POLITECNICO DI MILANO

MASTER THESIS

Lower-limb non-invasive sensory neuroprosthesis: real-time data visualization and calibration

Submitted by:
Michele MARAZZI
873616

Advisor:
Prof.ssa Alessandra Laura Giulia PEDROCCHI,
Politecnico di Milano
Co-Advisor:
Prof. Stanisa RASPOPOVIC,
ETH Zurich
Supervisor:
PhD Francesco PETRINI,
ETH Zurich

Neuroengineering Lab – at ETH Zurich
NEARLab - Neuroengineering and medical robotics – at Politecnico di Milano

Dipartimento di Elettronica, Informazione e Bioingegneria

Academic year 2017-2018

POLITECNICO DI MILANO

Abstract

Dipartimento di Elettronica, Informazione e Bioingegneria

Masters of Science in Biomedical Engineering

Lower-limb non-invasive sensory neuroprosthesis: real-time data visualization and calibration

by Michele MARAZZI

Sensory-feedback neuroprosthesis are the new frontier of supporting devices for amputees. The concept is to restore the missing sensation via electrical-nerve stimulation. This increases the embodiment of the prosthesis, adds the possibility to modulate the applied force, lowers the cognitive burdensome and increases the trustfulness towards the device.

Invasive stimulation methods have been successfully tested on upper-limb amputees and lower-limb amputees, obtaining somatotopic evoked sensations. These methods requires surgery to implant electrodes, which are then subjected to encapsulation and migration.

A non-invasive method to elicit sensations in patients is to use superficial-electrical stimulation (TENS); this idea has been tested with good results on upper-limb amputees.

The topic of this thesis is to provide the basis to apply the same concept to lower-limb amputees and diabetic peripheral neuropathy patients. The sensory-feedback restoration should be driven by sensorized insoles, thus a Real-Time GUI to visually evaluate the values of the sensors and the corresponding stimulation has been developed. The concept of TENS to evoke somatotopic sensations requires highly patient-dependent mapping (i.e.: association between electrodes positions, stimuli settings, evoked distal-sensations and location of these). For this reason, a Mapping Software has been developed and validated on volunteers.

POLITECNICO DI MILANO

Abstract – Italian

Dipartimento di Elettronica, Informazione e Bioingegneria

Laurea Magistrale in Ingegneria Biomedica

Calibrazione e controllo in tempo reale, di una neuroprotesi sensoriale non-invasiva per gli arti inferiori

by Michele MARAZZI

Le neuroprotesi in grado di restituire il feedback sensoriale sono il futuro per quanto riguarda il supporto agli amputati. Il concetto è di restituire la sensazione mancante tramite stimolazione elettrica dei nervi. Ripristinare il feedback sensoriale comporta un percepire maggiormente la protesi come fosse un proprio arto; introduce la possibilità di modulare la forza applicata, diminuisce il carico cognitivo, ed aumenta la sicurezza nei confronti della protesi.

Metodi di stimolazione elettrica invasivi sono stati testati con successo su amputati agli arti superiori ed inferiori, ottenendo sensazioni evocate somatotopiche. Questi metodi richiedono operazioni chirurgiche per impiantare gli elettrodi, i quali subiscono gli effetti di migrazione e vengono ricoperti da tessuto cicatrizzato.

Un metodo non invasivo per evocare sensazioni nei pazienti è quello di utilizzare stimolazione elettrica transcutanea; l'idea è stata testata con successo per pazienti amputati agli arti superiori.

Il tema di questa tesi è quello di realizzare gli strumenti di base per applicare lo stesso concetto di stimolazione non-invasiva anche a pazienti amputati agli arti inferiori, e pazienti con neuropatia diabetica. Il feedback sensoriale implementato dovrebbe essere posto in relazione alla pressione letta da sensori posti sotto una suola. Per questo motivo è stata sviluppata una interfaccia grafica a tempo-reale, in grado di mostrare contemporaneamente la lettura dei sensori e la stimolazione applicata. Il concetto di stimolazione trans-cutanea per evocare sensazioni somatotopiche richiede fortemente un "mapping" che cambia da paziente a paziente. Tale "mapping" consiste nell'associare le posizioni degli elettrodi, le configurazioni di stimolazione, le percezioni sentite e le loro posizioni. Per questa ragione, un software per il Mapping è stato sviluppato e validato su volontari.

Acknowledgements

Vorrei ringraziare la Professoressa Alessandra Pedrocchi per la sua pazienza e disponibilità, e per avermi reso possibile intraprendere questa incredibile ed indimenticabile esperienza. Il Professore Stanisa Raspopovic ed il PhD Francesco Petrini per il loro ruolo nell'avermi guidato durante la realizzazione del progetto, e tutti i membri del Sensory Motor-Systems Lab. La mia collega Oriella Gnarra per avermi sopportato durante tutti questi mesi, ed aver offerto un'ottima amicizia. Infine, la mia famiglia per avermi sempre dato supporto, sempre spinto a resistere alle avversità ed a dare il massimo; per aver sempre creduto in me.

Grazie,
Michele

Contents

List of Figures

To my family.

Extended Summary

Introduction

Low-limb amputees and diabetic peripheral neurophaty patients lose the sensory-feedback that is physiologically present from the foot. This causes for diabetics the possibility to develop ulcers, leading to amputation. While for amputees it means a reduced gait-symmetry, low-embodiment of the prosthesis, low walking speed and distrust of the prosthesis [45] [11].

Sensory-feedback neuroprosthesis are prosthesis able to artificially restore the missing sensation, using electrical stimulation. Successful Proof of Concepts have been made for upper-limb amputees, using both invasive [37] and non-invasive [12] methods. Providing a somatotopic sensory-feedback via non-invasive stimulation for lower-limb neuroprosthesis is instead an unexplored field.

Scope of work

The aim of this thesis is to provide the necessary blocks to implement a non-invasive sensory-feedback nueroprosthesis, for lower-limb. Using an already developed sensorized insoles, a superficial electrical stimulation should be driven by the force values.

For this reason a graphical interface to control the neuroprosthesis and visualize data in **real-time** needs to be developed, allowing an easy and intuitive evaluation of the system, and providing the possibility to modify neuroprosthesis parameters online.

A Mapping software to **calibrate** the prosthesis is also needed: this would allow the mandatory step to associate electrodes positions, stimuli parameters, and evoked sensations. These evaluations are highly subject dependent due to the intrinsic anatomical variabilities and can not be taken for granted.

Lastly, a working **closed-loop system** should correctly modulate the electrical stimuli, depending on the read force, with an unperceivable delay to the user.

Methods

Real-Time system to control and visualize data from the neuroprosthesis

As the aim of this thesis was to provide the *basis* to implement a sensory-feedback neuroprosthesis for lower-limb amputees, a **sensorized insole** was used to drive electrical stimulation on the subject. To correctly evaluate the functioning of the system, a first step has been the development of a Graphical User Interface that plotted the sensors readout and stimulation applied in **Real-Time**. The GUI is a Client that can remotely connect to the system-controller via WiFi or Ethernet cable communication, or be run directly on the system-controller. It's completely written in C++11 with Qt libraries.

To be able to plot data in Real-Time required a complicated **multithread** architecture of the Software implemented. Each Thread has a different aim, the technical details will be explain in Chapter 3. As a general concept, one is used for the networking operations needed to read data and one for post-processing and data display on the charts.

This meant an extensive preliminary planning of the various interconnection between the Threads, because the access to variables has to be regulated: otherwise the behavior of the system might be unpredictable.

Furthermore it is also required to use a signal event handling to "*talk*" between Threads. An example of such operation would be the Network Thread that signalize the data-display Thread that new data has been ready.

Algorithms were used, ensuring that:

- The delay due to data transmission is the minimum possible and is imperceptible to the user.
- There's no accumulation of Lag.

The obtained system provides a Real-Time display of the forces sensed by the sensorized insole, with an average delay of: $14.1ms \pm 9.1ms$[1]. Only $1ms$ is due to the Network transmission to the Client. More information about this evaluation can be found on Chapter 6.

