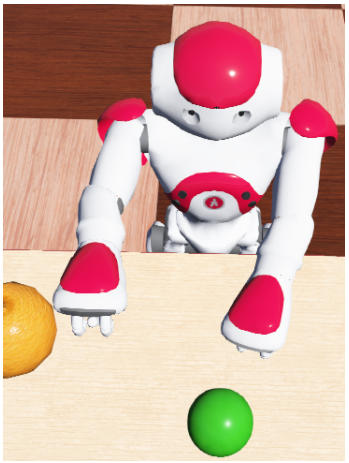


FIGURE 4.19: NAO reaching a ball out of reach. The ball was not touched and the arm presented a coherent position according to the real object pose.

The colour of the ball was well estimated 100% of the time. Parasite circles (when the hand is seen as an ellipse) were not deemed interesting.

Rat model stimulation

The stimulation was sent to the rat computational model, as per the estimated ball pose and colour. The rat model brain's regions show different outputs in accordance to the

input stimulation and Parkinson presence, or non presence. Results are presented table 4.6.

Once the stimulation received by the rat model, it starts being gradually excited (see Figure below). The following firing rates were obtained with the same seed (194.68). There is a clearly differentiated cortex-r response between the 1.2 and 3.5 mA stimulation input and a slight one between the PD and healthy case. In this thesis it was chosen to compare stable firing rates between them: the outputs of the computational model are extracted at 1000ms, when all the firing rates begin to stabilise.

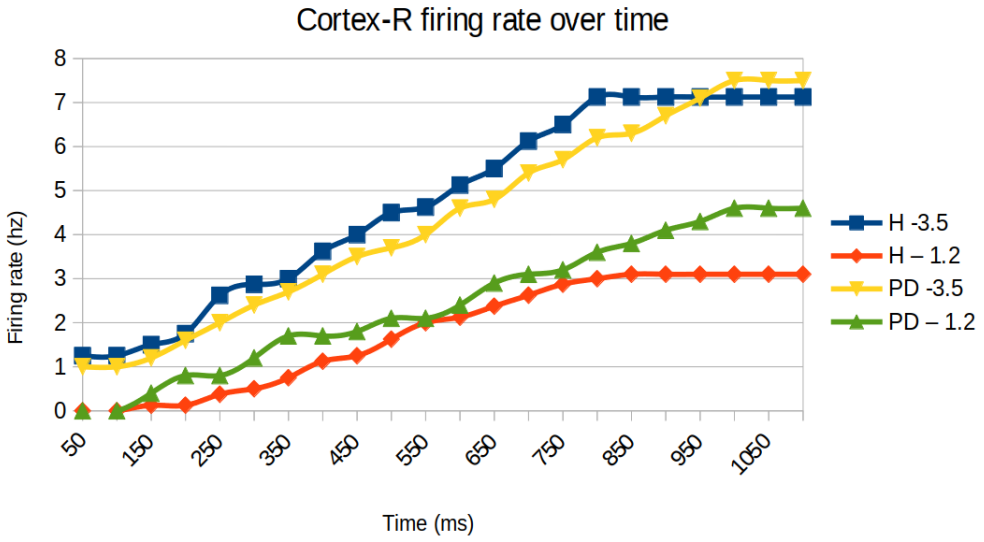


FIGURE 4.20: Cortex-R firing rate over time.

In the table below, PD stands for Parkinson Disease and H for Healthy. Each brain's region's population spiking rate (see Figure 2.21 in section 2.1.6 of the literature review for an explanation on spiking rates) were extracted to assess the pertinence of the rat model in studying PD.

stimulation (mA)\ model response (Hz)	H – 1.2	PD – 1.2	H - 3.5	PD - 3.5
StrD1 (Hz)	0.20	0	0.32	0
StrD2 (Hz)	0.10	4.21	0.32	6.69
TH (Hz)	26.09	19.89	31.40	49.92
Gpi (Hz)	24.17	35.78	24.45	44.82
Gpe (Hz)	35.52	22.39	37.15	20.22
CTX_RS (Hz)	3.46	4.30	6.43	7.95
CTX_FSI (Hz)	1.73	3.33	5.08	7.40
STN (Hz)	11.19	17.79	10.58	27.15

TABLE 4.6: answer of the cortex and other regions according to the stimulation and the presence or not (Healthy H) of Parkinson (PD). This table is the result of the average population frequency after 5 runs with random model seeds.

It can be observed that the difference of frequency rate between an healthy and PD brain model corresponds to the classical model of the brain comportment in both cases. (To compare to section 2.1.4.2 and Figure 2.5 of the literature review to recall the difference in comportment between an Healthy and PD basal ganglia). If the brain region has an higher firing rate and it is:

- excitatory (as the cortex-R), then the brain regions linked to it will be more excited,
- inhibitory (as D2) then it will inhibit more.

The stimulations, 1.2 and 3.5mA, offer marked difference between the PD and Healthy case, when comparing brain regions and in particular the Cortex-R region firing rate.

There is a resulting output that allows a distinction between the stimulated and non stimulated state and follows the observations made on PD basal ganglia comportment in the literature review.

It was chosen to extract the brain's outputs out of the Cortex-R, but the choice of the output neural coding and what to do with the brain's region's response will be discussed in prototype 3.

4.3 Prototype 3 and 4: motor control implementation with a healthy/PD neural activity

4.3.1 Design

As defined in section 3.3.3 and 3.3.4, the third and fourth prototype will present the mapping between the computational model and NAO for an healthy and PD brain model. The outputs of the rat model had to be extracted and treated in order to be sent to NAO, as noise in its reaching motion, inducing kinematic tremor.

To achieve this prototype it was necessary to know:

- From which brain area to decode data;
- the neuronal decoding method;
- how to treat this data once extracted;
- how will this data be sent to the robot.

The data extracted were:

- the mean firing rate;
- the mean Interspike Interval (ISI);
- and the ISI's standard deviation (STD) of the Cortex-R.

These data, once extracted were sent through an artificial neural network to determine the state of the neural activity (stimulation/is there PD or not). The stimulation has to be a binary response, 0 (non stimulated) or 1 (stimulated) in reaction to a 1.2 or 3.5 mA input. The PD state has to be a probability going from 0 (healthy) to 1 (PD).

Then the probability value is tempered in order to obtain a noise between 0 to 0.20rad (NAO's "muscles" are in radiant), This transformed probability is sent to the joints as a maximum/minimum noise scale (as positive and negative value). Each joint receives a random perturbation within the noise range at each sub-motion (the trajectory is sequenced into sub-motions).

Owing to space constraints and to maintain clarity, only the best options from the tests, the one implemented, will be presented here. However, the other options are available in the Appendix 7.2.

4.3.2 Implementation

4.3.2.1 Extracting and decoding the rat model outputs

As defined in the literature review (see section 2.1.5 of the literature review), the models presented consider that the motion order comes mainly from the motor cortex. The information was therefore extracted from the Cortex-R.

Once the extraction area defined, the question was which decoding method should be used, how should the data be treated and what kind of values should be sent to the robot.

The robot had to receive a certain noise level from the computational model: it should be quite low for a healthy case and relatively high for a PD case.

A choice had to be made between extracting only a population spiking average (firing rate) or a combination of the firing rate average plus individual/population spiking timing information.

At first a simple Kmean cluster (a classification machine learning method) had been implemented to determine whether using only the mean firing rate of the Cortex R would be sufficient, as this is the easier value to extract.

In Figure 4.21 (The X axis depicts the Healthy (0) and PD (1) cases), the stimulation, high in red and low in blue is well distinguished, however Healthy and PD states can't be differentiated with only this data. The clusters (Healthy/PD) corresponding to the same stimulation are superposed on the y axis. To only use the average spiking to differentiate the PD from the healthy case not being possible, it was necessary to use the cortex-R's Inter Spike Interval (ISI) too.

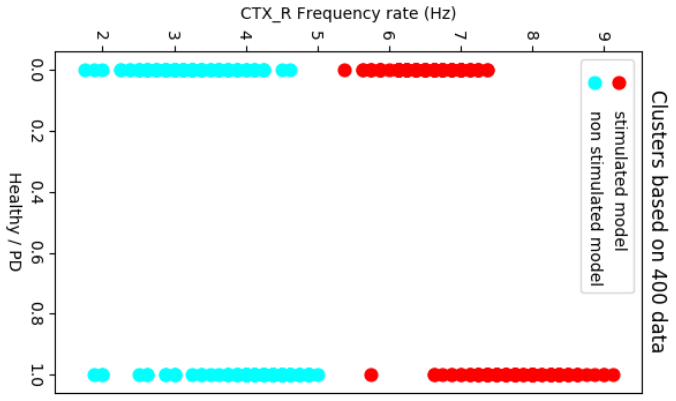


FIGURE 4.21: Cluster of cortex-R frequency rate stimulated/non stimulated with/without PD. The cortex frequency rate is higher when stimulated -3.5mA- and lower when non stimulated -1.2mA- The healthy model is on the left and PD model on the right.

The average Inter Spike Interval (ISI) and it's standard deviation (STD) were also extracted from the rat model and added to the kmean cluster as represented on Figure 4.22.

The non-stimulated Cortex-R shows a higher ISI average and a lower standard deviation of ISI. An higher ISI average means that there is a greater delay between each new firing, a lower standard deviation, means that the firing delays are more clustered around the mean firing interval (in ms). Those two information denote that there is less spiking and that this spiking is more regular when the rat model receives a 1.2 mA stimulation than when receiving 3.5mA.

The differentiation between the Healthy -0- and PD -100- case on the y axis seems to be more distinct. However, the clusters are still hardly able to differentiate between stimulated/non stimulated data and Healthy/PD data (without telling them the state of the model -PD/H- as it is in Figure 4.22).

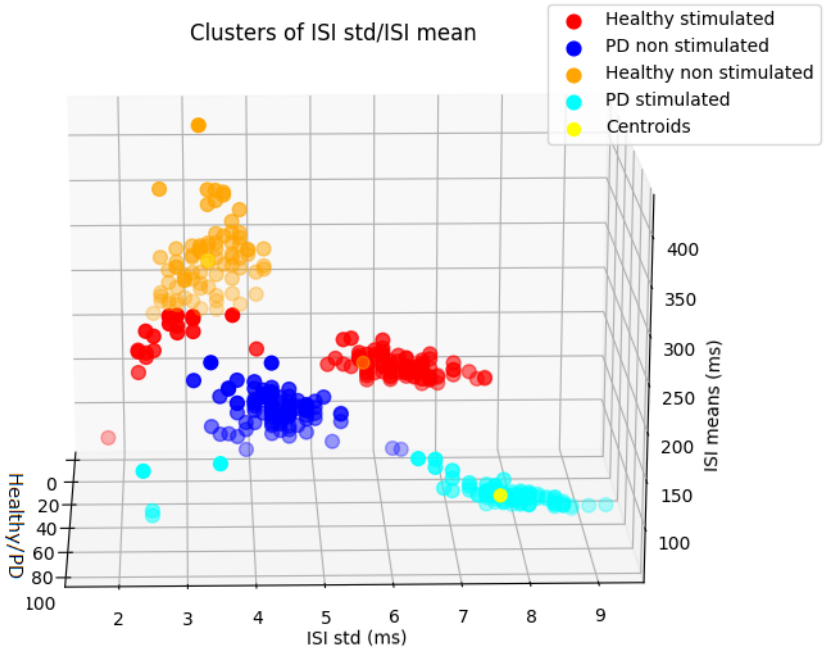


FIGURE 4.22: Cluster of ISI mean and ISI STD of the CTX-R for a stimulated/non stimulated model with/without PD, based on 400 data.

So to differentiate between PD and Healthy cases, a flag could be used to express whether or not the model has Parkinson. However, it is not really natural and may hinder further studies that wish to use this work to test assumptions on the brain model and see whether or not changing a parameter would stop the arm tremor.

Therefore, diverse machine learning algorithms were tested in order to obtain a good differentiation of the 4 cases (Healthy/PD, Stimulated/non stimulated).

It was decided to determine the stimulation and the probability of having PD out of the least amount of features possible, so as to not slow down the computational model. The probability of having PD will be the source of the noise value sent to the robot. and the stimulation the trigger to initiate the motion.

Not knowing how to best discriminate between machine learning models, several have been tested.

To create these the python library sklearn, TensorFlow and Keras were used.

Only the implemented artificial neural network will be presented here. This was the hardest to tune, but the most promising one, giving a correct fit of over 80% with two labels (outputs) without overfitting. Any other machine learning models (Logistic Regression and Naive Bayes) will be further detailed in the Appendix 7.2.

Dataset presentation

The dataset used was composed of the following elements:

Cortex-R firing rate (Hz)	ISI mean (ms)	ISI std (ms)	PD flag	Stimulation flag

The three first elements were the inputs (features) of the Feed Forward NN and the last two elements were the expected outputs (labels).

The dataset was divided in the following manner (”stimu” stands for stimulation):

Set/Division	total	PD -stimu	H -stimu	PD -non stimu	H -non stimu
Training data set	1920	480	480	480	480
Testing data set	480	120	120	120	120
Validation data set	120	30	30	30	30

TABLE 4.7: ANN dataset presentation

The database was realised by letting the rat model runs with different random seeds (to have various results) a certain number of times in each state.

The parameters were adjusted by recurrent training on the data in order to optimise the accuracy of the Neural Network (NN). Some parameters were set manually and will be presented in table 4.8, the other ones were automatically set by the MLPClassifier class of Sklearn (as the batch size).

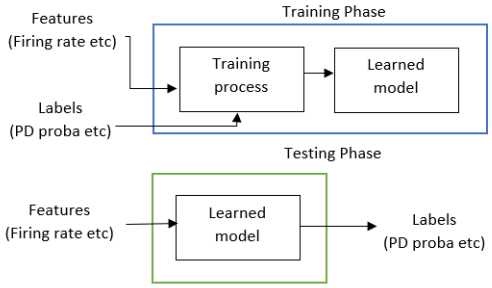


FIGURE 4.23: ANN training/testing phases presentation.

Set parameters	Values
Hidden layers.	3
Number of neurons	18, 8, 7
Activation method (hidden layers)	Relu
Learning rate .	0.001
Max iteration	200
Loss score tolerance	0.0001
Maximum n° of loss function calls	15000

TABLE 4.8: ANN parameters.

The parameter which sets the maximum number of loss function calls of the ANN will be briefly clarified here, the rest of the parameters won't be explained in this thesis (however the over-links in blue connect to short explanation in the appendix).

To assess the performance of the ANN, a loss function calculates the Loss score and the accuracy of the model and then modifies the parameters to minimise the first calculated element and increase the second. Once the loss score is under a certain threshold for a certain number of epoch, the training is considered finished. An epoch is the number of times the whole dataset has been used during training. A batch size is the number of training samples to test before updating the neuronal weights.

At each epoch, the loss function runs several times, the maximum number of loss function calls serve the same goal as the maximum number of iterations (the learning process is stopped if it's not improving by the time this maximum is reached).

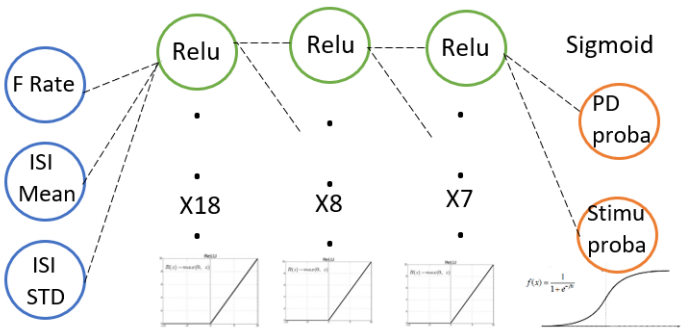


FIGURE 4.24: ANN composition, first layer are the inputs features, then there is the 3 hidden layers and finally the output labels.

The training of this Feed forward neural network takes a long time to compute. A python library, pickle, was used to store all the ANN information. Retrieving the ANN information from a file is much faster than to relaunch the whole training of the dataset each time the node is restarted. Also it ensures the ANN will always be identical at any given time. The downsize is, it will not be able to learn from new data anymore.

The results of the implementation will be presented in Result and analysis subsection 4.3.3.

Noise generation

Once, this Neural Network implemented and the PD and stimulation probability extracted, what is left, is to tune this data to send as stimulation and noise to the robot's joints. The question to be addressed here is how to treat this tuned-data once extracted.

The stimulation probability was simply rounded to the closest round value and used as a trigger to start NAO motion or not (0 no motion, 1 motion).

$$Stimulation = Rounded(Stimulation\_probability) \tag{4.15}$$

The PD probability was to be the noise sent to the robot and required more tuning. Through testing, it was asserted that the motion started to show tremor when an input of 0.05rad noise was sent to the joints (see next subsection 4.3.2.2 and Figure 4.30).