The Software provides also the possibility to change the stimuli parameters online, for example: maximum current, minimum current and associated electrodes.

It was also extended to add a control system to help improve gait symmetry in amputees: a negative audio feedback is produced each time an online algorithm evaluates that the exerted force is too low. Although this **weigh-balance controller** is ready to be tested, no evaluation has been made on patients.

The image in Figure 1 shows the main window of the Client-GUI software developed.

A complete description of the system is presented on Chapter 3, but a first summary of the main features implemented is:

1. Remote
   Local connection to a system-controller.
2. Auto-save of data read and stimuli used.
3. Auto-save of the chosen user-settings.
4. Auto-save of the stimuli settings, with the possibility to load an external file.
5. Scroll of the Charts to see precedent values.

[1]This value is obtained from the Chapter 6 as the sum of the Bluetooth delay (Arduino to Odroid/Raspberry) and the network-transmission delay to the Client (Odroid/Raspberry to Remote Client.

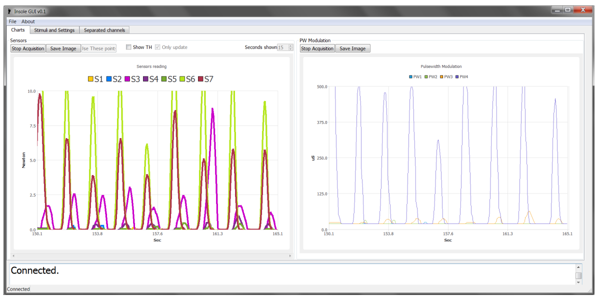


FIGURE 1: Client GUI: Main Window.

6. Possibility to Activate/Deactivate different sensors.
7. Time-out and lag control algorithms.
8. Weight-balance feedback, based on a two-threshold system, unique to each sensor.

Mapping Software

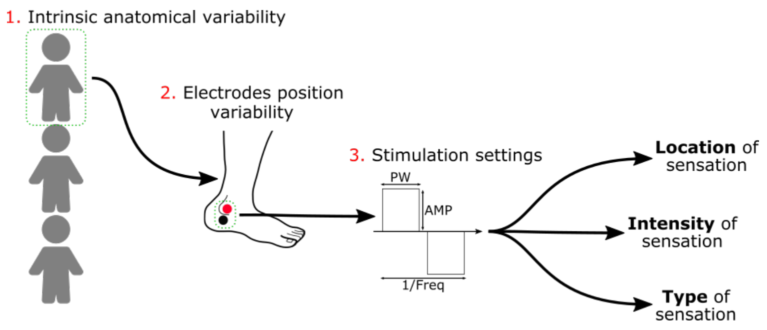

1. Intrinsic anatomical variability
2. Electrodes position variability
3. Stimulation settings
PW
AMP
1/Freq
Location of sensation
Intensity of sensation
Type of sensation

FIGURE 2: Concept of mapping: variability associated with the stimulation.

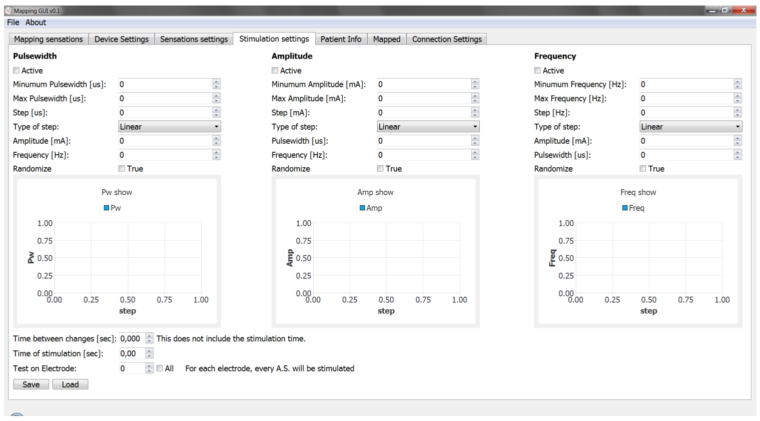
In order to have a functional and useful neuroprosthesis, the TENS[2] used on the patient should evoke distal-sensation in a wanted position, providing a known feeling both in terms of type and intensity.

Due to the inevitable intrinsic anatomical variability between different subjects, assumptions on the elicited sensations can't be made. Instead a **mapping** step is required.

This step consists in associating:

- Electrodes positions.
- Stimulus provided: Pulsewidth – Current amplitude – Frequency.
- In-situ sensation intensity.
- Type of distal-sensation.
- Intensity of the distal-sensation.
- Location of the distal-sensation.

[2]Transcutaneous Electrical Nerve Stimulator.



Pulsewidth
☐ Active
Minumum Pulsewidth [us]: 0
Max Pulsewidth [us]: 0
Step [us]: 0
Type of step: Linear
Amplitude [mA]: 0
Frequency [Hz]: 0
Randomize ☐ True

Amplitude
☐ Active
Minumum Amplitude [mA]: 0
Max Amplitude [mA]: 0
Step [mA]: 0
Type of step: Linear
Pulsewidth [us]: 0
Frequency [Hz]: 0
Randomize ☐ True

Frequency
☐ Active
Minumum Frequency [Hz]: 0
Max Frequency [Hz]: 0
Step [Hz]: 0
Type of step: Linear
Amplitude [mA]: 0
Pulsewidth [us]: 0
Randomize ☐ True

Time between changes [sec]: 0,000 This does not include the stimulation time.
Time of stimulation [sec]: 0,00
Test on Electrode: 0 ☐ All For each electrode, every A.S. will be stimulated

FIGURE 3: Mapping software: Stimuli settings.

The **in-situ** sensation is the feeling *under* the electrodes perceived by the Subject. Ideally this should be absent, because the aim is to artificially restore a *somatotopic* feedback: the sensation should be felt only in the missing, or damaged, limb. The in-situ feeling, if the electrodes positions are wrong, is uncomfortable even at low levels of stimuli and is perceived as pain. Otherwise it is felt as a vibration, and depending on the electrodes positions, it may be low even increasing the stimuli intensities.

The **distal-sensation** is instead the evoked feeling that is pursued. It's *distal* to respect the position of the electrodes. Achieving distal-sensations means that the nerve fibers are correctly being stimulated. For example, stimulating the tibial nerve at the ankle level means that some distal-sensations in the foot should be perceived by the Subject. For this thesis work, the type of sensation used by Subjects to describe the perception felt is a mix of *vibration* and *tingling*. This has been the choice for every Subject evaluated, and thus, in the preliminary results presented, *distal-sensation intensity* will always refer to this kind of feeling.

The mapping procedure should be repeated for each electrode position, for each subject, and for a whole sweep of the stimuli-variable chosen as modulation. For example if the feedback intensity is encoded in terms of pulsewidth, to have a mapping result the procedure should be done for many values of it, ranging from a minimum to a maximum.

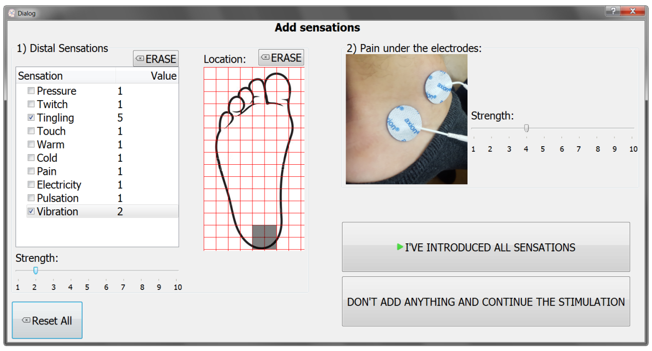
Figure 2 displays the concept of mapping, necessary due to the high variability associated with the evoked sensation.

A Software that manages to provide mapping functionality has been developed by scratch using C++ and Qt libraries. A validation of its functionality has been made on healthy volunteers, doing non-invasive stimulation on the ankle, stimulating the Tibial Nerve via surface electrodes.