To scale down the noise formula, it was also necessary to determine an upper limit to the wanted tremor. The limit was set through empirical testing and NAO motion observation. The upper limit of the tremor can be tuned, to fit any experiment's requirements, by modifying the following formula, based on the mean probability of all the Healthy outputs:

$$Noise = \frac{PD\_probability - 3 * mean\_probability\_of\_Healthy\_cases}{3} \tag{4.16}$$

The Noise can only be positive or null, as the numerator is bounded between those boundaries by a python function. The algorithm would be:

if $PD\_probability - 3 * mean\_probability\_of\_Healthy\_cases > 0$ **then**
  $Noise = \frac{PD\_probability - 3 * mean\_probability\_of\_Healthy\_cases}{3}$
else
  $Noise = 0$
end if

The "mean_probability_of_Healthy_cases" corresponds to the mean probability obtained from all the results associated with a known Healthy flag.

The result and actual values of this implementation will be presented at the end of the Results and analysis section 4.3.3.

4.3.2.2 Motor noise implementation

To be able to determine what to do with the cortex signal, it is necessary to choose where it would be sent in the robot (taking into consideration factors such as speed, acceleration, muscle interference). Therefore, five different options were tested and an interference was sent to see the performance of the motion in each case. The movement had to be as natural as possible, to be observable in a Parkinsonian patient (have a physical meaning) and display a clearly different comportment, whether the computational model is Healthy or not, in order to allow further study with this metric.

- First Test - Affecting the velocity of the motion;
- Second Test - Affecting the acceleration of the motion;
- Third Test - Affecting both the acceleration and velocity of the motion;
- Fourth Test - Affecting the trajectory of the motion;
- Fifth Test - Affecting the motor signal of the motion;

The fifth test was selected and became the option to be implemented for two reasons:

- PD patients muscle control is affected, so adding noise to NAO "muscles" (joints) seems relevant,
- it presented the most pertinent results with the used metric (observing a speed difference in a motion graph in rad/sec is not as pertinent than observing joints noise in a motion plot).

The rest of the tests will be detailed in **Appendix. Noise affectation**.

One of the characteristic symptoms of PD is a resting tremor. However postural or kinetic tremors are not unusual, they usually have similar frequencies and lower amplitudes than the resting one [63, 64]. For this study a kinetic tremor was considered, wherein the muscle will be affected while moving towards the goal pose. To do so the first prototype had to be altered to send intermediary positions to the robot arm. The trajectory was divided into 16 sub-positions using the following formula (on x, y and z):

$$sub\_goal\_pose = previous\_pose + \frac{goal\_pose - previous\_pose}{number\_of\_sub\_pose} * i \tag{4.17}$$

with i going from 1 to the number_of_sub_pose.

The trajectory was chosen to be divided into 16 sub-motions to enable a completely fluid healthy motion. The more the trajectory is divided, the more subtle the interference in the motion could be. All the joint sequences of all the sub-motions were computed then sent in one go to the simulation.

In this implementation the noise received determines the maximum and minimum values of a scale. A joints error (disturbance compared to an healthy motion) can be any value in between the limits of the noise value. The error obtained is different for each joint and for each sub-motion and can be negative or positive. It is added to the normal motion to obtain the perturbed one.

The noise received must be between 0 to around 0.20 rad, over 0.25rad the visual motion of the robot becomes aberrant. 0.20 was deemed an acceptable noise value in the context of this thesis.

Finally, to better visualise the motion and better observe the perturbation, the joint speed was reduced to 2 rad/s and the acceleration to 30 rad/s^2.

Limitation: The implementation realised in this thesis is not reproducible on a real NAO as it is not possible to send noise directly to the real NAO's joints. However to obtain

a kinetic tremor in a real NAO, noise could be sent to the trajectory. The joints were chosen here to fit PD observations.

4.3.3 Results and analysis

ANN Results

The implemented configuration was the one shown in Figure 4.24, the results and curves of the ANN will be presented and fully explained below:

Metric	Results
Mean Proba of the Healthy flag	0.1335
Mean Proba of the PD flag	0.8139
STD Proba of the Healthy flag	0.2212
STD Proba of the PD flag	0.3031
Training accuracy	92.9%
Testing accuracy	93.1%
Validation accuracy	88.3%

TABLE 4.9: Results of the ANN implementation

The mean and STD probabilities were obtained from all the results associated with a known flag (i.e. healthy/PD).

The higher the accuracy, the higher the probability that the new inputs will be correctly estimated. The mean probability was extracted to observe if there had been enough differentiation between the PD and the Healthy cases and the std to determine if the probabilities are spread in a wide range or not (the lowest the better, as it would mean the data are closer to the average value).

In addition the loss curve of the trained model and accuracy curves of the training and testing set were retrieved to examine if the model was well build and if it could be improved.

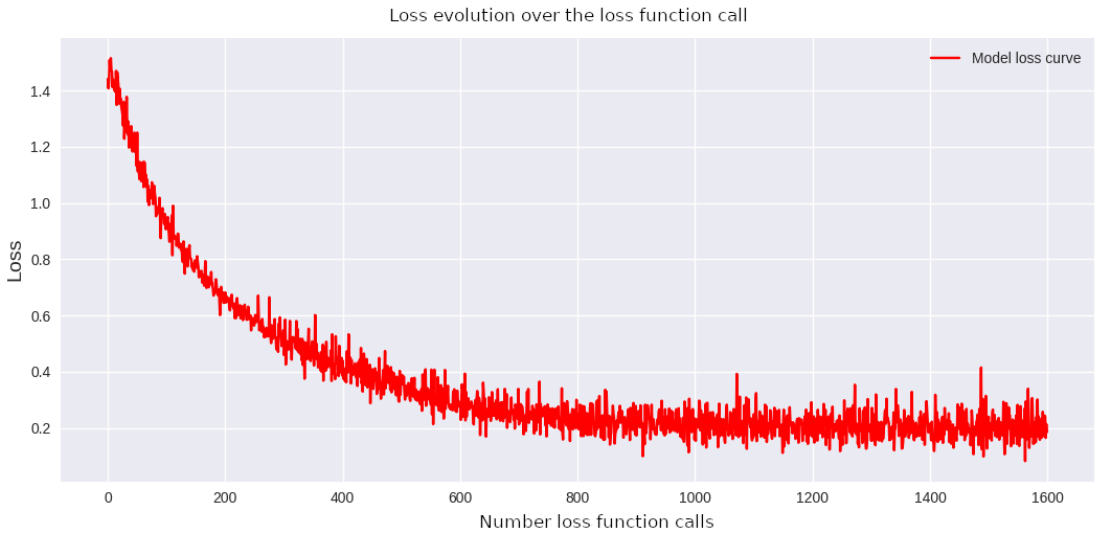


FIGURE 4.25: The ANN loss curve function looks acceptable with an appropriate learning rate.

According to Stanford CS class [65] a good learning rate should converge to 0 as fast as possible, if the loss curve converges to a high value (high being subjective, it depends on the requirements) then the learning rate is probably excessive, if the convergence has a shallow curve it means the learning rate is too low.

The loss curve of Figure 4.25 converges at 0.2, an acceptable result. However, according to this curve, the model can still be improved by further tuning the batch size and learning rate value.

The Feed Forward NN can either overfit, underfit or be well fit.

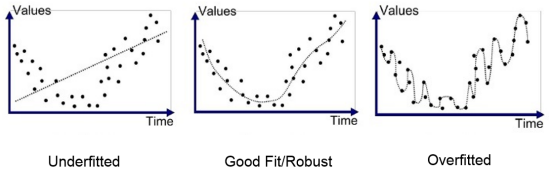


FIGURE 4.26: Example of overfit/underfit curves.

If the model is overfitted, it means it is overly adapted to the training set and is not generalised enough. It can be resolved by reducing the complexity of the system or collecting more data. If the model is underfit, it means it didn't learn enough from the training set and is unadapted to unknown data. Among the possible solutions, the model can be further trained during an higher number of epochs or further "complexified" in order to par this. The batch size and epoch also have an influence on the under or over fitting of the curve. If the number of epoch is too large the model will overfit. If its

too small it will underfit. The same goes about the batch size, if it's too large there is a risk of overfitting, if it's too small there is a risk of getting stuck in a Local minima. To verify that our model is not overfitting or underfitting it is interesting to test it with different batch sizes or epochs (cross-validation) to check if we really have a suitably-fit model.

The curve Figure 4.27 is ideal. Both the training and testing curves converge at a stable value at around 93%. The gap between them is small and the testing set has a higher accuracy. On the basis of this curve alone, the model may be considered well fit. However to be sure, more cross-validation tests should be conducted to confirm or invalidate this curve. It wasn't further assessed, considering the computational power needed and time required to assemble more data and test the model more in-depth. The outputs were mostly accurate (over 80%) and allowed a classification of the data as required.

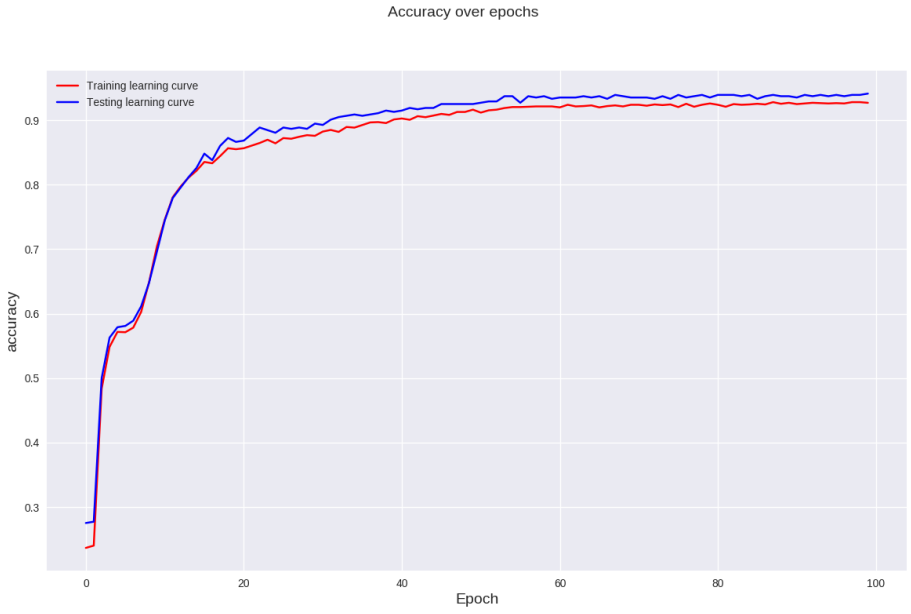


FIGURE 4.27: ANN accuracy over completion time, represented as epochs

Limitations: The model may be further improved with a more complex model (a fourth hidden layer, for example) and a bigger dataset. However, doing so would drastically increase the computational time required to run the model. Further improving the model with a fourth layer was not undertaken, as the actual model still has an acceptable accuracy and fits the need of this master thesis.

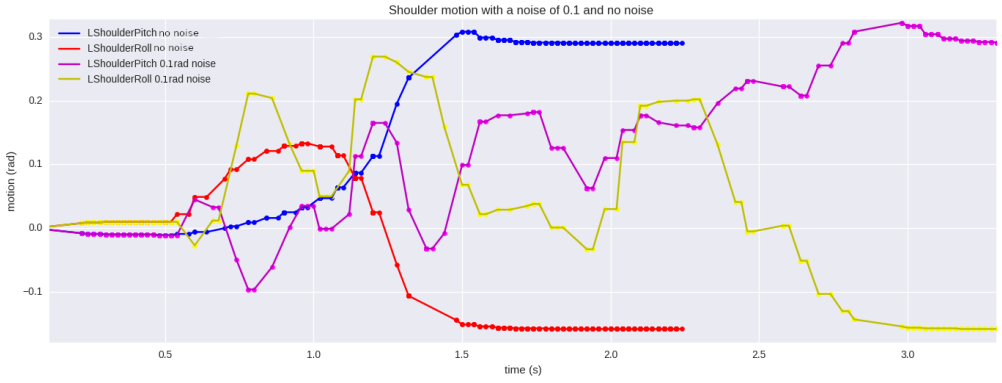
Extracting the model output

The control test is a test where the arm motion towards a green ball is performed without any error (no noise). The tremor test is the result of implementing a range -0.1 to 0.1 rad noise to the joints of the left arm.

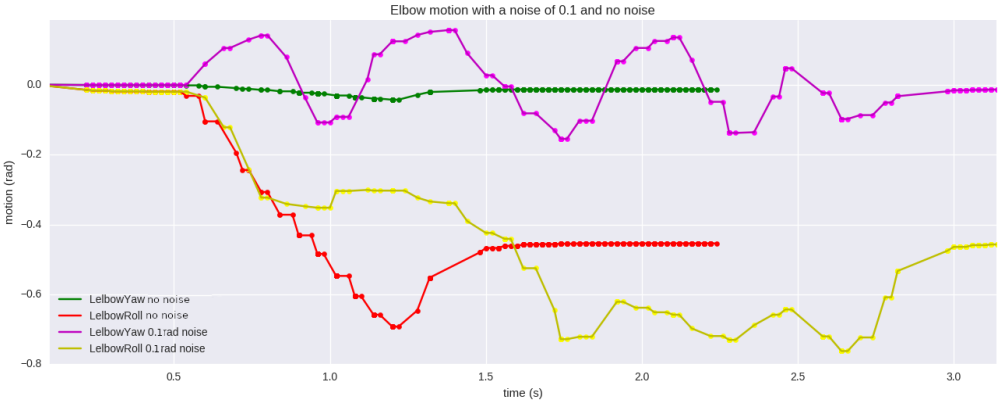
	Simulation (Ground truth)	**Estimated coordinates**
Object colour	Green	Green
Position x (mm)	196	195.7
Position y (mm)	15	14.4
Position z (mm)	5	5

Calculating MSE of the position:

$$\sqrt{(196 - 195.7)^2 + (15 - 14.4)^2} = 0.7mm \tag{4.18}$$



(A) Shoulder motion with a muscle error



(B) elbow motion with a muscle error

FIGURE 4.28: Shoulder and elbow motion with a muscle noise as the interference

The healthy comportment plot is comparable to the ones obtained prototype 1 (see Figure 4.9), with a slower and more decomposed motion than the ones in prototype 1. It should be noted here that the shoulder and elbow rolls have a motion exceeding their final position angle (at around 1s) before reaching their final position. This is due to the motion being divided into sub-motions. Visually, in simulation, the motion is fluid and non aberrant.

On the noisy motion of the graph (purple and yellow curves), each joint of the arm are visibly shaken. The elbow yaw even presented, a seemingly sinusoidal comportment comparable, on a bigger scale, to what Hess & al (2012) [64] highlighted in their experiment (see Figure 4.29). In their experiment, the displacement is in mm and on a resting tremor. In this thesis, it is expressed in radiant and as a kinetic tremor; at the end of the motion (once the goal reached) the tremor stops.

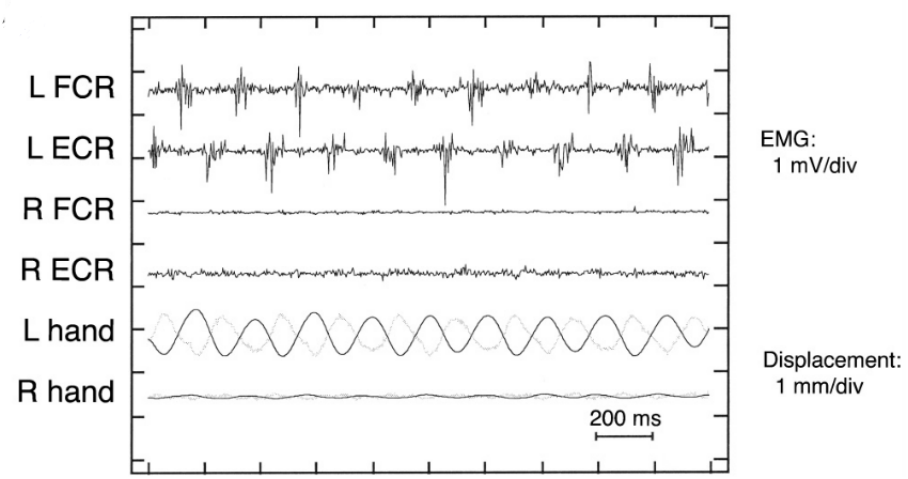


FIGURE 4.29: EMG and Movement Analysis of Rest Tremor in a Patient with PD. The first four traces represent surface EMG signal from forearm muscles; ECR stands for extensor carpi radialis and FCR for flexor carpi radialis. The bottom two traces reflect displacement (darker line) derived from accelerometry. In this case the Left hand is symptomatic [64]

The figure 4.29 shows the EMG of the left and right flexor carpi radialis (FCR) and extensor carpi radialis (ECR) (ones of the five main muscles that control movements of the wrist, and both do opposite motions). The arms are kept in a tense position (arms straight, before the patient), the left arm displays resting tremor while the right arm doesn't. The left forearm has a perturbed EMG compared to the right forearm with excessive noise and periodic augmentation of the frequency. An interesting thing here is, that when the FCR expresses itself too much, it corresponds to the ECR having the least noise and vice versa.