Images 3, 4 show the main windows of the Software.

A short summary of the implemented functions:

- Subjects database with personalized settings.
- Real-time graphs showing the electrical stimulation exerted.
- Possibility to stimulate with the RehaStim device.



Sensation	Value
☐ Pressure	1
☐ Twitch	1
☑ Tingling	5
☐ Touch	1
☐ Warm	1
☐ Cold	1
☐ Pain	1
☐ Electricity	1
☐ Pulsation	1
☑ Vibration	2

FIGURE 4: Mapping software: user view.

- Various types of mapping: linear ordered step, randomized linear step, discrimination tests.
- Possibility to sweep modulation of pulsewidth, current amplitude and frequency.
- A sensation-window proposed to the Subject, that allows selecting: sensations felt with an intensity from 1 to 10, in-situ intensity and distal-location.

Protocol for validation of the mapping software

A validation of the system has been made on four healthy volunteers, stimulating using superficial electrodes over the ankle, in order to induce spiking in the Tibial Nerve and consequently distal-sensations under the foot.

A first exploration of many different electrodes position has been made, to obtain an initial starting point of useful positions. These are the electrodes location that are able to elicit distal-sensations and that do not cause an excessive discomfort in-situ. The useful range is visible on Image 5. In this case the exploration has been done keeping the positive electrode fixed on the medial malleolus, and testing different positions for the negative one.

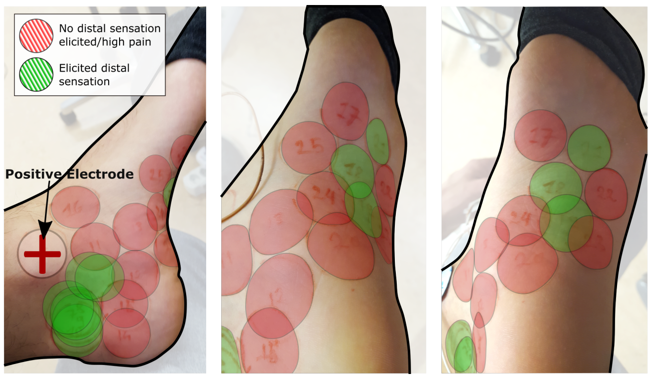

No distal sensation elicited/high pain
Elicited distal sensation
Positive Electrode

FIGURE 5: Exploration of useful electrodes placements.

Once a first starting point was obtained, the mapping was done on healthy volunteers. The mapping results has been associated to an ankle position, because relative movement between the superficial electrodes and the internal nerves change the distal-location affected.

A summary of the used **protocol** is as follows:

1. The interested skin zones (i.e.: the ankle) were washed and if hairs were present clipped.
2. The positive electrode is placed on the malleolus, while the negative is placed on one of the proposed positions. To obtain comparable results, 4 main positions were used as reference. These are visible on the left part of the image 6.
3. After making sure the Subject feels comfortable with the electrical stimulation, the first test begins.
4. An ankle position is chosen, between: stance, lifted, push-off, heelstrike.
5. A linear increase in pulsewidth stimulation is used, starting from 20us, with steps of 10us every 3 seconds. The frequency will always be 80Hz, and the Current 2mA.
6. If the subject reports too much discomfort before reaching 100us, the position is changed.
7. The subject is asked to report when it's the first time that he feels something distal, and whenever the stimulation has become too much uncomfortable. These will be used as minimum and maximum values for the range.
8. The subject is trained about the "adding sensation" window, explaining the different types of sensation and what he'll be asked about (in-situ intensity, distal intensity, distal-location).
9. The mapping begins: the pulsewidth is made change between the minimum and the maximum found, in a random shuffled way. The minimum step between values is 20us.
10. Every time a stimulus occurs, if the subject feels anything, he is asked to compile the "Adding sensations" window.
11. Once every stimulus has been provided, the procedure stops and the choices are saved with their associated stimuli-settings.

Results

Validation of the mapping software

Four healthy volunteers have participated in these preliminary tests. Figure 6 shows the mapping results for Subject1. Figure 7 compares the evoked distal-location for different Subjects, for similar electrodes positions and similar ankle angles.

An example of repeatability for the same subject is shown in Figure 8.

The intensity of the distal sensation for various pulsewidth is presented on Image 9a, with the in-situ discomfort being on Figure 9b.

These results highlight that the intra-subject results are repeatable, while the extra-subject comparison are highly different due to intrinsic anatomical differences; this was expected, and confirms that a mapping software is an unavoidable necessity.

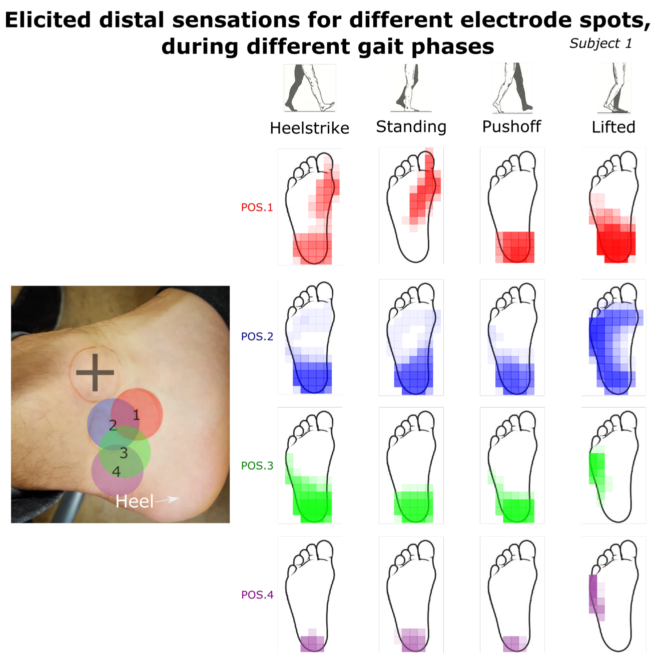

Elicited distal sensations for different electrode spots, during different gait phases
Subject 1
Heelstrike
Standing
Pushoff
Lifted
POS.1
POS.2
POS.3
POS.4
1
2
3
4
Heel

FIGURE 6: Mapping results for Subject 1.

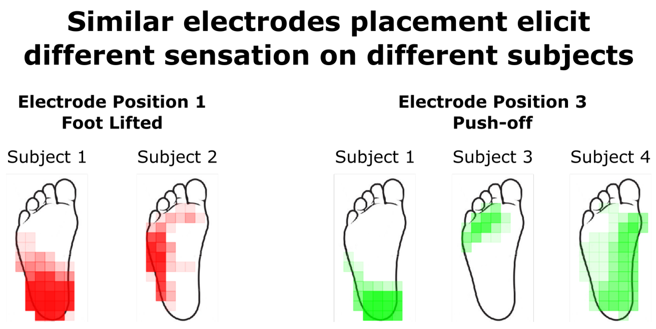


FIGURE 7: Inter-subject comparison.

The results shows that the subjects are able to correctly associate an increase in the stimuli provided, with an increase in distal sensations. The increase in stimuli causes an increase in distal-perception, while the discomfort directly under the electrodes (in-situ) saturates to an acceptable value.

The complete results obtained are shown and explained in details in Chapter 7, with also more emphasis on anatomical descriptions. Tests were made also to determine if inverting the electrodes polarity affected the sensations.

The Weber coefficient was also extrapolated from discrimination tests, obtaining 0.10 ± 0.01. More details are provided in Chapter 7.