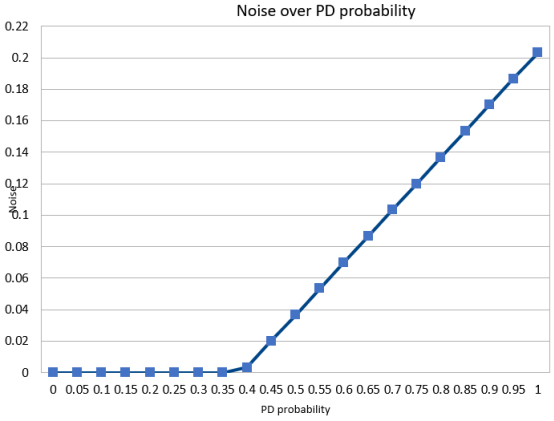
Next, the last two parameters of the figure show the displacement of the hands (mm) over time (ms) (in a resting position). It can be noted that the Right hand doesn't shows any displacement while the Left has a sinusoidal motion of around 2mm amplitude in a sinusoidal motion (in accordance with the FCR and ECR pics) [64]. This is a resting tremor, but it is known that a kinetic tremor should present the same noise frequency at a lower amplitude [63, 64].

Limitations: If the plot obtained through adding noise is compared to the joints in Figure 4.29, it is noticeable that the tremor's amplitude is much higher and frequency lower in graph 4.28. The observations realised by Hess & al [64] are not reproducible on NAO, as it has no antagonist muscle as a human has. However, the frequency of the tremor could be augmented by adding sub-motion and the amplitude diminished by reducing the maximum noise. Nonetheless, even if what can be seen on graph 4.28 is not 100% realistic, it is an exploitable tremor, allowing a clear differentiation of the Healthy and PD case and corresponding to general PD observations.

Healthy computational model noise to robot muscle

Here the result of the implementation of the equation 4.16 will be presented and explained through tests.

On Figure 4.30, the right panel associates the PD probability and the solution of equation 4.16 and the left panel presents the solution as a graph. The values were scaled with a resulting noise between 0 to 0.2rad, which is ideal, as a noise above 0.25rad displays aberrant arm motion. The Table and graph shows that the equation begins to generate a noise at a PD probability of 0.4. 90% of the Healthy cases PD probabilities were under this value (on 2230 stimulated run). The Healthy mean value used in the equation was 0.13.



PD Probability	Noise
0	0
...	...
0.4	0.00033
0.45	0.0200
0.5	0.0367
0.55	0.0533
0.6	0.0700
0.65	0.0867
0.7	0.1033
0.75	0.1200
0.8	0.1366
0.85	0.1530
0.9	0.1700
0.95	0.1867
1	0.2033

FIGURE 4.30: Noise over Pd probability resulting from the Noise formula.

A significant enough noise is generated after 0.05rad, which can be observed on a motion graph (see Fig 4.31). This noise is generated at a Pd probability of 0.55. The result will be presented below

Table corresponding to the object reached:

	Simulation (Ground truth)	**Estimated coordinates**
Object colour	Blue	Blue
Position x (mm)	186	184.9
Position y (mm)	25	24.8
Position z (mm)	5	5

Calculating MSE:

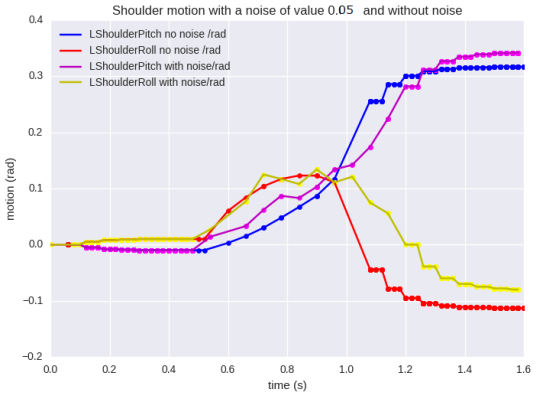
$$\sqrt{(186 - 184.9)^2 + (25 - 24.8)^2} = 1.1mm \tag{4.19}$$

Table giving the results (in this test the PD noise was fixed, not generated by the rat model):

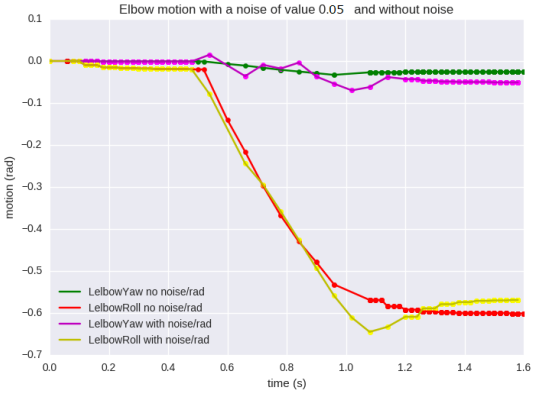
	Healthy	**PD**
PD probability	3,3%	55%
Noise	0	0.05
Mean Ctx-r firing rate	6.25	x
Mean ISI	152.18	x
STD ISI	47.98	x
Object Reached	True	True
Rat model seed	37.23	x

TABLE 4.10: Prot3: Table of results Healthy test minimum noise

In Figure 4.31 the normal shoulder motion is displayed by a blue line (Pitch motion) and a red line (Roll motion), while the perturbed motion is displayed by the yellow and pink curves.



(A) Shoulder motion with a minimal noise and without noise



(B) Elbow motion with a minimal noise and without noise

FIGURE 4.31: Visual presentation of the value at which noise appears in the motion

The Healthy motion was fluid, presenting no apparent tremor. The PD motion shows visible perturbation that still follows the healthy curve (while at noise of 0.1rad, the perturbed motion doesn't follow the healthy comportment pattern, see Figure 4.28).

PD neural activity consequences

The ANN accuracy of a PD evaluation was 94.8% (2750runs -Healthy/PD cases-). This percent was calculated by comparing the PD probability to the correct answer:

- if the probability of an Healthy case is above 0.4, it is considered an error,
- if the PD probability of a PD case is below 0.5 it is considered an error.

After 0.4 a positive noise is sent to the joints, below 0.5, the noise is not significant in the joint muscle.

On 2750 runs the PD probability accuracy, Healthy and PD well estimated according to the previous criteria was therefore 94.8%.

In PD cases, showing no sign of tremor is compatible with the physical observations of researches conducted before (see the Literature review section 2.1.2). A test was conducted to check the maximum tremor inputs, and another one to show what happens when a noise over 0.25 rad is sent.

	Simulation (Ground truth)	**Healthy/PD Estimated coordinates**
Object colour	Blue	Blue
Position x (mm)	180	184.8
Position y (mm)	31	30.9
Position z (mm)	0	0

Calculating MSE of the position:

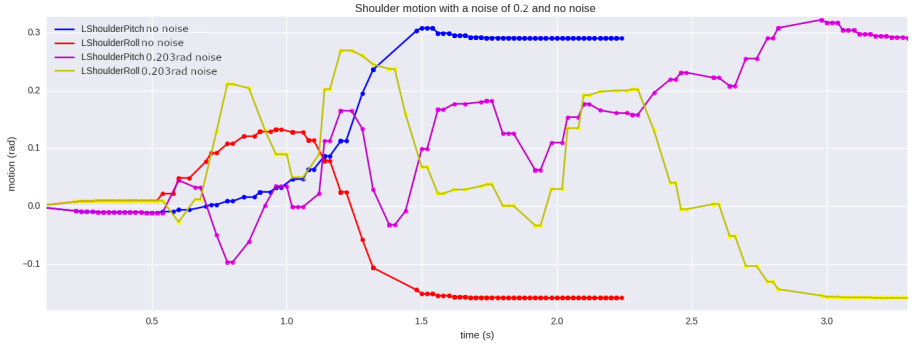
$$\sqrt{(180 - 184.8)^2 + (31 - 30.9c)^2} = 4.8mm \tag{4.20}$$

PD probabilities obtained in this test:

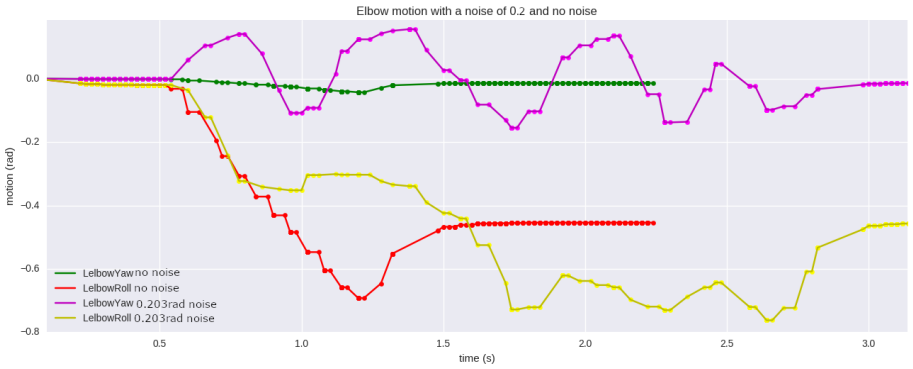
	Healthy	**PD**
PD probability	0.3%	99%
Noise	0	0.203
Mean Ctx-r firing rate	5.75	7.63
Mean ISI	154.49	123.72
STD ISI	48.38	22.31
Object Reached	True	True
Rat model seed	173.03	462.40

TABLE 4.11: Prot4: Table of results PD test maximum noise

The Figure 4.32 displays an healthy motion perfectly exempt of any noise and a PD motion with the biggest noise possible authorised by the formula 4.16 .



(A) Shoulder motion of an healthy and Pd model



(B) Elbow motion motion of an healthy and Pd model

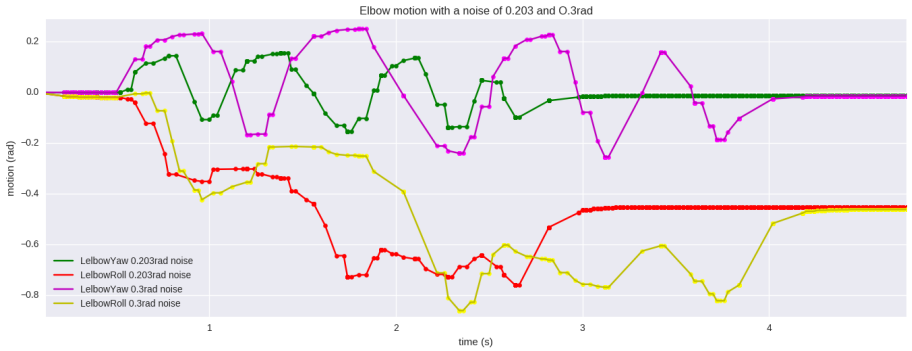
FIGURE 4.32: Comparison Healthy Pd motion, with both model well predicted by the ANN

A somewhat sinusoidal pattern can be seen in the PD motion- the chaotic motion is well visible and well distinguishable from the Healthy's one. The noise observed here, though having a bigger amplitude, is not incompatible with observations made on PD patients.

Let's observe on the plot above that the full reaching movement in PD takes about 600ms longer than for an Healthy movement (which is also in accordance with PD observed comportment (see Literature review section 2.1.5)).

The noise was limited to 0.203rad, to keep the motion realistic (i.e. absence of sudden and extreme change of direction). The following figure compares a 0.203rad tremor with a 0.3. It can be observed that not only is the 0.3. rad motion taking longer to stabilise (1 more sec than the 0.203rad) but also that the amplitude of the sub-motions are higher. Visually, in simulation, it shows large arm movements that seem to have an entirely different goal - trying to reach the sky, before going back on the table. This kind of motion is unnatural, and is therefore not relevant.

(A) Shoulder motion motion with 0.203rad and 0.3rad noise



(B) Elbow motion motion with 0.203rad and 0.3rad noise

FIGURE 4.33: Comparison between different noise input

4.4 Prototype 5: Implementation of the grasp motion with and without PD

4.4.1 Design

As outlined in section 3.3.5 the fifth prototype will describe a full reach-and-grasp motion with both, PD symptoms present and not present, subject to whether the brain model has PD or is Healthy.

Prototypes tests 1 to 4 used balls, however it was remarked in prototype 4 that a ball was not so easy to grasp as it would roll on contact. The ball was replaced by a cylinder and a circular top was added to cylinder (see Figure 4.34) to be able to lift it. The implementation of this section involved a remastering of the visual processing steps and of the kinematic, in order to achieve this prototype. The full communication between the nodes will also be described and a flowchart of the whole process be presented. In the Result and analysis section the comportment of the final version will be explained.

4.4.2 Implementation

4.4.2.1 Visual processing

New object parameters:

Object	Topped cylinder
Mass	0.5kg
Height	70mm
Cylinder Radius	10mm
Top Radius	15mm

The key objective of the visual processing is to obtain a stable result, which means that for a given distance (x) the radius of the object must always be estimated the same way, wherever it is on the y axis.

To use the same visual processing method as prototype 1 was giving a wrong radius estimation because the whole cylinder was identified as an ellipse. In and of itself, it would have been fine but the truncated cylinders were also accounted for and recognised as additional elliptic forms, which hindered the fulfilment of the key objective. Therefore to avoid this issue from happening, the base of the cylinder was erased from the image.

The limitations to this prototype's implementation were: the cylinder must be well-light (as mentioned in prototype 1) and, the light must come from above or behind the object (and not from the front). -The bottom of the cylinder was erased on the assumption that it would be darker at the bottom than the top. -

Once the elliptic top of the cylinder extracted, the image pass through a canny filter and the elliptic detection is applied, as represented in the figure below.

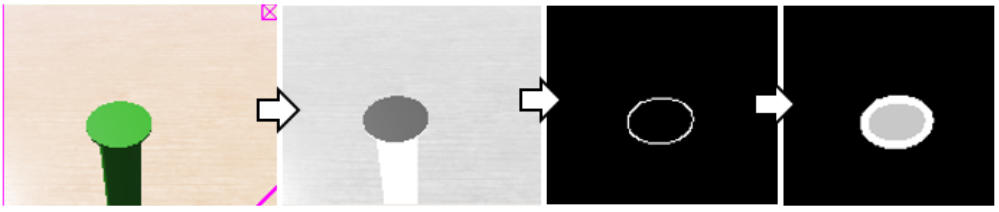


FIGURE 4.34: Cylinder visual processing steps: darker area cut/ canny filter applied / ellipse extraction

The rest of the visual processing is similar to what was explained in prototype 1 and 2.

As in prototype 1, the ellipse could be approximated into a single radius with the equation 4.2 as the width and height of the ellipse decreased proportionally as the object distance increased. The resulting radius is constant at a set distance.

However, the distance equation is fundamentally different. In prototype 1, the distance equation was based on the real ball radius, the focal length of the camera and the radius in px of the ball. These factors made the equation valid for any ball size, with the condition of modifying the real ball size parameter in the code. However, in this prototype, the distance equation is only based on the radius and a curve fitting was realised to find the most appropriate variables.

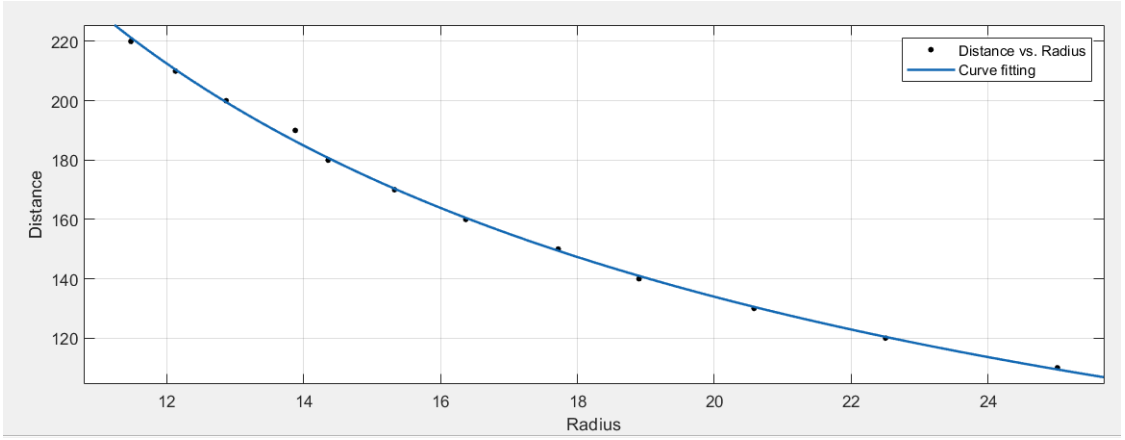


FIGURE 4.35: Curve fitting of the distance equation, in x the actual radius, in y the wanted position (realised on MATLAB).

In Figure 4.35, the x axis shows the radius extracted from the visual processing and on y the real distance to evaluate. The table 4.12 corresponds to this curve radius.