Closed-loop system

The neuroprosthesis should act as a closed-loop: the force is sensed in the insole, sent via Bluetooth to a system-controller, which processes it and generates accordingly an electrical stimulation. The more the force, the more the electrical stimulation, which in our case is a modulation of the generated pulsewidth. The transfer function is a

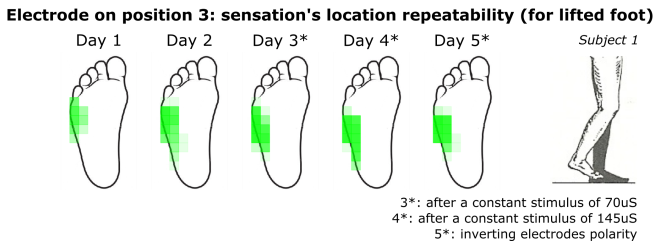
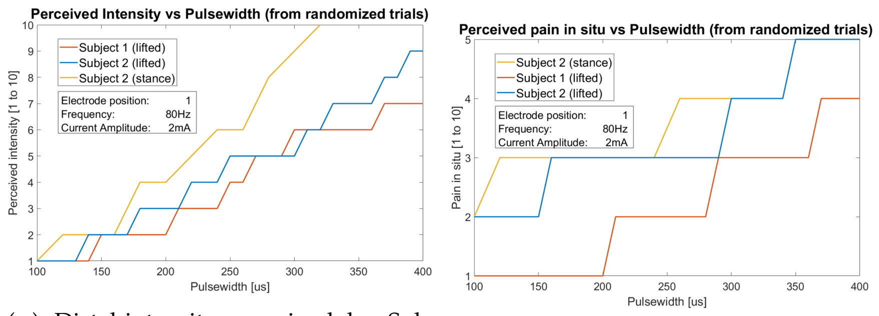

Electrode on position 3: sensation's location repeatability (for lifted foot)
Day 1
Day 2
Day 3*
Day 4*
Day 5*
Subject 1
3*: after a constant stimulus of 70uS
4*: after a constant stimulus of 145uS
5*: inverting electrodes polarity

FIGURE 8: Repeatability for Subject1, electrode position 3.



(A) Distal-intensity perceived by Subject 1 and Subject 2 increasing the pulsewidth.

(B) In-situ pain perceived by Subject 1 and Subject 2 increasing the pulsewidth.

FIGURE 9: Perceived intensities versus applied pulsewidth.

linear mapping between force and pulsewidth. As the results have shown, an increase in pulsewidth is associated with an increase in the distal-sensation perceived, so the base concept is correct.

In this work, a closed-loop system ready to be tested with real patients, has been developed.

The force is sensed using a sensorized insole, and the data is sent to a system-controller that in turns controls a TENS to induce sensations in the Subject. As Figure 10 shows, the system acts as a closed-loop: this stimulation will influence the Subjects actions, therefore the exerted force, varying the artificially-induced sensory feedback.

A multithread software, running on the system-controller, has been developed as

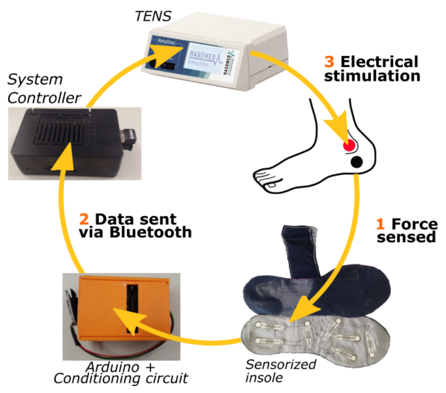


FIGURE 10: Concept of neuroprosthesis, example case of diabetic patient.

part of this thesis. The obtained result produces a modulation of the stimuli generated in real-time, depending on the force. In the Appendix A a link to a video showing a test of the system is present.

The multithread implementation achieves security from failures, and an high performance. The closed-loop has an average worst-case-scenario of the global-delay of 25.4ms, which is unperceivable by the user.

Conclusions

The Client-GUI developed achieves the possibility to evaluate force data in **Real-Time**, and to change the stimuli settings online. The Client can remotely connect to the system-controller, allowing the data to be seen using any common laptop, PC or even smartphones. A high performance is obtained with a multithread implementation and no lag is accumulated. An audio feedback to increase the SI[3] in gait has also been implemented.

The main limitation of the system is that the graphical display and update of charts might be a heavy graphical process, this is an intrinsic limitation with the graphical libraries used. That said, no suitable alternative that could produce better results has been found. Furthermore, the weight-balance control needs to be tested with patients to see if the implementation correctly produces the feedback when needed, and if this increases the SI.

Improvements of the Client should be on creating a more advanced weight-balance control, using a machine-learning approach to identify wrong strides, or evaluating also the first derivatives of the sensed-force. A second sensorized insole could be put in the other shoes to compare the two stance phases.

A **Mapping software** provides the possibility to correctly calibrate the neuroprosthesis, which is seen as a mandatory step, due to the many variables associated with the system. Some preliminary tests have been made, validating the functionality of the software and its applicability, but also providing insights in an unexplored area.

The software could be improved introducing the possibility to stimulate with more complex patterns, for example exponential increase of the stimuli. It could also be added an automated system to select the best electrodes positions.

A complete working **closed-loop system** has been implemented. The system provides a *TENS*[4] with pulsewidth-modulation driven by force sensors. Multiple output channels can be used concurrently, associating them to different sensors. The global delay of the system has been evaluated to be under 30ms: less than one-third of the suggested 100ms by literature [15].

From a technological point of view the closed-loop system is ready to be tested as a non-invasive sensory-feedback neuroprosthesis, but it can not be ignored the important limitation of the system: it has not been tested with real patients.

A strong and extensive testing should be made on patients, to identify first if a distal-sensation can be elicited, and if so, which electrodes locations are useful. A more

[3]Symmetry Index.
[4]The device used is the RehaStim.

portable electrical stimulator should be used in order to allow testing the closed-loop on patients, while they are walking. Leading to the ultimate goal: the actual evaluation of how much impact this system would have on patients, in terms of gait symmetry, walking speed, embodiment of the prosthesis, confidence in the prosthesis and PLP[5] variations.

Future implementations and tests could also involve the use a different size of electrodes, that could result in a better accuracy in the locations of the evoked sensations.

In conclusion, this thesis work has provided the implementation of many parts of a non-invasive sensory-feedback neuroprosthesis, up to obtain a working closed-loop system. A technological validation has been made on each part developed. Moreover, some preliminary tests on volunteers have posed the basis to characterize the stimulation of the tibial nerve, with electrodes positioned over the ankle. No study in literature has yet explored this field, but the results, even if preliminary, suggest that the implementation of such system is indeed possible, and in the future researches with real patients will be performed.

Structure of the thesis

This thesis is organized as follows:

1. **Chapter 1** – *Context and aim of the project*: the introductory concept of sensory-feedback and why it is needed is provided, describing a possible basic closed-loop neuroprosthesis.
2. **Chapter 2** – *Starting point and missing parts*: as this project is a branch from a past work, the blocks that were already available are discussed and their limitations explained.
3. **Chapter 3** – *Real-Time GUI for insole-based applications with weight-balance controller*: the development of a software able to remotely control the neuroprosthesis, and display data in real-time is discussed.
4. **Chapter 4** – *Integration of the TENS*: the electrical stimulator used is RehaStim, a non-invasive device, and in this chapter is introduced as a main block of the lower-limb neuroprosthesis.
5. **Chapter 5** – *Mapping Software*: the development of a software able to calibrate a neuroprosthesis to a patient is discussed.
6. **Chapter 6** – *Global System*: the complete closed-loop system is discussed, and an evaluation of the delays is presented.
7. **Chapter 7** – *Preliminary validation of the platform*: the Mapping Software is tested on four healthy Subjects, providing a validation of the technological implementation, and insights into evoking somatotopic sensations in lower-limbs.
8. **Chapter 8** – *Discussion and future developments*: the main limitations of the various blocks are discussed, and possible improvements are proposed.

[5]Phantom Limb Pain.

Sommario Esteso

Introduzione

Amputati agli arti inferiori e diabetici che soffrono di neuropatia periferica perdono il feedback sensoriale dal piede, che è fisiologicamente presente. Questo causa per i diabetici la possibilità di sviluppare ulcere e successivamente dover amputare l'arto. Mentre per amputati, significa una riduzione della simmetria del passo, una bassa personificazione della protesi, una bassa velocità di cammino ed una bassa fiducia nella protesi utilizzata [45] [11].

Le neuroprotesi a feedback sensoriale permettono di restituire la sensazione mancante, utilizzando stimolazione elettrica. Proof of Concept funzionanti sono stati svolti per amputati agli arti superiori, utilizzando sia una stimolazione invasiva [37] che non-invasiva [12]. Invece, ottenere un feedback sensoriale somatotopico tramite stimolazione elettrica non-invasiva per gli arti inferiori, è ancora un campo inesplorato.

Scopo del lavoro

Lo scopo di questa tesi è quello di realizzare i blocchi necessari per l'implementazione di una neuroprotesi a feedback sensoriale, non-invasiva, per arti inferiori. Utilizzando una suola sensorizzata già sviluppata, una stimolazione elettrica superficiale dovrebbe essere modulata dalla lettura delle pressioni.

Per queste ragioni, una interfaccia grafica in grado di controllare la neuroprotesi, e di visualizzare pressioni e stimolazioni in tempo-reale, è necessaria. Il suo utilizzo permetterebbe una valutazione intuitiva ed immediata del corretto funzionamento del sistema, e la possibilità di variare i parametri della neuroprotesi sul momento.

Anche un software di calibrazione è necessario: lo step di associare le posizioni degli elettrodi, i parametri di stimolazione e le sensazioni evocate è assolutamente necessario. Queste valutazioni dipendono fortemente da paziente a paziente, data la variazione anatomica intrinseca nei soggetti, e non possono essere ignorate per ottenere un sistema funzionale.

Per ultimo, un sistema ad anello chiuso dovrebbe modulare l'intensità degli stimoli elettrici, associandoli al valore letto di forza, il tutto con un ritardo impercettibile all'utente.

Metodi

Software a tempo-reale per la visualizzazione dei dati ed il controllo di una neuroprotesi

Dato che lo scopo di questa tesi era di ottenere le basi per l'implementazione di una neuroprotesi a feedback sensoriale per arti inferiori, una suola con sensori di forza è stata utilizzata per modulare l'intensità degli stimoli elettrici generati sul soggetto. Per valutare correttamente il funzionamento del sistema, un primo step è stato lo sviluppo di una interfaccia grafica che potesse mostrare i dati della forza letti in tempo-reale, insieme alla stimolazione generata. L'interfaccia utente è un Client che si può collegare da remoto alla neuroprotesi, via WiFi, cavo Ethernet, o può essere avviata direttamente sul device di controllo generale. E' completamente scritta in C++11 utilizzando le librerie Qt.

Per poter mostrare i dati in tempo-reale, il software è stato realizzato utilizzando una complessa architettura multithread. Ogni Thread ha uno scopo differente, ed i dettagli tecnici saranno spiegati nel Capitolo 3. Come concetto generale, un Thread è usato per l'interfaccia di rete, ed uno per il post-processing dei dati e l'aggiornamento dei grafici.

Questo ha significato una pianificazione preliminare molto esaustiva delle interazioni tra i differenti Threads, perché l'accesso alle variabili deve essere regolato, altrimenti il comportamento del sistema potrebbe essere imprevedibile.

In più, un metodo per segnalare eventi tra Thread diversi è necessario. Un esempio potrebbe essere il Thread della rete, che notifica la presenza di nuovi dati, pronti per essere aggiornati sui grafici.

Nella implementazione, diversi algoritmi sono stati utilizzati, garantendo che:

- Il delay dovuto alla trasmissione dati è relativamente basso ed impercettibile dall'utente.
- Non c'è accumulo di ritardo.

Il software ottenuto consente la visualizzazione a tempo-reale delle pressioni lette dai sensori nella suola, con un tempo medio di ritardo di: $14.1ms \pm 9.1ms$[6].

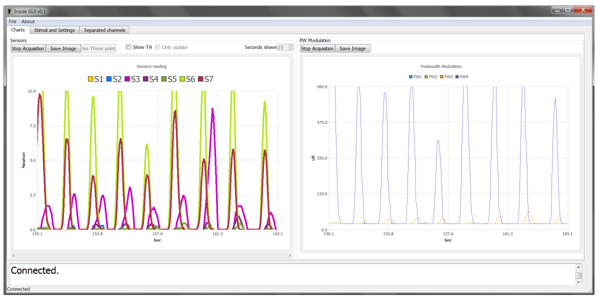
Il software consente anche la possibilità di cambiare i parametri di stimolazione sul momento, come ad esempio: corrente massima, corrente minima, ed elettrodi associati.

Come funzionalità aggiuntiva, ha anche un sistema per aumentare la simmetria del passo in amputati: un feedback audio negativo viene prodotto ogni volta che un algoritmo valuta che la forza generata è troppo bassa. Anche se questo controllo è pronto per essere testato, nessuna valutazione è stata fatta su pazienti o soggetti.

L'immagine in Figura 11 mostra la finestra principale del Client sviluppato.

Una descrizione completa del sistema è presentata al Capitolo 3, ma viene qui riportato un sommario introduttivo delle principali caratteristiche implementate:

[6]Questa valutazione è ottenuta dal Capitolo 6 come la somma del ritardo dovuto alla tramissione Bluetooth, e la trasmissione via rete verso il Client. Solo $1ms$ è dovuto alla trasmissione via rete. Più informazioni possono essere lette al Capitolo 6.



Connected.

Connected

FIGURE 11: Client GUI: Finestra principale.

1. Connessione Remota o Locale al controller della neuroprotesi.
2. Auto-salvataggio dei dati letti e della stimolazione effettuata.
3. Auto-salvataggio delle impostazioni scelte dall'utente.
4. Auto-salvataggio delle impostazioni di stimolazione, con la possibilità di caricarle da un file esterno.
5. Scorrimento dei grafici per vedere i valori precedenti.
6. Possibilità di attivare/disattivare sensori differenti.
7. Algoritmi di controllo per il time-out e ritardo.
8. Feedback per il controllo di bilanciamento del peso a due threshold, unico per ogni sensore.

Software di calibrazione

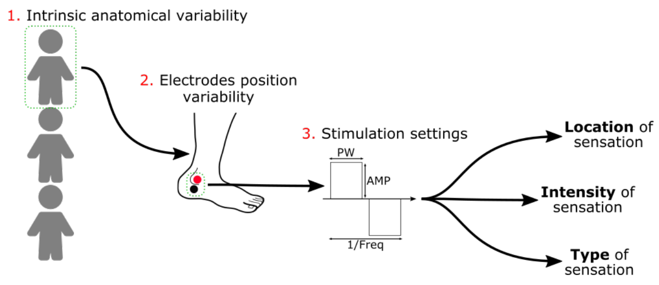


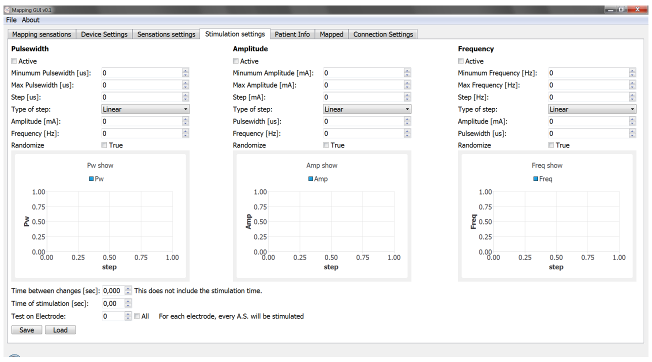
FIGURE 12: Concetto di calibrazione: variabilità associata con la stimolazione.

Per avere una neuroprotesi utile e funzionale, la stimolazione elettrica transcutanea usata sul soggetto dovrebbe evocare sensazioni distali in una posizione voluta, ottenendo una sensazione nota sia in termini di intensità ma anche di tipo.

Data l'inevitabile variabilità anatomica intrinseca tra diversi soggetti, assunzioni sulle sensazioni percepite non possono essere fatte. Invece è necessario uno step di **calibrazione**.