The distance equation resulting from this curve fitting was:

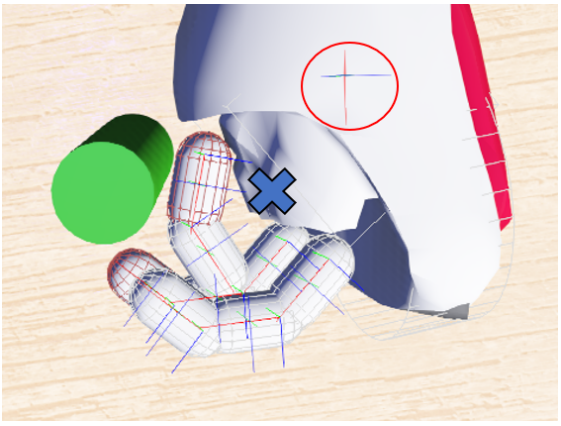
$$X = 2010 * object\_radius\_in\_px^{-0.9042} \tag{4.21}$$

The results of this equation implementation will be presented in the next subsection.

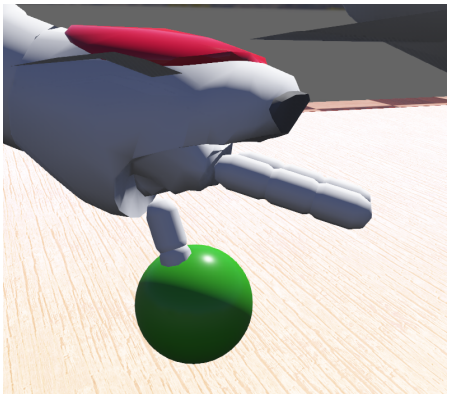
The Y position was still calculated with the same equation 4.4 as in prototype 1.

If the radius of the cylinder is different, but still circular, then a simple offset can be added to the equation to adapt it to the new object.

The kinematic calculates the motion from the shoulder to the centre of NAO's hand as can be seen on Figure 4.36a. The centre of NAO's hand is highlighted by a red circle and the centre of the fingers is represented by a blue cross.



(A) the kinematic centre of NAO hand is not the same as the centre point between the fingers.



(B) NAO reachs with the centre of NAO hands as reference, not from the centre of the fingers

FIGURE 4.36: visual presentation of the kinematic centre of NAO hand and the implication with an object.

Between the centre of NAO hand and the centre of the fingers there is a 2cm distance on the x axis and a 1.2cm distance on the y axis (see Figure 4.36a).

The X and the Y position of the cylinder sent to the kinematic has been a bit altered in order for the hand to reach the ball with the centre of his fingers. The position X received by the kinematic and the workspace verification is therefore:

$$X = 2010 * object\_radius\_in\_px^{-0.9042} - 20 \tag{4.22}$$

$$Y = -1 \times (object\_position\_px\_horizontal\_axis - 80) + 12 \tag{4.23}$$

4.4.2.2 Kinematic

As the ball in all previous prototypes was reached, by NAO from above the object (see Figure 4.6b in prot1 or Figure 4.36b above), the angle of the reach was therefore, in the x,y,z axis: 0°,0°,0°.

In order to actuate a grasp on the cylinder, the hand had to move from the side, and hence a static angle was set as follows:

X angle: -5°
Y angle: 10°
Z angle: -32°

Sub-motions were implemented in prototype 3 and set at 16 sub motions, in this prototype a 17th sub-motion was added.

The object was attained from a height (Z axis) offset, to get into a position above the object, then once this end point was reached, the arm moves down to the level of the object's height and grabs it.

At 2/3 of the motion completion, the hand opens, independently of the number of sub-motion implemented, and closes at the end of the motion.

Once the object was grasped (and even if it wasn't), the shoulder lifts it to shoulder-height, and then goes down and up again.

4.4.2.3 Software architecture

The general architecture of the prototype is presented in Figures 4.37 and 4.38.

In Figure 4.37, each node of the system is depicted in a square. There are 3 ROS nodes plus the simulation benchmark. This figure outlines the communication messages between the nodes i.e. who sends and who receives the ROS messages.

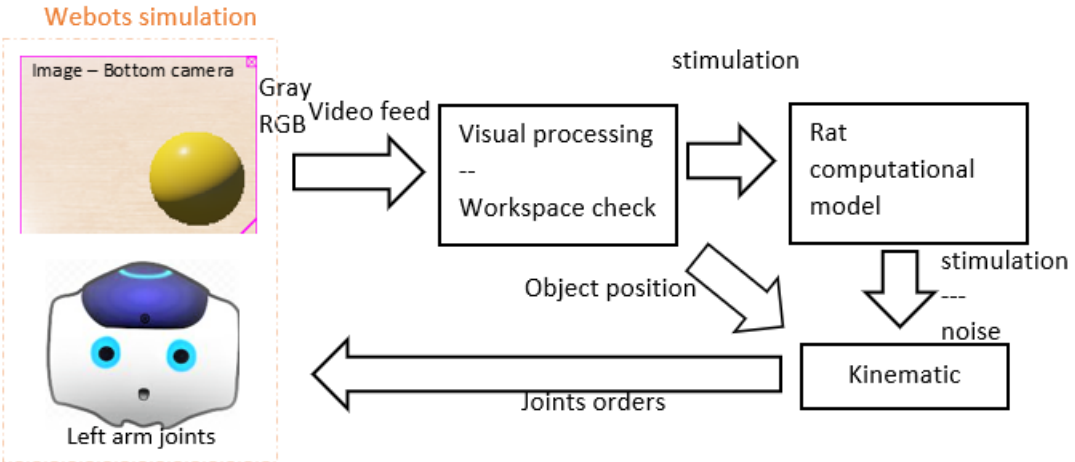


FIGURE 4.37: General architecture, communication between nodes

The Figure 4.38 presents a general flowchart of the whole process (from visual processing to reaching and grasping). It is simplified here for representational purposes, but it should be noted that each node works in parallel to each other, while the flowchart presents a linear process. Actually, when the kinematic works out the joints orders to send to the simulation, the visual processing keeps receiving a video feed from the simulation and processes the image.

However, no new data sent to the rat model/kinematic will be treated until the previous work is finished and sent to the next node. That's why it was simplified into a seemingly linear process.

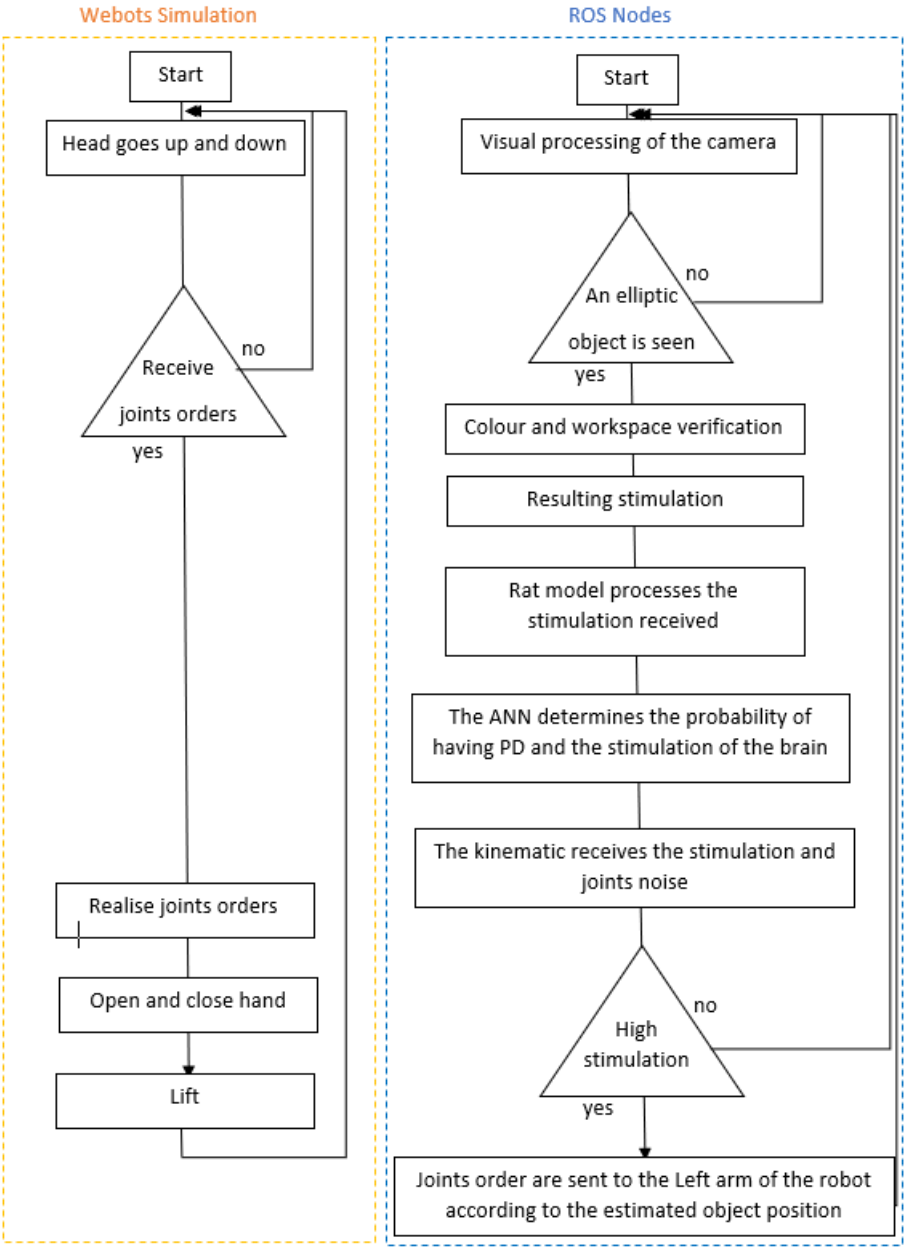


FIGURE 4.38: Simplified general flow chart of the process

4.4.3 Results and Analysis

Visual processing

The distance of the object was well-estimated with an error of a few millimetres as reported in the table 4.12 (biggest error: 3.9mm). The Y position didn't present a greater error than the one reported in prototype 1, namely it never exceeded an error of 1.5cm in the reachable zone. This would indicate that the biggest MSE error is of:

$$\sqrt{(3.9)^2 + (15)^2} = 15.50mm \tag{4.24}$$

Real D	Estimated D	Cylinder R
110	111.66	25.01
120	120.61	22.5
130	130.04	20.58
140	140.74	18.9
150	150.05	17.72
160	158.18	16.37
170	168.75	15.33
180	177.77	14.36
190	186.39	13.88
200	196.1	12.87
210	210.07	12.13
220	220.24	11.48

TABLE 4.12: Cylinder distance estimation compared to real distance. D stands for Distance, and R for Radius

To fully verify if the visual processing was really adapted to a reaching and grasping task in the whole workspace, positions were tested in the whole workspace . The results are shown in Figure 4.39, where the green-fill area represents the reachable zone; the boxes bordered in red are the grasping zones; the white boxes are out of NAO's visual area and dotted boxes represent zones that should be exploitable but are not. Nao could reach 93% of the workspace surface and grasp in 62.7% of the workspace surface.

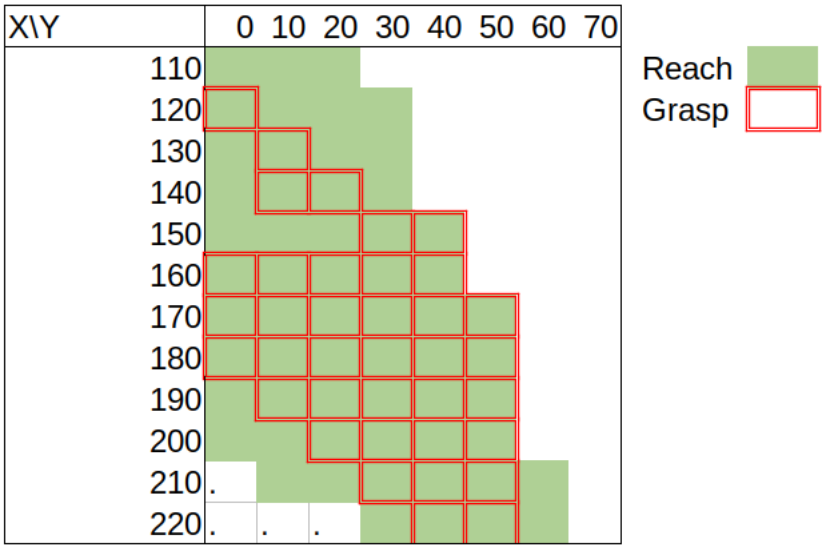


FIGURE 4.39: exploitable workspace for reach and grasp

- To evaluate if a reaching was well-effectuated, the object had to be either grasped or knocked over.
- To consider a grasping motion successful, the object had to be lifted.

  The motion must be natural, i.e. without a sudden change of direction in-between the starting pose and the goal pose (i.e. between the arm extended in front of NAO and the object).

Almost the whole workspace is reachable except for the further left zone of the grid. At this position, the kinematic generates non natural joints orders so even if the object was reached it was considered a failure. The kinematic calculates joints configuration above their maximum possible angle. The 220 mm perimeter was not tested in prototype 1 because it was above the limit of the workspace implemented (218.65mm in the X axis). However, fingers being at the extremity of the hand, this zone can now be considered part of the workspace, gaining 2cm of grasping zone.

Limitations: The grasping zone doesn't completely covers the reaching area. The level of precision required to effectively grasp a small object with a NAO hand is high. For instance, 1cm offset from the real position means that one of the fingers will bump into the object. The angle of approach is also at fault here, the static angle being not well adapted to objects close to NAO, as one of the fingers can touch and knock down the object, preventing any successful grasping.

Healthy /PD Results

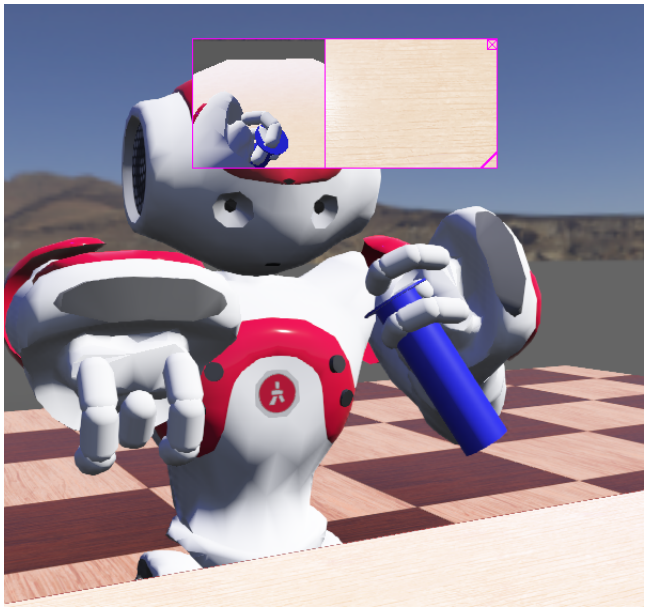


FIGURE 4.40: NAO lifting the cylinder

	Simulation (Ground truth)	**Estimated coordinates**	**Reached position**
Object colour	Blue	Blue	Blue
Position x (mm)	170	169.06	149.06
Position y (mm)	10	12.5	24.5
Position z (mm)	-20	-20	-20

Calculating the MSE of the estimated position:

$$\sqrt{(170 - 169.06)^2 + (10 - 12.5)^2} = 2.67mm \tag{4.25}$$

	Healthy	**PD**
PD probability	38%	99.9%
Noise	0	0.203
Object reached	True	True
Object grasped	True	True
Mean Ctx-r firing rate	6.88	8.13
Mean ISI	126.25	130.12
STD ISI	30.71	35.50
Rat model seed	129.53	474.45

TABLE 4.13: Prot5: Table of results Healthy/PD lifting test

10 Healthy and 12 PD tries have been realised: the PD and healthy cases were always well estimated, the highest PD probability (calculated by the neural network) while being Healthy was 38%, the lowest PD probability while having PD was 53% (associated noise: 0.046rad).