La calibrazione consiste nell'associare:

- Posizioni degli elettrodi.
- Stimolo generato, in termini di pulsewidth, corrente e frequenza.



Pulsewidth

☐ Active	
Minumum Pulsewidth [us]:	0
Max Pulsewidth [us]:	0
Step [us]:	0
Type of step:	Linear
Amplitude [mA]:	0
Frequency [Hz]:	0
Randomize	☐ True

Amplitude

☐ Active	
Minumum Amplitude [mA]:	0
Max Amplitude [mA]:	0
Step [mA]:	0
Type of step:	Linear
Pulsewidth [us]:	0
Frequency [Hz]:	0
Randomize	☐ True

Frequency

☐ Active	
Minumum Frequency [Hz]:	0
Max Frequency [Hz]:	0
Step [Hz]:	0
Type of step:	Linear
Amplitude [mA]:	0
Pulsewidth [us]:	0
Randomize	☐ True

Time between changes [sec]:	0,000	This does not include the stimulation time.
Time of stimulation [sec]:	0,00	
Test on Electrode:	0	☐ All For each electrode, every A.S. will be stimulated

FIGURE 13: Software di Calibrazione (mapping) software: impostazioni degli stimoli.

- Intensità della sensazione percepita sugli elettrodi.
- Tipo della sensazione distale.
- Intensità della sensazione distale.
- Posizione della sensazione distale.

La sensazione percepita dai soggetti sotto gli elettrodi, idealmente dovrebbe essere assente, dato che lo scopo è di implementare un feedback somatotopico: la sensazione dovrebbe essere sentita solo nella parte dell'arto mancante o danneggiato. Se le posizioni degli elettrodi sono sbagliate, questa percezione è molto fastidiosa anche ai livelli più bassi di stimolazione, talvolta presentandosi come dolore. Altrimenti la sensazione percepita è vibrazione, e dipendendo dalle posizioni degli elettrodi, potrebbe essere molto bassa anche all'aumento dell'intensità di stimolazione.

La **sensazione distale**, è invece la percezione evocata voluta. E' **distale** rispetto la posizione degli elettrodi. Ottenerla significa che le fibre nervose sono correttamente stimolate. Per esempio, stimolando il nervo tibiale al livello della caviglia significa che il soggetto dovrebbe percepire alcune sensazioni nel piede. Per il lavoro di questa tesi, il tipo di sensazione usata dai soggetti per descrivere la percezione è sempre stato un insieme tra formicolio e vibrazione. Perciò, nei dati preliminari presentati, l'intensità della sensazione distale si riferirà sempre a questo tipo di percezione.

La procedura di calibrazione dovrebbe essere ripetuta, per ogni posizione degli elettrodi (per ogni soggetto) con una variazione dell'intensità di stimolazione. Per esempio, se il feedback è codificato in termini di pulsewidth, per avere il risultato di una singola calibrazione è necessario modulare questa intensità tra un minimo ed un massimo.

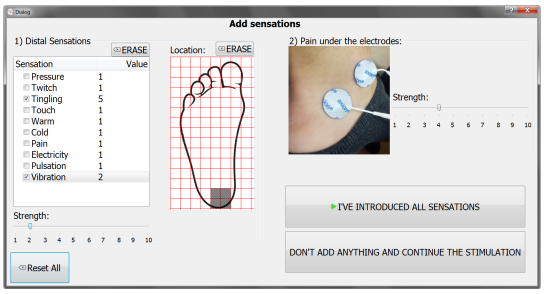
La Figura 12 mostra il concetto di calibrazione (mapping) necessario data l'alta variabilità associata al sistema.

Un software che consente funzionalità di calibrazione è stato sviluppato da zero, usando C++ e librerie Qt. Una validazione delle sue funzionalità è stata fatta su volontari sani, realizzando una stimolazione non-invasiva sulla caviglia, stimolando il nervo tibiale con elettrodi superficiali.

Immagini 13, 14 mostrano le figure principali del software.

Di seguito un veloce riassunto delle funzioni implementate:

- Database dei soggetti con impostazioni di stimolazione personalizzate.



Sensation	Value
☐ Pressure	1
☐ Twitch	1
☑ Tingling	5
☐ Touch	1
☐ Warm	1
☐ Cold	1
☐ Pain	1
☐ Electricity	1
☐ Pulsation	1
☑ Vibration	2

FIGURE 14: Software di Calibrazione (mapping): vista utente.

- Grafici in tempo-reale che mostrano la stimolazione eseguita.
- Possibilità di stimolare utilizzando il device RehaStim.
- Calibrazioni possibili: usando un set ordinato di stimoli, randomizzando l'ordine degli stimoli, test di discriminazione.
- Possibilità di fare Calibrazione utilizzando modulazione di: pulsewidth, corrente e frequenza.
- La finestra utilizzata per descrivere le sensazioni permette di scegliere: varie percezioni distali, ognuna con un livello da 1 a 10. La posizione della sensazione distale e l'intensità della sensazione percepita sotto gli elettrodi.

Protocollo per la validazione del software di calibrazione

Una validazione del sistema è stata fatta su quattro volontari sani, stimolando utilizzando elettrodi superficiali posti sulla caviglia, in maniera da indurre attivazione del nervo tibiale, con conseguente sensazione distale sotto il piede.

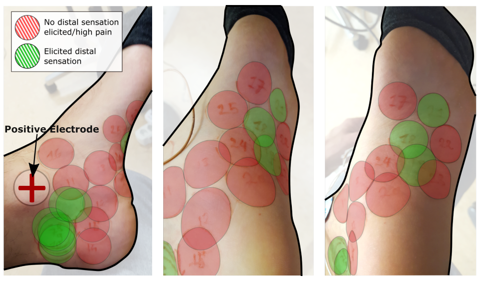

No distal sensation elicited/high pain
Elicited distal sensation
Positive Electrode

FIGURE 15: Prima valutazione delle posizioni degli elettrodi.

Una prima valutazione di numerose posizioni degli elettrodi è stata fatta, ottenendo così un range di posizioni possibili per gli elettrodi. Le posizioni utili sono quelle in grado di generare sensazioni distali, ma non che non causano troppo fastidio direttamente sotto gli elettrodi. La zona degli elettrodi utile è visibile nell'immagine 15. In questo caso, la valutazione è stata fatta mantenendo l'elettrodo positivo sul malleolo mediale, testando invece le varie posizioni per l'elettrodo negativo.

Una volta che una prima valutazione iniziale è stata ottenuta, la calibrazione è stata fatta su volontari sani. I risultati della calibrazione sono sempre stati associati alla posizione della caviglia, dato che il movimento relativo tra i nervi e gli elettrodi superficiali influisce sui risultati.

Un riassunto del **protocollo** utilizzato per la validazione è il seguente:

1. Le zone di pelle interessate (cioè, la caviglia) sono state lavate, ed in caso i peli presenti sono stati rimossi.

2. L'elettrodo positivo è posizionato sul malleolo mediale, mentre il negativo è posizionato su una delle posizioni proposte. Per ottenere risultati comparabili, quattro posizioni principali sono state usate come punto di riferimento. Queste posizioni sono visibili sulla parte sinistra dell'immagine 16.
3. Dopo essersi assicurati che il Soggetto fosse a suo agio con la stimolazione elettrica, il primo test inizia.
4. Una posizione della caviglia è scelta, tra: sollevata, in piedi, spinta con le dita, spinta col tallone.
5. Un aumento lineare nella stimolazione tramite pulsewidth è usato, partendo da 20us con step di 10us, ogni 3 secondi. La frequenza usata sarà sempre 80Hz, mentre la corrente è 2mA.
6. Se il soggetto ha eccessivo fastidio prima di raggiungere i 100us, la posizione è cambiata ed il procedimento è ripetuto da 20us.
7. Al soggetto viene chiesto di notificare la prima volta che percepisce una sensazione distale, e quando la stimolazione è troppo intensa. Questi valori saranno usati nel range come minimo e massimo.
8. Il soggetto è istruito sulla pagina di "aggiunta sensazione", gli vengono spiegate le diverse sensazioni selezionabili, e che dovrà compilare l'intensità di stimolazione sugli elettrodi e distale.
9. La fase di calibrazione inizia: la pulsewidth è cambiata tra il minimo ed il massimo trovati, in un ordine casuale. Lo step minimo tra le stimolazioni è 20us.
10. Ogni volta che una stimolazione viene generata, se il soggetto percepisce sensazioni, gli viene chiesto di compilare la finestra di "aggiunta sensazione".
11. Una volta che ogni stimolo nel range è stato generato, la procedura finisce e le scelte sono salvate ed associate agli stimoli.