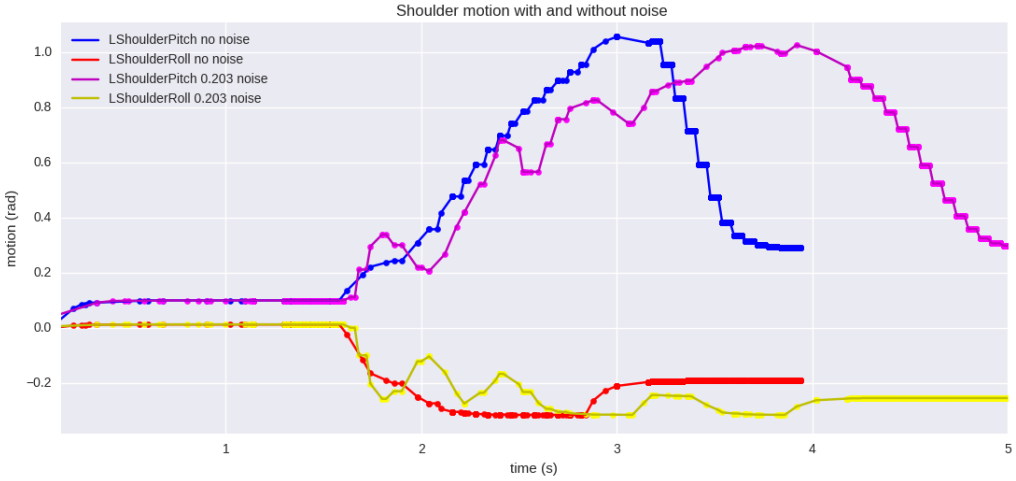
	N° of runs	Reaching success	Grasping success
Healthy	10	100%	100%
PD	12	41.6%	16.7%

TABLE 4.14: Reaching and Grasping success rate

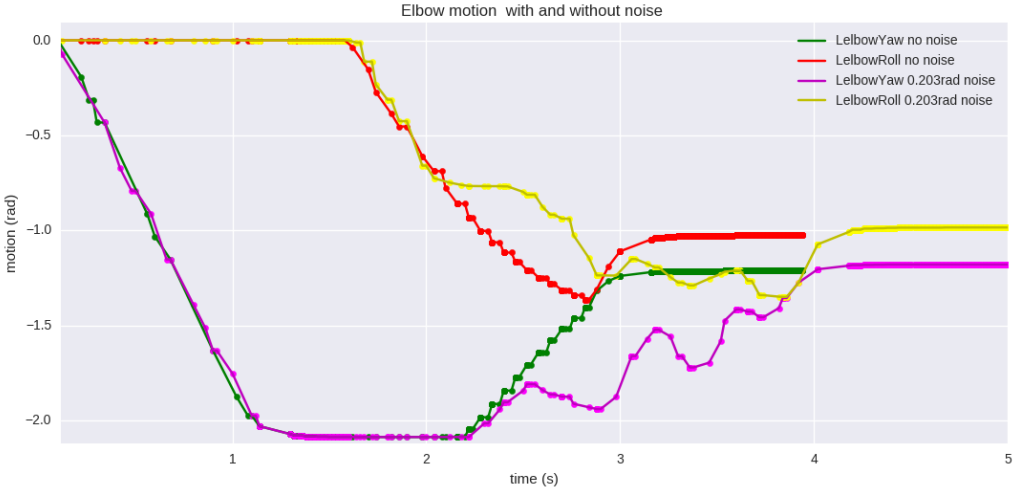
- "Reaching successfully" means the object location was covered by the hand.
- "Grasping successfully" means the object was lifted from the table.

With PD, 25% of the time, the object was knocked off by the trembling arm, 8.3% of the time the object was lost during grasping, the rest of the failures (50% of the time), Nao's grasp was beside the object. There is no arm tremor during lifting.

It was observed that NAO could grasp the object while having PD, especially when the PD probability was under 65%, therefore reducing the maximum noise sent to the joints would augment the reaching and grasping probabilities.



(A) Shoulder motion with a PD and a Healthy brain model



(B) Elbow motion with a PD and a Healthy brain model

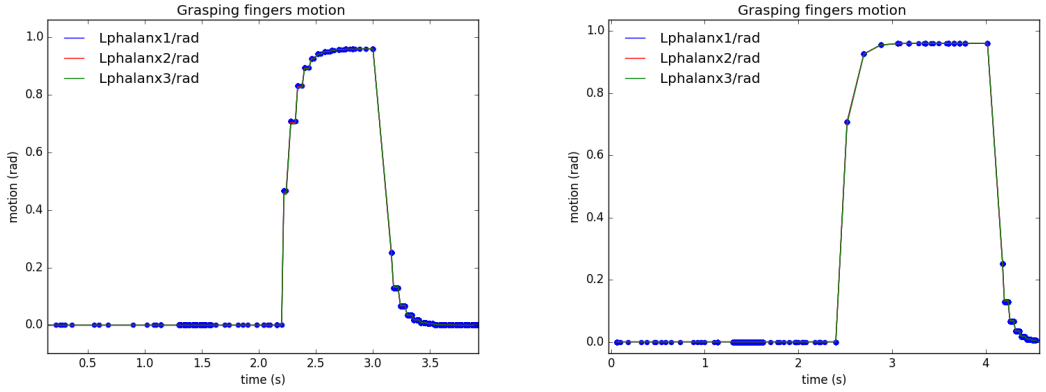
FIGURE 4.41: Comparison between different noise input

Considering the healthy motion (Figure 4.41: blue and red line in the shoulder motion, green and red line in the elbow motion), the plots are not strictly comparable to the ones obtained in prototype 3 and 4. That's because a reaching and grasping angle was added in this prototype. The shoulder Pitch settles 0.7rad higher than in the previous prototypes. Visually, in simulation, the shoulder goes lower than it did previously. The Shoulder Roll is not affected by the angle change. Both the Elbow Yaw and Roll are affected by this difference in the reaching angle, the elbow yaw being 1.2 rads lower than in prototype 3 and 4 and Roll 0.3 rads higher.

Still on the healthy graph, at 2.93s, the 16th motion (the motion was divided in 16 sub-motions in prototype 3) is executed, the arm reached the end position with a height

offset. In the figure, the lowering of the arm is visible on all the joints between 2.93 and 3.1s. Here, only the shoulder Pitch is affected by the lifting starting at 3.17, after a short delay of 100ms at the end of the motion and grasping. The motion goes from 3.17s to 3.98s, the hand being at shoulder height at the end of this motion.

In the Pd case, the kinematic tremor can be distinctly seen on the reaching motion. This is the highest level of tremor the motion can have (noise of 0.203rad) and keep the motion realistic (absence of sudden and extreme change of direction). The motion reaches its goal about 500ms after the healthy model, corresponding to U. Castiello & al (1999) [32] observations in PD patients (see the literature review section 2.1.5 and the explication why the speed and acceleration could be used as a sign of PD in **Appendix. Noise affectation**). The lifting of the object has no tremor.



(A) Healthy fingers motion

(B) PD fingers motion

FIGURE 4.42: Comparison between Healthy and PD finger in a open-close motion

The open-close hand plot is much more detailed here than in prototype 1 (Figure 4.8), the joint motion is composed of more data. The hands open at around the same time, between 2 and 2.6s and there is no noise in the hand opening angle. However the left hand stays opens much longer when the robot has Parkinson. The Healthy hand is open for about 0.54s, while the Pd hand stays open for almost double the time, which is also in accordance with the observation of U. Castiello & al (1999) [32].

Chapter 5

Discussion

The objective described in Chapter 3 section 3.1.1 has been reached. Namely, the aim was to embed a computational model in NAO, capable of performing a reach-and-grasp motion, based on visual stimulation and neural activity. Software architecture was designed and implemented in a simulated robot, to provide a body-environment context to an existing computational rat brain model. The whole project was implemented in a Webots simulation with ROS, Python and CPP.

The simulated NAO had to be able to discriminate between visual outputs and determine from these outputs the position of a given object, and finally decide whether the seen object is interesting or not. This requirement was fulfilled by Prototype 1 and 2 (section 4.1 and 4.2).NAO could see the zone in front of it (in between its arms and from its torso to the horizon) could extract elliptic shapes from its visual inputs and determine the position and colour of the object seen. If the object was green or blue and located in its left arm workspace, then a high stimulation was sent to the rat model, otherwise, a low stimulation was sent. The determination of the position of a cylinder was accurate with a maximum MSE of 1.55cm (Result of section 4.4). The colour of the object was well estimated 100% of the time.

Stimulating inputs were sent to the thalamus area of the rat's brain model, provoking a different brain behaviour depending on whether the object was deemed interesting (colour green/blue and in the workspace) or not. The low stimulation level was 1.2mA and the high stimulation 3.5 mA (Prototype 2 section 4.2).

The rat brain model R-Cortex reacted by displaying very distinct behaviour:

when lowly stimulated

- the mean firing rates were generally situated between 3 and 5 Hz (depending on whether the model had PD or not),
- the spiking was regular and distant in time (high mean ISI and low std -in ms-)

when highly stimulated

- the mean firing rates generally situated between 6 and 9 Hz,
- the spiking was irregular and much more frequent (low mean ISI and high std -in ms-)

The rat brain model's behaviour (whether it had PD or not) was in line with the classical models presented in the literature review (Section 2.1.4.2).

Next, the outputs of this rat model, namely, the mean firing rate and mean/std ISI of the Cortex-R, have been extracted and converted into noise in order to be sent to the simulated NAO, to induce kinematic tremor in its reaching motion (Prototype 3 section 4.3).

The PD and Healthy states distinction being difficult to observe, a Feed Forward Neural Network was implemented to best discriminate between PD/Healthy and stimulating/non-stimulating rat model outputs. Once the Cortex R outputs stable, they were extracted from the brain model and inserted into the ANN. The ANN treated this data and generated new outputs, providing:

- the stimulation of the brain in a binary value (0/1)
- the PD probability (from 0 -Healthy- to 1 -PD-).

The accuracy of the neural network, when asserting the stimulation and PD probability, were respectively, 100% and 94.8% (on 2750 values).

Once the Pd probability extracted from the ANN, it was re-scaled between 0 and 0.203rad to be sent as noise to the joints of the robot. Above a PD probability of 0.4, the model starts having noise, though its only small perturbations until a probability of 0.55 . The noise was sent to the joints as a maximum noise scale. Each joint received a random perturbation with the noise as maximum and minimum (rendered negative) all along the trajectory. When the noise is high, NAO displays symptoms in simulation.

This implementation is impossible on a real NAO: the trajectory itself should be perturbed, not the joints directly. Moreover, the plot of the noisy motion over time is

realistic, but not 100%. Kinetic tremors should display similar characteristics to the resting tremors, though with a lower amplitude. The tremor obtained was much more visible than the resting tremor observations reported in this thesis. The frequency and amplitude of the tremor could be tuned by modifying the noise scale and reducing the maximum noise sent to the joints. The graphs (motion over time) are still relevant for further studies as Healthy and PD neural activities in the brain model resulted in clearly different motions.

Finally, the robot was meant to grasp the object and lift it to prove it was grasped (Prototype 5 section 4.4). The arm's trajectory was slightly altered in order to account for the gap between NAO's fingers and its hand centre. With this adjustment, NAO could successfully reach and grasp a cylinder:

Reach (to be considered reached the object position had to be covered):

- 100% of the time while displaying an Healthy neural activity,
- 41.6% with a PD neural activity.

Grasp (the object had to be lifted):

- 100% of the time while Healthy,
- 16.7% with a PD neural activity.

NAO grasped the object effectively, when the PD probability was under 65%. Therefore, reducing the maximum noise sent to the joints would augment the reaching and grasping probabilities.

Moreover, NAO could reach toward the object in 93% and grasp in 62.7% of the workspace surface area, respectively (see prototype 5 Figure 4.39).

The resulting motion plotted over time still displays a clear distinction between a PD vs a Healthy rat model output. This work may allow further researches to deepen the understanding of PD mechanisms and help develop remedies on reusable models. Hence, this would reduce tests and studies conducted on animals, increasing the number of experimentation and greatly reduce financial costs on long-term studies.

However, to be relevant, this work should first be adapted and realised on a real NAO. The implementation of this thesis was, unfortunately, exclusively realised in simulation due to the Covid-19 situation. As the present implementation did not use NAO libraries, a large amount of re-writing of the architecture to fit the real robot had to be done.

Some adaptation suggestions can be found in the respective implementation sections of the prototypes.

Chapter 6

Conclusion

6.1 Summary

The objective of this thesis was to realise a robot architecture to reach-and-grasp while displaying characteristics of an healthy and PD case movement based on an already existing computational model. The purpose was to contribute to the Neuro4P project by creating a body-environment for a brain model, in order to facilitate further PD studies on the brain-body-environment interactions (to reduce the use of animals in tests). The work was realised in simulation with NAO (webots), ROS, Python, Cpp and Netpyne.

The robot was able to reach and grasp a defined object in a given workspace with its left arm. The joints positions were extracted at a given interval to supervise the motion and compare the behavioural characteristics of an Healthy vs PD brain model.

The results showed a visible difference between the Healthy and PD case, with PD displaying a kinetic tremor while effectuating the reaching motion.

In the left-arm reaching workspace, the healthy NAO successfully reached for the object in all tests, while the PD succeeded only 41.6% of the time. In the left-arm grasping workspace, which was a relatively smaller than the reaching one, the healthy vs the PD neural activity were able to grasp the object, 100% vs 16.7% of the time, respectively.

The joints motion plotted over time display tremors that are almost realistic, although not 100%, as the tremor is very obvious as compared to a natural tremor observed on real PD case. However, it is merely a question of tuning to meet the requirements and not an architecture shortcoming, as the tremor was exaggerated in order to distinctly observe it in this setting, compromising realism a little bit, in order to get a distinct result.

Nonetheless, a good differentiation of the PD and healthy case is observed 94.8% of the time (on 2750 values) without actually telling the architecture whether the rat computational model had Parkinson or not. This should provide flexibility to modify the model, and observe if the resulting computational outputs are still considered Parkinsonian or not.

6.2 Future work

This work [1] was realised with the objective of being reused in future studies. Therefore, the architecture is meant to be modified and tuned to fit future requirements. Unfortunately, due to the Covid-19 pandemic, this work was exclusively realised in simulation and not on a real NAO as forecasted. Adapting this work to a real NAO would greatly improve the results obtained here. Moreover, only the thalamus received a binary stimulation in this work and, due to limitations of time, all the possible configurations could not be tested. It may be interesting to complexify the system to attain more realism and add metrics to measure the speed of the motion independently from the motion itself.

An ANN was realised in this thesis, with a good computational power, it may be further improved to increase the distinction between a healthy vs PD case. Finally, this study was meant to allow further understanding of unclear PD mechanisms and help other teams search for remedies.

[1] Github of the work: https://github.com/my-name-is-D/object-detection-Motion-Nao-webots

Chapter 7

Appendix

7.1 Prototype 1

kinematic matrix

Matrix of the Forward kinematic of Nao's Left arm T_0^4.

The symbolic matrix was obtained using Wolfram Matematica. Wolfram matematica is a software tool to compute mathematical expressions created by the Wolfram company. It allows the performance of large-scale symbolic computations with matrices and simplify symbolic expressions [1].

[1] https://www.wolfram.com/mathematica/

$$
\begin{pmatrix}
\begin{array}{l}
\text{Cos}[\text{theta4}]\,\text{Sin}[\text{theta1}]\,\text{Sin}[\text{theta3}] + \\
\quad \text{Cos}[\text{theta1}]\left(\text{Cos}[\text{theta2}]\,\text{Cos}[\text{theta3}]\right. \\
\qquad \left.\text{Cos}[\text{theta4}] - \text{Sin}[\text{theta2}]\,\text{Sin}[\text{theta4}]\right)
\end{array} &
\begin{array}{l}
-\text{Sin}[\text{theta1}]\,\text{Sin}[\text{theta3}]\,\text{Sin}[\text{theta4}] - \\
\quad \text{Cos}[\text{theta1}]\left(\text{Cos}[\text{theta4}]\,\text{Sin}[\text{theta2}] + \right. \\
\qquad \left.\text{Cos}[\text{theta2}]\,\text{Cos}[\text{theta3}]\,\text{Sin}[\text{theta4}]\right)
\end{array} &
\begin{array}{l}
\text{Cos}[\text{theta3}]\,\text{Sin}[\text{theta1}] - \text{Cos}[\text{theta1}] \\
\quad \text{Cos}[\text{theta2}]\,\text{Sin}[\text{theta3}]
\end{array} &
\text{Cos}[\text{theta1}]\left(\text{l1}\,\text{Cos}[\text{theta2}] + \text{l2}\,\text{Sin}[\text{theta2}]\right) \\
\begin{array}{l}
\text{Cos}[\text{theta3}]\,\text{Cos}[\text{theta4}]\,\text{Sin}[\text{theta2}] + \\
\quad \text{Cos}[\text{theta2}]\,\text{Sin}[\text{theta4}]
\end{array} &
\begin{array}{l}
\text{Cos}[\text{theta2}]\,\text{Cos}[\text{theta4}] - \text{Cos}[\text{theta3}] \\
\quad \text{Sin}[\text{theta2}]\,\text{Sin}[\text{theta4}]
\end{array} &
-\text{Sin}[\text{theta2}]\,\text{Sin}[\text{theta3}] &
-\text{l2}\,\text{Cos}[\text{theta2}] + \text{l1}\,\text{Sin}[\text{theta2}] \\
\begin{array}{l}
-\text{Cos}[\text{theta2}]\,\text{Cos}[\text{theta3}]\,\text{Cos}[\text{theta4}] \\
\quad \text{Sin}[\text{theta1}] + \text{Cos}[\text{theta1}]\,\text{Cos}[\text{theta4}] \\
\quad \text{Sin}[\text{theta3}] + \text{Sin}[\text{theta1}]\,\text{Sin}[\text{theta2}] \\
\quad \text{Sin}[\text{theta4}]
\end{array} &
\begin{array}{l}
\text{Cos}[\text{theta4}]\,\text{Sin}[\text{theta1}] \\
\quad \text{Sin}[\text{theta2}] + \left(\text{Cos}[\text{theta2}]\,\text{Cos}[\text{theta3}]\right. \\
\qquad \left.\text{Sin}[\text{theta1}] - \text{Cos}[\text{theta1}]\,\text{Sin}[\text{theta3}]\right) \\
\quad \text{Sin}[\text{theta4}]
\end{array} &
\begin{array}{l}
\text{Cos}[\text{theta1}]\,\text{Cos}[\text{theta3}] + \text{Cos}[\text{theta2}] \\
\quad \text{Sin}[\text{theta1}]\,\text{Sin}[\text{theta3}]
\end{array} &
-\text{Sin}[\text{theta1}]\left(\text{l1}\,\text{Cos}[\text{theta2}] + \text{l2}\,\text{Sin}[\text{theta2}]\right) \\
0 & 0 & 0 & 1
\end{pmatrix}
$$

Prototype 1 alternative version with a webcam image feed

This second version was created in order to have results closer to reality and adapted to a real camera.