Risultati

Validazione del software di calibrazione

Quattro volontari sani hanno partecipato in questi test preliminari. La Figura 16 mostra i risultati della calibrazione totale per il Soggetto 1. La Figura 17 confronta le sensazioni distali evocate per i diversi Soggetti, con posizioni di elettrodi e caviglia simili.

Un esempio della ripetibilità per lo stesso soggetto è mostrata in Figura 18.

Nell'immagine 19a, l'intensità della sensazione distale percepita dai soggetti viene messa in relazione alla pulsewidth usata per la stimolazione. Il fastidio presente direttamente sotto gli elettrodi è invece mostrato in Figura 19b.

Questi risultati sottolineano come ci sia una buona ripetibilità sullo stesso soggetto, mentre la comparazione dei risultati tra soggetti diversi mostra una forte variabilità, dovuta alle differenze anatomiche intrinseche. Questo risultato era previsto, e conferma che il software di calibrazione è assolutamente necessario.

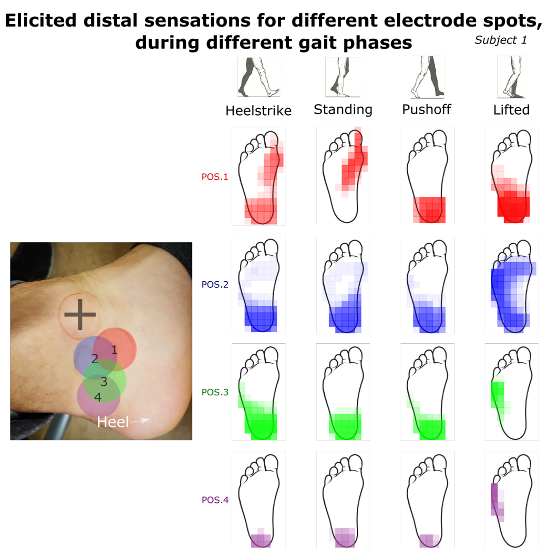


FIGURE 16: Risultati di calibrazione per il Soggetto 1.

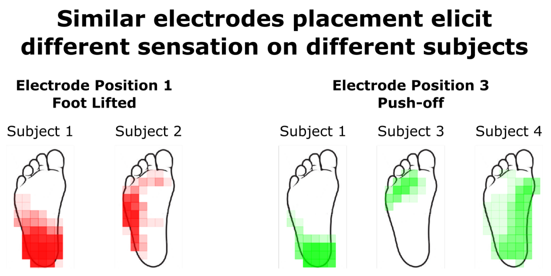


FIGURE 17: Confronto tra soggetti diversi.

I risultati mostrano anche che i soggetti sono capaci di associare correttamente un aumento della stimolazione, con un aumento di intensità distale percepita. L'aumento di stimolazione comporta perciò un aumento di percezione distale, mentre il fastidio sotto gli elettrodi satura ad un valore accettabile.

I risultati completi sono mostrati e spiegati in dettaglio nel Capitolo 7, con anche più enfasi nelle descrizioni anatomiche. Test sono stati fatti anche per determinare se invertire la polarità degli elettrodi influisce sulle sensazioni.

Il coefficiente di Weber è stato estrapolato dai test di discriminazione, ottenendo 0.10 ± 0.01. Più dettagli sono disponibili al Capitolo7.

Sistema ad anello chiuso

La neuroprotesi dovrebbe comportarsi come un sistema ad anello chiuso: la forza è letta nella suola ed inviata tramite Bluetooth ad un controller globale, che l'analizza e genera una stimolazione elettrica appropriata. Più è alta la forza, maggiore sarà la stimolazione elettrica, che nel nostro caso è codificata come la modulazione di pulsewidth. La funzione di trasferimento è una associazione lineare tra forza e pulsewidth generata. Come hanno mostrato i risultati, un aumento di pulsewidth (intensità della stimolazione) è associato ad un aumento nella sensazione distale percepita, perciò il concetto alla base del sistema è corretto. In questa tesi è stato sviluppato un sistema ad anello chiuso, pronto per essere testato con pazienti.

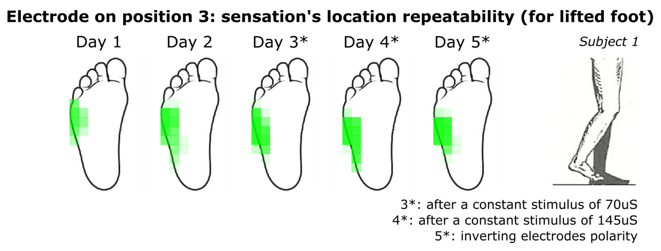
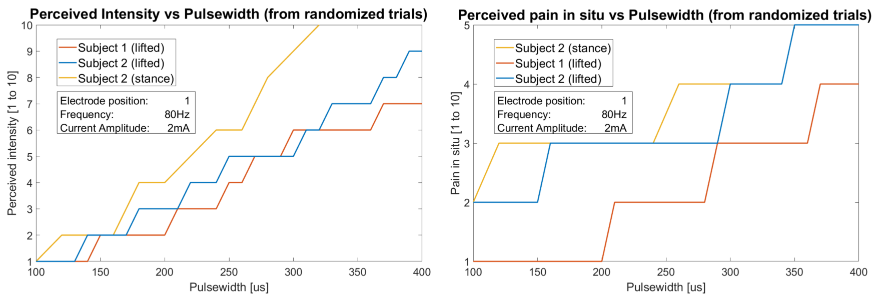

Electrode on position 3: sensation's location repeatability (for lifted foot)
Day 1
Day 2
Day 3*
Day 4*
Day 5*
Subject 1
3*: after a constant stimulus of 70uS
4*: after a constant stimulus of 145uS
5*: inverting electrodes polarity

FIGURE 18: Ripetibilità per il Soggetto 1, posizione dell'elettrodo 3.



(A) Intensità distale percepita dal Soggetto 1 e Soggetto 2 all'aumento della stimolazione.

(B) Dolore/Fastidio percepito dal Soggetto 1 e Soggetto 2 all'aumento della stimolazione.

FIGURE 19: Intensità percepite all'aumento dello stimolo.

La forza è letta utilizzando una suola sensorizzata, ed i dati sono inviati ad un controller generale, che a sua volta cambia la stimolazione prodotta dallo stimolatore elettrico per indurre sensazioni sul Soggetto. Come la Figura 20 mostra, il sistema funziona come un anello chiuso: la stimolazione influenzerà le azioni del Soggetto, e di conseguenza la forza generata, variando perciò il feedback sensoriale artificiale indotto.

Un software multithread, funzionante sul controller globale, è stato sviluppato come parte del lavoro della tesi. Il risultato ottenuto realizza una modulazione degli stimoli generati in tempo-reale, variandoli a seconda della forza. Nella Appendice A sono presenti due link a video che mostrano il funzionamento del sistema.