Additional version of prototype 1 with a real image feed

Design

This version was realised to fit a real case situation and consider the issues to face with a real image feed.

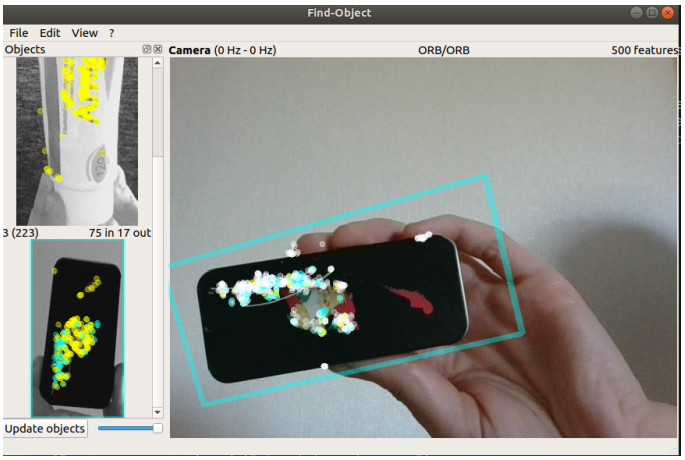
Images were extracted from a computer webcam, sent to a ROS package of image recognition "find 2D object" and calculate a real object position in space according to the webcam specifications (field of view etc), then the object position is sent to the kinematic package as presented in Prototype 1 section 3.3.1.

Implementation

Real images have been extracted from a computer webcam, the object detection was realised through ROS with the "find 2D object" package.

The use and configuration of "find 2D object" works as follow. pictures of the object are fed to the platform which will then extract and store the main features from it. One picture may be enough if there is no need to recognise it from different angles. Once it recognises the object a list of data (11 elements) can be retrieved from a topic:

- Object ID,
- width and height of the object in pixel,
- the position of its 4 cardinal points in x and y in the picture,



(A) Picture of the platform with an object recognised

```
---
header:
  seq: 70
  stamp:
    secs: 1590082174
    nsecs: 629040218
  frame_id: "3"
point:
  x: 25.5917816162
  y: 4.4684510231
  z: 10.3488759995
---
```

(B) Point stamped topic after treatment to retrieve the x,y,z position of the object in space in cm

FIGURE 7.1: Object recognised and position extracted.

From the package topic the centre position could be extracted and the object position could be estimated in a camera reference frame. figure 7.2.

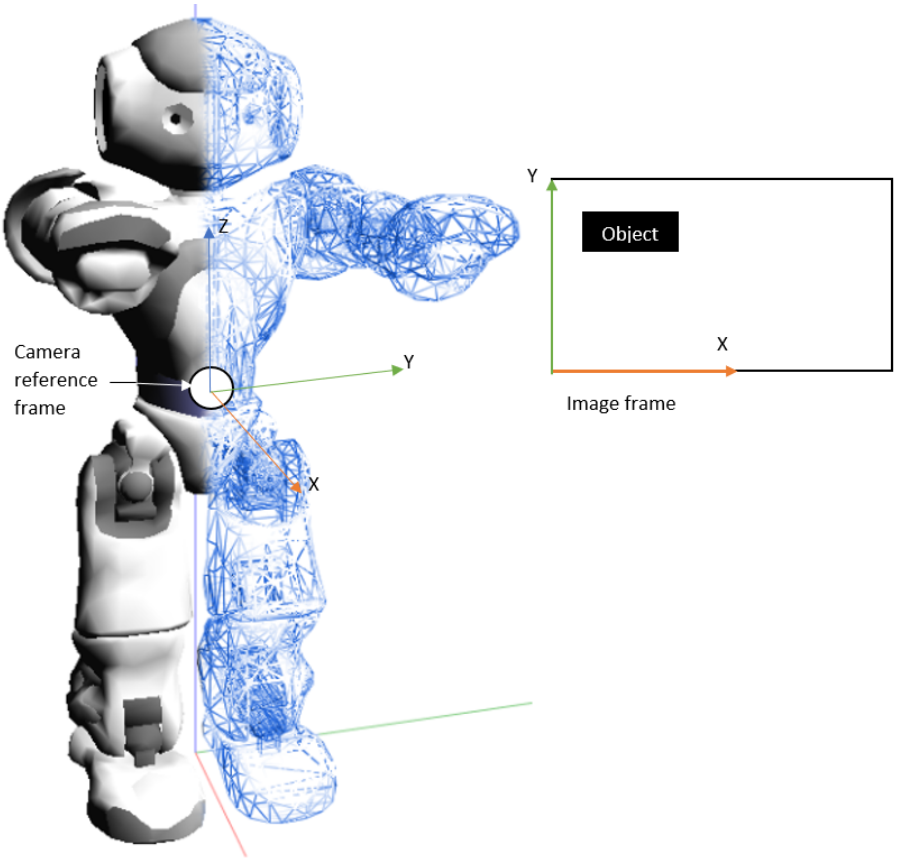


FIGURE 7.2: Nao's camera frame and image references. An image is 2D and composed only of x and y disposed on the left down angle, thus the reference was shifted to the centre of the image and modify the axis.

The distance x was estimated by comparing the object area in pixel and the real object dimension. The surface of simple forms gives better are easier to accurately calculate and gives better results.
The following calculus were based on the camera parameters presented in the table below:

Focal length	605px
Vertical resolution	720p
Horizontal resolution	1280px
Estimated angle perspective	65°

TABLE 7.1: the used webcam parameters, the Vertical and Horizontal resolution compose the webcam resolution.

The position of the object is given in a camera reference frame (see figure 7.2).

$$X = camera\_focal\_length \times \frac{real\_object\_area\_in\_mm}{object\_area\_in\_px} \tag{7.1}$$

the object angular position was then extracted

$$Angular\_position = -1 \times (65 \times \frac{object\_position\_in\_pixel\_y\_axis}{720} - 32.5) \tag{7.2}$$

With 32.5 to centre the result, the centre of the image being therefore 0 in the y axis (in a camera reference frame).

$$Y = X \times Sin(Angular\_position \times \frac{PI}{180}) \tag{7.3}$$

$$Z = -1 \times 0.2645833333 \times (object\_position\_in\_pixel\_z\_axis - 360) \tag{7.4}$$

With 1pixel = 0.2645833333mm, and 360 to centre the position (the resolution being 720p vertically), the centre of the image is therefore 0 in the z axis.

X,Y,Z are in a camera frame reference, therefore all the axis mentioned in the equations are camera axis. Since for Nao's kinematic the base of reference is the centre of the torso

and that the values weren't shifted from one reference to another (to simulate having Nao's bottom camera) the camera is considered being in the centre of Nao's torso as presented in 7.2.

Results and analysis

This additional version of prototype 1 takes as input the object position seen on camera (with x,y,z in a webcam camera reference, not Nao's camera reference, the objects won't be positioned on the table but all around Nao with the torso as origin).

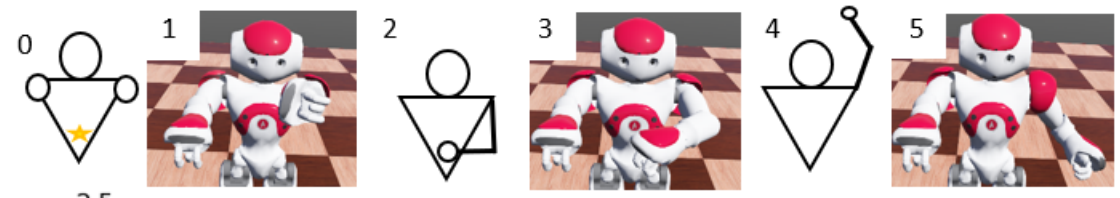
On the experiment presented in figure 7.3 the object was moved in 5 different positions in front of the webcam.

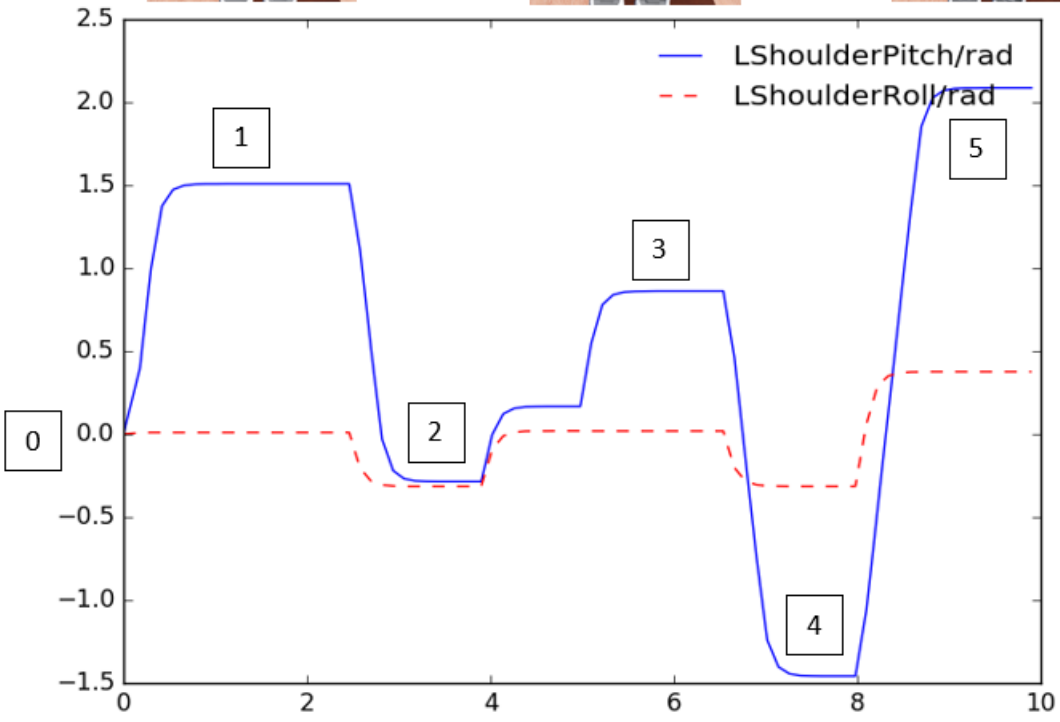
The robot didn't distinguish from reachable and unreachable position in this prototype, it tried to get as close as possible to the objective whatsoever. Therefore the arm poses in the following figure do not seem to correspond to the given object position. It's because the goal being out of Nao's workspace the given joints positions are aberrant.

In this experiment only the third position is accurate, the goal is in the workspace. None of the others are because the distance from the camera and the object is too great. The focus couldn't be well realised under a 100mm distance, the object was not recognised. Considering that the maximum reaching distance of Nao is of 218mm, the possible range in those circumstances is narrow. It is possible to subtract 100mm from the actual distance object/camera to have more margin, but it hasn't been realised on the following test.

In figure 7.3 the object position is in a torso reference frame (mm), under there is a display of Nao final reaching position for each objective and the under the plot of the joints position in rad/s.

X (mm)	584	446	79	867	360
Y (mm)	-176	-98	2	143	-41
Z (mm)	59	-17	1.7	75	104





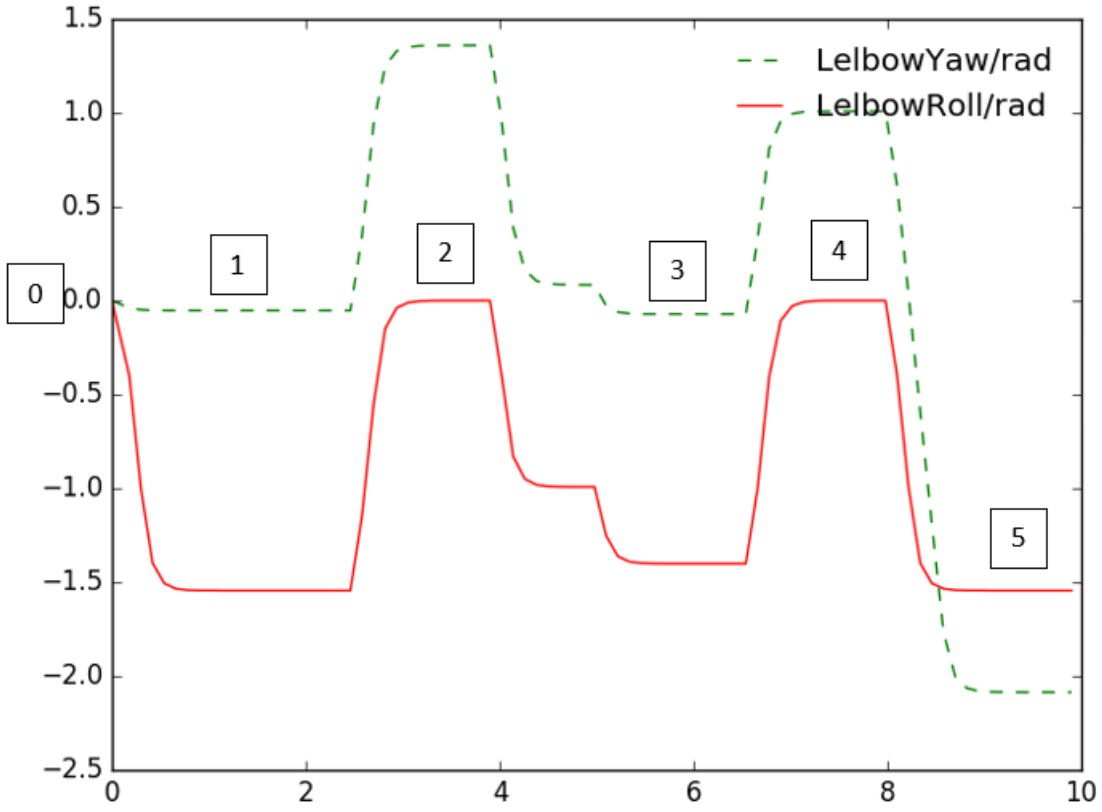


FIGURE 7.3: Plot of the shoulder and elbow joints in radiant according to an object seen with the camera. The object was presented at 5 different places, the centre point (0,0,0) is in the middle of the torso represented with a yellow star. The object position is given in mm, and the graph axis are rad/s

7.2 Prototype 3

Can be found here the main tests that were realised in order to choose the best implementation possible in prototype 3

The first part will cover the machine learning algorithms tested and their resulting accuracy. All the results will be first summarised in a table, and then each of them will be briefly presented.

The second part will cover the tests realised in order to decide where to send the noise (speed, trajectory, etc.) that were not kept in the final implementation. The results of each implementation will be presented with the explanation of why they were considered as an interesting possibility.

Machine learning algorithms

To implement the machine learning models the python library sklearn, Keras and TensorFlow were used.

Machine Learning	Presentation	Metric used	Results
Naive bayes	Probabilistic classifier based on Bayes' Theorem	accuracy	67%
Logistic regression	Probabilistic modelling of an event existing in a binary fashion	accuracy/R^2	46% / -0.52

TABLE 7.2: Machine learning algorithms presentation, the metrics are the metrics used to compare the models.

Dataset presentation

In the Naive Bayes the dataset used was composed of the following elements in gray, and the following output in yellow. In the Logistic regression model only the element of the first row, and the output (yellow) were used.

Cortex-R firing rate (Hz)	ISI mean (ms)	ISI std (ms)
Mean Firing Rate evolution (Hz)	PD flag	

The mean firing rate evolution is the average difference of spiking from one measurement to the next (each 50ms).

Naive Bayes

A Naive Bayes is based on the Bayes theorem of probability.

$$P(A|B) = \frac{P(B|A).P(A)}{P(B)} \tag{7.5}$$

A,B = events

P(A|B) = probability of A given B is true

P(B|A) = probability of B given A is true

P(A),P(B) = the independent probabilities of A and B

The Naives Bayes algorithm considers each input as independent and estimate the probability of having or not PD according to the database.

This method was implemented to determine the probability of having PD only. If the model has PD then the resulting label would be 1, if the model is healthy, the label would be 0. The accuracy of the model was calculated based on the following formula [66] :

$$Accuracy = \frac{(TP + TN)}{(TP + TN + FP + FN)} \tag{7.6}$$

TP = True Positive, are cases with positive labels which have been correctly classified as positive.

TN = True Negative, are cases with negative labels which have been correctly classified as negative.

FP = False Positive, are cases with negative labels which have been incorrectly classified as positive.

FN = False Negative, are cases with positive labels which have been incorrectly classified as negative.

The best accuracy obtained with this method was 67% A confusion matrix was used to observe the accuracy of the model, itc is composed as the table below:

		Predicted class	
		0	1
True class	0	True Positive (TP)	False Negative (FN)
	1	False positive (FP)	True Negative (TN)

TABLE 7.3: Confusion matrix visual explanation

The resulting confusion matrix of the Naive bayes on the testing set :

$$\begin{bmatrix} 170 & 133 \\ 119 & 195 \end{bmatrix}$$

(The confusion matrix is based on 20% of the training data presented Table 4.7).

The mean difference between the probability of having PD or not having PD was not well enough defined to be used to generate noise in the robot. As the healthy case would trigger an incorrect answer a third of the time. Moreover, extracting a firing rate value every 50ms considerably slowed down the computational model of the rat.

Logistic Regression

The Logistic regression is used to classify data into the probability of having PD or not (between 1 and 0). A simple logistic regression is defined by a sigmoid function.

For this the R^2 score was used to determine the performance of the model. It is meaningful when compared to other model using the same metric. This score is the coefficient of determination, it represents the proportion of variance of the output (label) depending on the independent variables of the model. It measures the probability of how well new samples will be predicted by the algorithm [66]. The result can vary from -1 to 1, 1 being the best result, 0 meaning that 50% of the sample are not well estimated and -1 that the model perfectly reverse all the answers. This score is based on the used dataset, changing dataset may completely change the result.

The resulting confusion matrix of the Logistic regression :

$$\begin{bmatrix} 148 & 155 \\ 150 & 164 \end{bmatrix}$$

Metric	Results
R^2 score	-0.5228
Accuracy scor	46%

The accuracy was yet too low to be acceptable and the logistic regression was only used to estimate if the model was Parkinsonian or not, adding another label reduced the accuracy of this machine learning algorithm.

Appendix. Noise affectation

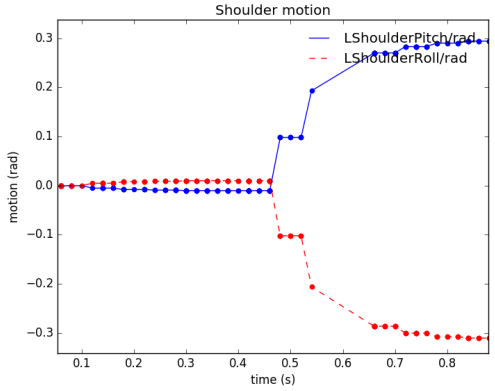
Here will be presented the tests realised to chose where to send the noise generated by the ANN. The healthy control test will be presented at the beginning and then each test will be presented with their associated results.

- First Test - Affecting the velocity of the motion
- Second Test - Affecting the acceleration of the motion
- Third Test - Affecting both the acceleration and velocity of the motion
- Fourth Test - Affecting the trajectory of the motion

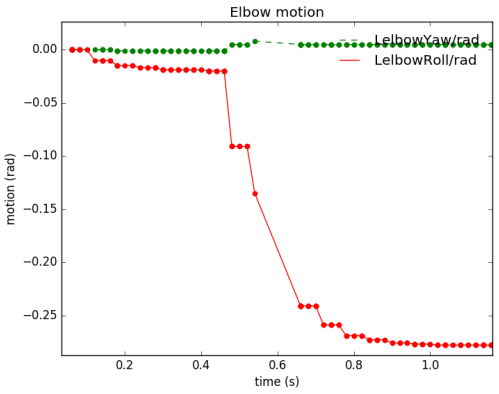
Control test

The control test is a normal motion conducted without any error toward a green ball.

	Simulation (Ground truth)	**Estimated coordinates**
Object colour	Green	Green
Position x (mm)	196	188.2
Position y (mm)	15	15.1
Position z (mm)	5	5



(A) Shoulder motion with no interference implemented



(B) elbow motion with no interference implemented

FIGURE 7.4: Control Test - Shoulder and elbow motion without any interference

First Test - Velocity

In the literature review Section 2.1.5, the speed and acceleration of a PD reaching is affected by the disease.

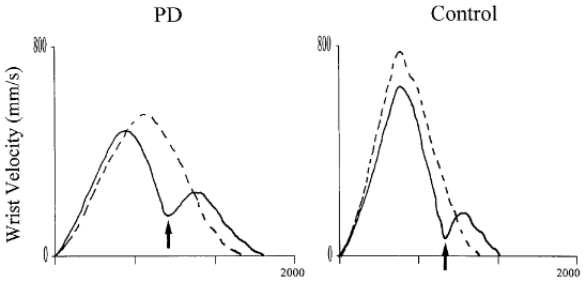


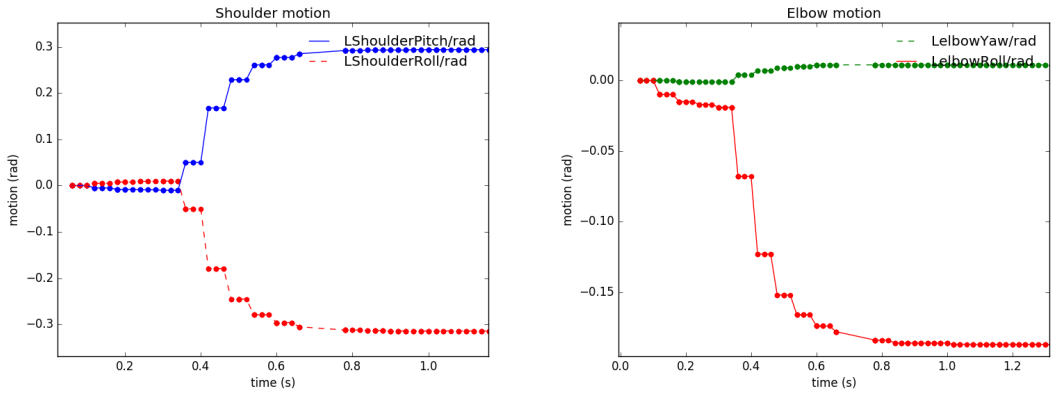
FIGURE 7.5: Reaching speed with or without PD. The dotted line results of a reach toward the same object twice, the continuous line results of an experiment of reaching toward two objects having different sizes and position [32]. The plot displays speed (mm/s) over time (ms).

In the plot above, the overall speed in PD is much slower then the control's one. And when passing from an object to the next, the velocity is slow to adapt. The maximum speed is achieved around 100ms later in PD patients than healthy individuals (also based on Figure 2.10 and other written results from U. Castiello & al study (1999) [32] experiments in the literature review section 2.1.5). Moreover the whole execution time of the motion is around 400 to 500 ms slower in PD.

To affect the velocity of the motion a ROS message was created in order to simulate a signal going from the minimum to maximum value accepted by the muscle (knowing

that each joint as its own min/max speed), If the signal is low then the motion is slow and inversely. In this test the signal is high then low then rise to a high value once again.

The visual result were not significant, the ball was reached more or less rapidly, without exceeding the final position.



(A) Shoulder motion with a variant speed

(B) elbow motion with a variant speed

FIGURE 7.6: Shoulder and elbow motion with a variant speed as the interference

On the graph 7.6, the final position was reached at 0.9s, the motion finished the motion 100ms after the control test, the noisy and control motion are comparable.

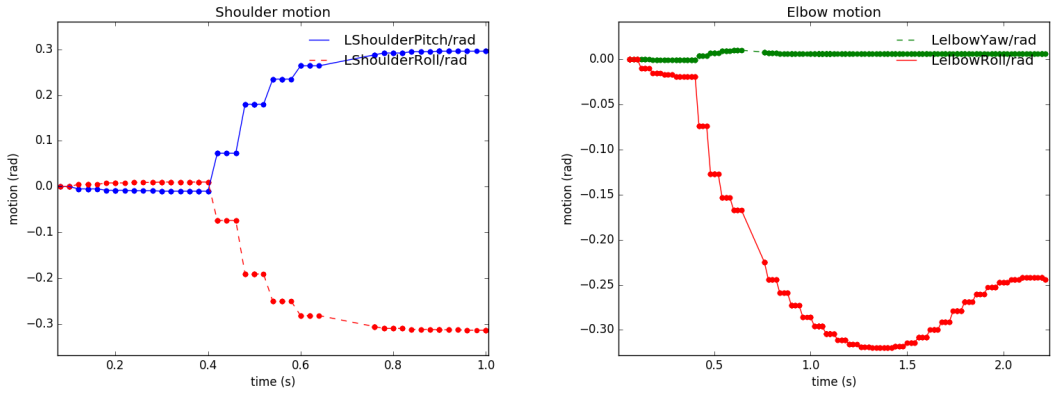
The motion speed could be an interesting parameter to affect, however the speed difference between an healthy and PD case is not defined enough to be observed in a muscle motion plot. A slowness of the motion can indeed be observed, but the difference may not be relevant enough to discriminate between a PD or Healthy neural activity.

The metric is not adapted to observe speed.

Second Test -Acceleration

According to U. Castiello & al study (1999) [32] there is a 100ms difference between the peak acceleration of a PD and an healthy reaching (PD being slower). And there is 300ms difference while decelerating, with PD patients taking longer to decelerate. This affected deceleration may lead to an exaggerated motion missing the target or bumping too violently into it. In this test the acceleration was affected in the same fashion as the speed (test 1), with a different min and max value.

The acceleration was going from a high to low value then high again. The plots displays a motion that go past the wanted position before correcting itself.



(A) Shoulder motion with a variant acceleration

(B) elbow motion with a variant acceleration

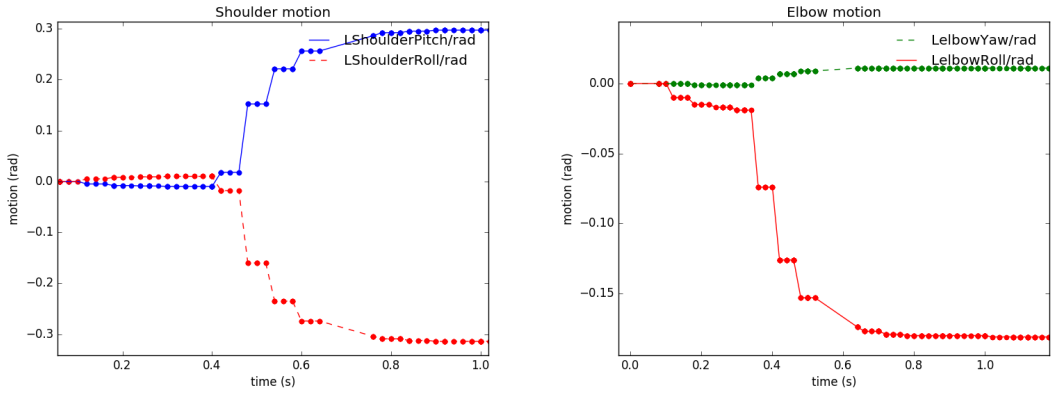
FIGURE 7.7: Shoulder and elbow motion with a variant acceleration as the interference

The elbow acceleration clearly passed the goal at around 1.2s before correcting its position at around 2s. The shoulder presents a comparable graph as the control test's. This is coherent with the assumption the motion may pass the wanted position with a too long deceleration phase. However, since the muscle motion are observed and not the speed/acceleration directly, depending on the object position, the phenomenon may not be as well defined as it is here. Finally, running this test several times, there is a slightly different result at each run (the goal position being passed over or not. It is due to a synchronisation issue, the deceleration command reception may vary of some microseconds at each run).

There is an issue of repeatably with this implementation.

Third Test - Velocity and acceleration

Finally to be complete, both the speed and the acceleration were affected. With an affected acceleration and velocity the motion can reach the end position extremely rapidly and precisely, or not, the result varied from one run to the next.



(A) Shoulder motion with a variant speed and acceleration

(B) elbow motion with a variant speed and acceleration

FIGURE 7.8: Shoulder and elbow motion with a variant speed and acceleration as the interference

The reaching was quite precise in this run, it can only be observed that the motion was slightly slower than in the control test 7.4. The scale of the graph is not appropriate to observe such precise time difference.

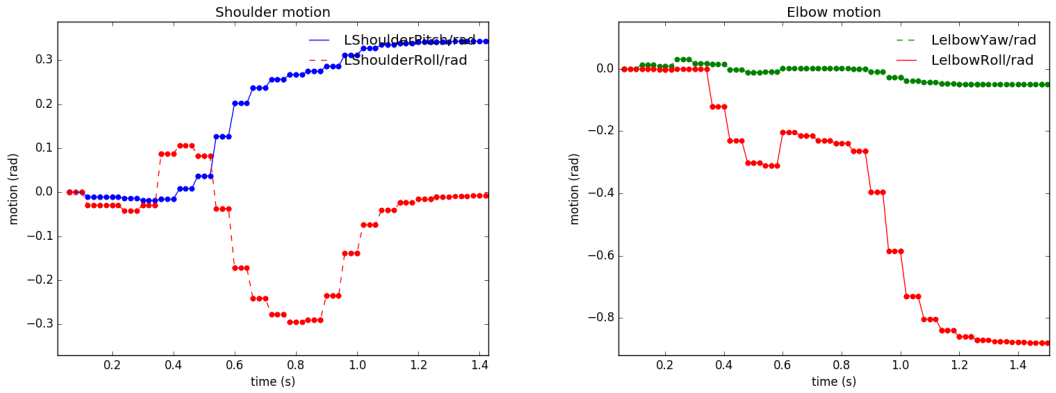
The motion plot metric is not adapted for measuring speed and acceleration.

Fourth test - Trajectory

This test follows the same justification as the one implemented (interference sent to NAo's "muscles"), namely: kinetic tremors symptoms, with similar frequencies and lower amplitudes than resting tremor [63, 64].

This test makes also use of the sub-divided motion in order to observe noise. The noise was sent to the end point x,y,z with a random offset chosen between -5 and 5mm. It was directly added to the trajectory, instead of the muscles in the implemented solution.

The plot below shows an uncontrolled comportment of the left arm while trying to reach the ball.



(A) Shoulder motion with a variant trajectory line (B) elbow motion with a variant trajectory line

FIGURE 7.9: Shoulder and elbow motion with a trajectory error as the interference

In this graph the arm completely changes trajectories at 0.4, 0.6 and 1s. If this result is compared to the rest tremor displayed below, the realism of this realisation can be discussed. It is far from realistic, the motion had too big displacements, they are too accentuated change of direction.

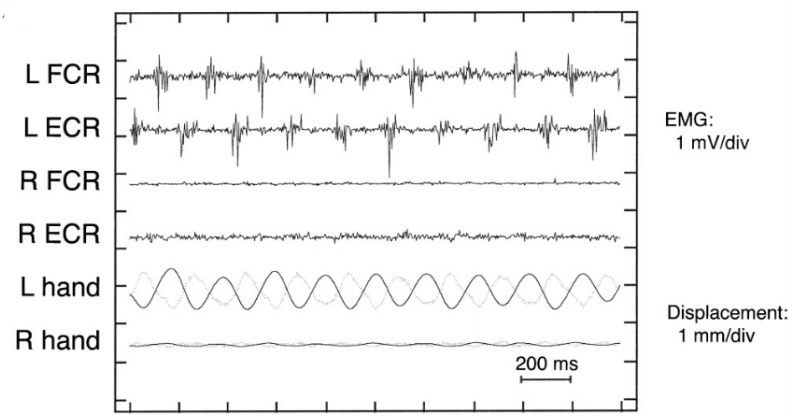


FIGURE 7.10: EMG and Movement Analysis of Rest Tremor in a Patient with PD. The first four traces represent surface EMG signal from forearm muscles; ECR stands for extensor carpi radialis and FCR for flexor carpi radialis. The bottom two traces reflect displacement (darker line) derived from accelerometry. In this case the Left hand is symptomatic [64]

However, if the result of this experiment was not concluding and not implemented, as affecting the muscles directly was more realistic, I assume it may be possible to tune the noise sent to the trajectory and finally observe similar results to the one obtained in this thesis. The frequency of the tremor can be affected by adding or subtracting the number of sub_pose and alter the amplitude by choosing the maximum error implemented.

7.3 Explanation List

This appendix assemble short explanations on terminology that may hinder the lecture. Those explanations are far from complete for a in-depth understanding of the subject.

basal ganglia: The basal ganglia is a part of the brain much affected by the Parkinson disease. See further details section 2.1.4.2 -Click here: , to return to text -abstract-

Neurorobotics: Neurorobotics is a branch combining Neuroscience with robotics, which deals with embodied autonomous neural systems like brain-inspired algorithms. At its core, neurorobotics is based on the idea that the brain is embodied and the body is embedded in the environment. See more section 2.2 -Click here: , to return to text -abstract-

L-DOPA: it is a molecule that is a normal part of human biology. It is a precursor to dopamine. More section 2.1.3 -Click here: , to return to text -Motivation-

modified DH method: The difference between the classical DH and modified DH method is the determination of the parameters and the DH matrix calculation. Whichever method is used, the end result should be the same. In this case the DH parameters correspond to the modified DH.
The classical DH calculates each joint matrix as follow:

$$T_{i-1}^{\ i} = R(z_{i-1},\theta_i)T(z_{i-1},d_i)T(x_i,a_i)R(x_i,\alpha_i) \tag{7.7}$$

$$T_{i-1}^{\ i} = \begin{bmatrix} C\theta_i & -S\theta_i & 0 & 0 \\ S\theta_i & C\theta_i & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & a_i \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & C\alpha_i & -S\alpha_i & 0 \\ 0 & S\alpha_i & C\alpha_i & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} C\theta_i & -C\alpha_i S\theta_i & -S\alpha_i S\theta_i & a_i C\theta_i \\ S\theta_i & C\alpha_i C\theta_i & -S\alpha_i C\theta_i & a_i S\theta_i \\ 0 & S\alpha_i & C\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

with C and S representing respectively Cos and Sin, θ_i being the joint i angle around the axis z_{i-1}, α_i being the rotation around the axis $x_i, d_i$the translation along the axis z_{i-1} and a_i the translation along x_i.

The modified DH calculates them in the following manner:

$$T_{i-1}^{\ i} = R(x_{i-1}, \alpha_{i-1})T(x_{i-1}, a_i)R(z_i, \theta_i)T(z_i, d_i) \tag{7.8}$$

$$T_{i-1}^{\ i} = \begin{bmatrix} C\theta_i & -S\theta_i & 0 & a_{i-1} \\ S\theta_i C\alpha_{i-1} & C\theta_i C\alpha_{i-1} & -S\alpha_{i-1} & -d_i S\alpha_{i-1} \\ S\theta_i S\alpha_{i-1} & C\theta_i S\alpha_{i-1} & C\alpha_{i-1} & d_i C\alpha_{i-1} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

with θ_i being the joint i angle around the axis z_i, α_i being the rotation around the axis x_{i-1}, d_i the translation along the axis z_i and a_{i-1} the translation along x_{i-1}.

This is the difference between the classical and modified DH [67], the forward kinematic is calculated the same way in both methods. -Click here: click to return to text -Prot1-

seed: A random seed is a starting point when a computer generates a random number sequence. -Click here: . to get back to text -Prot2-

r-cortex (or cortex-r): The r-cortex stands for Regular spiking neurons in the cortex brain and the i-cortex stands for Fast spiking neurons on the cortex brain. The r-cortex is an excitatory firing while i-cortex is inhibitory (see Figure 4.15) -Click here: (to get back to text -Prot2-

Interspike Interval (ISI) : The interspike interval is the time between spikes, in this thesis it's the time between spikes of the same population. -Click here:; to get back to text -Prot3-

Kmean cluster: Really briefly, a Kmean cluster aims to partition all the observations into a k number of clusters in which each observation belongs to the cluster with the nearest mean (called "cluster centroid"), serving as a prototype of the cluster. -Click here:(to get back to text -Prot3-

Hidden layers: a hidden layer is located between the input and output of the algorithm, it corresponds to the green layers in Figure 4.24. The Hidden layers are used to perform nonlinear transformations on the inputs entered into the network -Click on the point to get back to text -Prot3- : .

Learning rate: The learning rate determines how big a step will be applied at each iteration while moving toward a minimum of a Loss function -Click on the point to get back to text -Prot3- : .

Local minima: A local minima in Neural Network (to explain it in an overly simplified way) is a solution in which the network is stuck that is not the best that could be found.

Note that this explanation is incomplete -Click on the point to get back to text -Prot3- : .

cross-validation: The goal of cross-validation is to test the model's ability to predict new data that was not used in estimating it, so to flag issues (such as over and underfitting). In most methods multiple rounds of cross-validation are performed using different partitions -Click here:) to get back to text -Prot3-

Bibliography

[1] P. N. Today, "Parkinson's disease statistics." https://parkinsonsnewstoday.com/parkinsons-disease-statistics/.

[2] B. Bilgiç, H. A. Hanağası, and M. Emre, "Parkinson's disease," *Neuropsychiatric and Cognitive Changes in Parkinson's Disease and Related Movement Disorders: Diagnosis and Management*, vol. 363, pp. 177–191, 2010.

[3] A. R. a. Hauser, C. Editor, and S. R. Benbadis, "Parkinson Disease Clinical Presentation.," *Medscape*, vol. 8, pp. 1–14, 2018.

[4] NHS, "Parkinson's disease overview." https://www.nhs.uk/conditions/parkinsons-disease/.

[5] M. M. Mcgregor and A. B. Nelson, "Review Circuit Mechanisms of Parkinson's Disease," *Neuron*, vol. 101, no. 6, pp. 1042–1056, 2019.

[6] A. L. Benabid, "Deep brain stimulation for parkinson's disease," *Current opinion in neurobiology*, vol. 13, no. 6, pp. 696–706, 2003.

[7] M. F. Bear, B. W. Connors, and M. A. Paradiso, *Exploring the Brain*. No. 3, 2012.

[8] P. N. Today, "Akinesia." https://parkinsonsnewstoday.com/parkinsons-disease-symptoms/motor/akinesia/.

[9] C. C. Aquino and S. H. Fox, "Clinical spectrum of levodopa-induced complications," *Movement Disorders*, vol. 30, no. 1, pp. 80–89, 2015.

[10] T. P. Ma and H. L. Geyer, "The Basal Nuclei," *Fundamental Neuroscience for Basic and Clinical Applications: Fifth Edition*, pp. 377–393.e1, 2018.

[11] V. Rajshekhar *et al.*, "Whither lesional surgery for movement disorders," *Neurology India*, vol. 54, no. 3, p. 241, 2006.

[12] M. S. Okun, "Deep-brain stimulation for parkinson's disease," *New England Journal of Medicine*, vol. 367, no. 16, pp. 1529–1538, 2012.

[13] C. L. Tomlinson, S. Patel, C. Meek, C. P. Herd, C. E. Clarke, R. Stowe, L. Shah, C. M. Sackley, K. H. Deane, K. Wheatley, *et al.*, "Physiotherapy versus placebo or no intervention in parkinson's disease," *Cochrane database of systematic reviews*, no. 9, 2013.

[14] B. Den, "What are neurotransmitters and neuroactive peptides?." http://www.biochemden.com/neurotransmitters-neuropepptides/.

[15] M. Gillies, J. Hyam, A. Weiss, C. Antoniades, R. Bogacz, J. Fitzgerald, T. Aziz, M. A. Whittington, and A. L. Green, "The cognitive role of the globus pallidus interna; insights from disease states," *Experimental brain research*, vol. 235, no. 5, pp. 1455–1465, 2017.

[16] A. Bischoff *et al.*, *Modeling the basal ganglia in the control of arm movements*. Citeseer, 1998.

[17] T. J. Prescott, F. M. Montes González, K. Gurney, M. D. Humphries, and P. Redgrave, "A robot model of the basal ganglia: Behavior and intrinsic processing," *Neural Networks*, vol. 19, no. 1, pp. 31–61, 2006.

[18] P. Greengard, "The Neurobiology of Slow Synaptic Transmission," *Science*, vol. 294, pp. 1024 LP – 1030, nov 2001.

[19] V. Fernández, C. Llinares-Benadero, and V. Borrell, "Cerebral cortex expansion and folding: what have we learned?," *The EMBO journal*, vol. 35, no. 10, pp. 1021–1044, 2016.

[20] S. Schwerin, "The anatomy of movement." https://brainconnection.brainhq.com/2013/03/05/the-anatomy-of-movement/.

[21] S. Betti, G. Zani, S. Guerra, U. Castiello, and L. Sartori, "Reach-to-grasp movements: A multimodal techniques study," *Frontiers in Psychology*, vol. 9, no. JUN, pp. 1–10, 2018.

[22] n. HMCDA Skilled, "How does the brain produce movement?,"

[23] M. Arbib, P. Verschure, and U. Seifert, *Action, Language, and Music: Events in Time and Models of the Brain*, p. 357–391. 07 2013.

[24] C. Begliomini, T. De Sanctis, M. Marangon, V. Tarantino, L. Sartori, D. Miotto, R. Motta, R. Stramare, and U. Castiello, "An investigation of the neural circuits underlying reaching and reach-to-grasp movements: from planning to execution," *Frontiers in human neuroscience*, vol. 8, p. 676, 2014.

[25] K. Nelissen, P. A. Fiave, and W. Vanduffel, "Decoding grasping movements from the parieto-frontal reaching circuit in the nonhuman primate," *Cerebral Cortex*, vol. 28, no. 4, pp. 1245–1259, 2018.

[26] E. M. R. M. H. S. M. A. M. E. J. Huesler, M.-C. Hepp-Reymond, and J. R. F. A. W. W. P. H. Friden, "Hand and brain- The Neurophysiology and Psychology of Hand Movements," no. 329, pp. 100–120, 1998.

[27] G. Rizzolatti and G. Luppino, "The cortical motor system," 2001.

[28] K. Nelissen, P. A. Fiave, and W. Vanduffel, "Decoding grasping movements from the parieto-frontal reaching circuit in the nonhuman primate," *Cerebral Cortex*, vol. 28, no. 4, pp. 1245–1259, 2018.

[29] J. Prodoehl, D. M. Corcos, and D. E. Vaillancourt, "Basal ganglia mechanisms underlying precision grip force control," *Neuroscience and Biobehavioral Reviews*, vol. 33, no. 6, pp. 900–908, 2009.

[30] U. Castiello, "The neuroscience of grasping," *Nature Reviews Neuroscience*, vol. 6, no. 9, pp. 726–736, 2005.

[31] N. F. . al., *Novartis Foundation Symposium 218 - Sensory Guidance of Movement*. 2007.

[32] U. Castiello, K. Bennett, C. Bonfiglioli, S. Lim, and R. F. Peppard, "The reach-to-grasp movement in Parkinson's disease: Response to a simultaneous perturbation of object position and object size," *Experimental Brain Research*, vol. 125, no. 4, pp. 453–462, 1999.

[33] K. Gurney, T. J. Prescott, J. R. Wickens, and P. Redgrave, "Computational models of the basal ganglia: from robots to membranes," *Trends in neurosciences*, vol. 27, no. 8, pp. 453–459, 2004.

[34] W. Gerstner, "Spiking neurons," tech. rep., MIT-press, 1998.

[35] N. dynamics, "Mean firing rate." https://neuronaldynamics.epfl.ch/online/Ch7.S2.html.

[36] K. Kumaravelu, D. T. Brocker, and W. M. Grill, "A biophysical model of the cortex-basal ganglia-thalamus network in the 6-OHDA lesioned rat model of Parkinson's disease," 2016.

[37] R. A. B. D. S. P. V. Ranieri, Moioli, "Unveiling Parkinson ' s Disease Features from a Primate Model with Deep Neural Networks,"

[38] S. O'Connell, "What the tortoise taught us." https://www.theguardian.com/science/2000/dec/07/robots.

[39] J. L. Krichmar, "Neurorobotics—a thriving community and a promising pathway toward intelligent cognitive robots," *Frontiers in Neurorobotics*, vol. 12, p. 42, 2018.

[40] J.-r. King, L. Gwilliams, C. Holdgraf, J. Sassenhagen, A. Barachant, D. Engemann, E. Larson, A. Gramfort, J.-r. King, L. Gwilliams, C. Holdgraf, J. Sassenhagen, and A. Barachant, "Encoding and Decoding Neuronal Dynamics : Methodological Framework to Uncover the Algorithms of Cognition," 2018.

[41] S. F. Giszter, K. A. Moxon, I. A. Rybak, and J. K. Chapin, "Neurobiological and neurorobotic approaches to control architectures for a humanoid motor system," *Robotics and Autonomous Systems*, vol. 37, no. 2-3, pp. 219–235, 2001.

[42] G. Massera, A. Cangelosi, and S. Nolfi, "Evolution of prehension ability in an anthropomorphic neurorobotic arm," *Frontiers in Neurorobotics*, vol. 1, p. 4, 2007.

[43] E. V. D. Wal, "Object Grasping with the NAO Egbert van der Wal," 2012.

[44] S. Eskiizmirliler, N. Forestier, B. Tondu, and C. Darlot, "A model of the cerebellar pathways applied to the control of a single-joint robot arm actuated by mckibben artificial muscles," *Biological cybernetics*, vol. 86, no. 5, pp. 379–394, 2002.

[45] S. Nolfi, D. Floreano, and D. D. Floreano, *Evolutionary robotics: The biology, intelligence, and technology of self-organizing machines*. MIT press, 2000.

[46] C. R. Holdgraf, J. W. Rieger, C. Micheli, S. Martin, R. T. Knight, and F. E. Theunissen, "Encoding and decoding models in cognitive electrophysiology," *Frontiers in Systems Neuroscience*, vol. 11, p. 61, 2017.

[47] J. Bahuguna, P. Weidel, and A. Morrison, "Exploring the role of striatal d1 and d2 medium spiny neurons in action selection using a virtual robotic framework," *European Journal of Neuroscience*, vol. 49, no. 6, pp. 737–753, 2019.

[48] M. Kawato, "Internal models for motor control and trajectory planning.," *Curr Opin Neurobiol*, vol. 9, pp. 718–27, 01 1999.

[49] B. M. Yu, C. Kemere, G. Santhanam, A. Afshar, S. I. Ryu, T. H. Meng, M. Sahani, and K. V. Shenoy, "Mixture of Trajectory Models for Neural Decoding of Goal-Directed Movements," pp. 3763–3780, 2020.

[50] A. E. Brockwell, A. L. Rojas, and R. E. Kass, "Recursive Bayesian Decoding of Motor Cortical Signals by Particle Filtering," vol. 2, pp. 1899–1907, 2020.

[51] P. Cisek, *Internal Models*, pp. 2009–2012. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009.

[52] www.cs.cmu.edu, "Inverse models." https://www.cs.cmu.edu/~schneide/tut5/node51.html.

[53] A. documentation, "index_robots." http://doc.aldebaran.com/2-1/family/robots/index_robots.html.

[54] D. Gouaillier, V. Hugel, P. Blazevic, C. Kilner, J. Monceaux, P. Lafourcade, B. Marnier, J. Serre, and B. Maisonnier, "Mechatronic design of NAO humanoid," pp. 769–774, 2009.

[55] NetPyne.org, "About netpyne." http://www.netpyne.org/about.html.

[56] ros.org, "About ros." https://www.ros.org/about-ros/.

[57] cyberbotics, "Webots user guide." https://cyberbotics.com/doc/guide/foreword.

[58] Klaim, "Installation issue with the naoqi python sdk on ubuntu." https://github.com/aldebaran/libqi/issues/28.

[59] N. Kofinas, E. Orfanoudakis, and M. G. Lagoudakis, "Complete analytical forward and inverse kinematics for the nao humanoid robot," *Journal of Intelligent & Robotic Systems*, vol. 77, no. 2, pp. 251–264, 2015.

[60] aldebaran, "control-cartesian-api." http://doc.aldebaran.com/2-1/naoqi/motion/control-cartesian-api.html.

[61] G. Schillaci, V. V. Hafner, and B. Lara, "Coupled inverse-forward models for action execution leading to tool-use in a humanoid robot," in *2012 7th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, pp. 231–232, IEEE, 2012.

[62] M. Pimentel, "Toward a neurorobotics model of parkinson disease (tbc)," *Frontiers in Neurorobotics*, Under review.

[63] O. Jitkritsadakul, P. Jagota, and R. Bhidayasiri, "Pathophysiology of parkinsonian tremor: a focused narrative review," *Asian Biomedicine*, vol. 10, no. s1, pp. s15 – s22, 2017.

[64] C. Hess and S. Pullman, "Tremor: Clinical phenomenology and assessment techniques," *Tremor and other hyperkinetic movements (New York, N.Y.)*, vol. 2, 06 2012.

[65] S. C. class, "Cs231n: Convolutional neural networks for visual recognition." https://cs231n.github.io/neural-networks-3/.

[66] scikit learn, "scikit-learn explanation menu." https://scikit-learn.org/.

[67] A. C. Reddy, "Difference between denavit-hartenberg (dh) classical and modified conventions for forward kinematics of robots with case study," in *International Conference on Advanced Materials and manufacturing Technologies (AMMT)*, JNTUH College of Engineering Hyderabad Chandigarh, India, 2014.