L'implementazione multithread ottiene più sicurezza nei confronti di possibili errori del codice, ed una velocità di esecuzione maggiore. Il sistema ad anello chiuso ha

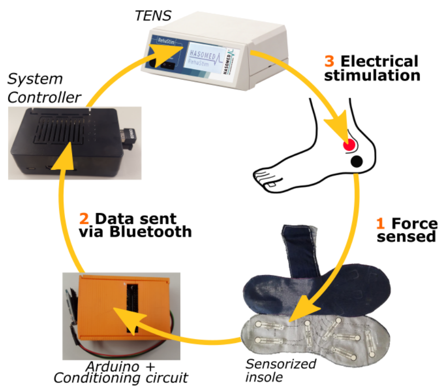


FIGURE 20: Concetto di neuroprotesi, caso di esempio per un paziente diabetico.

una media di ritardo, come caso peggiore, di 25.4ms, che risulta impercettibile al soggetto.

Conclusioni

L'interfaccia grafica per la visualizzazione dati, il Client GUI sviluppato, ha comportato la possibilità di valutare la forza in tempo-reale, e di cambiare le impostazioni della stimolazione elettrica sul momento. Il Client si può connettere da remoto al sistema di controllo, consentendo la possibilità di visualizzare i dati da un comune device: portatile, desktop fisso, od anche smartphones. Un'alta performance è ottenuta con una implementazione a multithread e nessun ritardo è accumulato. In più, un feedback audio per aumentare la simmetria del passo è stato implementato.

La limitazione principale del sistema è che la visualizzazione grafica dei plot ed il conseguente aggiornamento in tempo reale, è un processo grafico molto pesante, ed è una limitazione intrinseca alle librerie grafiche utilizzate. Detto questo, non è stata trovata nessuna alternativa che potesse produrre risultati migliori, e che fosse comparabile con le librerie usate. In più, il controllo di bilanciamento del passo, deve essere testato con pazienti, per verificare se l'implementazione migliora effettivamente l'indice di simmetria.

Un miglioramento del Client dovrebbe essere il creare un sistema di controllo della simmetria del cammino più avanzato, utilizzando un approccio di Machine Learning, oppure valutando le derivate prime delle pressioni lette. Una seconda suola sensorizzata potrebbe essere messa nell'altra scarpa, potendo comparare così più facilmente la simmetria del passo.

Un **software di calibrazione** consente di impostare correttamente la neuroprotesi, e questo è un passo obbligatorio visto le numerose variabilità associate al sistema. Alcuni test preliminari sono stati svolti, validando la funzionalità del software e la sua applicabilità, ed allo stesso tempo ottenendo informazioni riguardo un metodo non ancora testato dal mondo scientifico.

Il software potrebbe essere migliorato introducendo la possibilità di stimolare con combinazioni più complesse, per esempio un aumento esponenziale delle intensità degli stimoli. Anche un sistema automatizzato per la decisione delle migliori posizioni degli elettrodi potrebbe essere aggiunto.

Un completo sistema ad anello chiuso è stato implementato. Il sistema genera una stimolazione transcutanea dei nervi utilizzando lo stimolatore RehaStim, modulando le intensità delle stimolazioni a seconda della forza letta dai sensori. Più output dello stimolatore possono essere usate contemporaneamente, associandole a differenti sensori della soletta. Il ritardo globale del sistema ottenuto è minore di 30ms, cioè meno di un terzo di quello suggerito dalla letteratura [15].

Da un punto di vista tecnologico, il sistema ad anello chiuso è pronto per essere testato come una neuroprotesi a feedback sensoriale, non-invasiva. Detto questo, non si può ignorare la fondamentale limitazione del sistema: non è ancora stato valutato su effettivi amputati o diabetici.

Una ricerca approfondita dovrebbe essere svolta sui pazienti, in modo da identificare, come prima cosa se la sensazione distale può essere evocata, ed in quel caso, quali posizioni degli elettrodi sono utili. Uno stimolatore elettrico più miniaturizzato dovrebbe essere usato, consentendo così lo studio del sistema ad anello-chiuso con effettiva stimolazione mentre il paziente cammina. Arrivando così allo scopo finale: una valutazione su quanto impatto questo sistema avrebbe su pazienti. Sia in termini di cammino simmetrico, ma anche di velocità del cammino, confidenza nella protesi, variazioni sul dolore dell'arto fantasma, e personificazione della protesi.

In futuro test con elettrodi di dimensioni inferiori potrebbero essere usati, ottenendo così una maggiore accuratezza nelle sensazioni evocate.

In conclusione, il lavoro di questa tesi ha comportato l'implementazione di molte parti di una neuroprotesi a feedback-sensoriale non-invasivo, fino ad ottenere un sistema ad anello chiuso. Una validazione tecnologica è stata svolta per ogni parte sviluppata. In più, test preliminari svolti su volontari hanno posto le basi per caratterizzare la stimolazione del nervo tibiale, con elettrodi posti sulla caviglia. Nessuno studio in letteratura ha ancora esplorato questo campo, ma i risultati, anche se preliminari, suggeriscono che l'implementazione di questo sistema sia possibile, ed in futuro verrà svolta ricerca su pazienti.

Struttura della tesi

La tesi è organizzata come segue:

1. **Capitolo 1** – *Contesto e scopo del progetto*: viene descritto il concetto di feedback sesnoriale e la sua importanza, descrivendo una possibile implementazione di base del sistema.
2. **Capitolo 2** – *Punto di inizio e parti mancanti*: dato che questo lavoro parte come branchia di progetto passato, le parti che erano già disponibili sono presentate, insieme alle loro limitazioni.
3. **Capitolo 3** – *Interfaccia grafica a tempo-reale per applicazioni basate su suole sensorizzate, con controllo del bilanciamento*: in questo capitolo viene descritto lo sviluppo di un software in grado di controllare da remoto la neuroprotesi, e di visualizzarne i dati di forza e stimolazione in tempo-reale.
4. **Capitolo 4** – *Integrazione del TENS*: lo stimolatore elettrico usato è RehaStim, uno strumento non-invasivo. In questo capitolo viene introdotto come uno dei blocchi principali della neuroprotesi.
5. **Capitolo 5** – *Software di calibrazione*: viene descritto lo sviluppo di un software per la calibrazione di una neuroprotesi.
6. **Capitolo 6** – *Sistema globale*: in questo capitolo viene presentato il sistema ad anello chiuso completo, con una valutazione dei ritardi presenti.
7. **Capitolo 7** – *Validazione preliminare del software di calibrazione*: il software di calibrazione è stato testato su quattro volontari sani, ottenendo così una validazione un punto di vista tecnologico del sistema, ed allo stesso tempo informazioni sulle sensazioni somatotopiche evocate negli arti inferiori.
8. **Capitolo 8** – *Discussione e sviluppi futuri*: le limitazioni dei blocchi introdotti vengono discusse, insieme a proposte per possibili miglioramenti del sistema.

Chapter 1

Context and aim of the project

1.1 Amputees

Traumatic events, or pathologies, can lead to the loss of limbs. For industrialized countries the most common cause is diabetes, while in non-industrialized it's mostly due to trauma [27].

Amputations make a huge impact on the patient life, reducing substantially its quality. A method to evaluate the patient quality of life (with a general and broad validity) has been proposed by the **WHO** (*World Health Organization*) and it's the *International Classification of Functioning, Disability and Health*, also known as **ICF**.

Some examples of problems for amputees, following the ICF conceptual model, includes and are not limited to [18]:

- Decreased movement amplitude
- Pain in the stump extremity
- Difficulties/Impossibility in performing common home tasks
- Difficulties in using transportation
- Difficulties in obtaining and keeping a job
- Social isolation
- Difficulties/Impossibility to do sports
- Phantom Limb Pain

Phantom limb pain (**PLP**) is a pain sensation located in the missing limb. Even if the limb is not present, a strong pain is felt; this happens for about 80% of patients and even after years it may still be present [43].

In particular, for lower-limb amputees, *even in case of using high-tech commercially available prosthesis*, some of the found problems are [45] [11]:

- Increased metabolic cost
- Gait asymmetry
- Poor balance
- Osteoarthritis
- Back pain

- Increased cognitive burden

Indeed using the usual passive prosthesis (Body-powered or not), or even the more recent active neuroprosthesis, these problems are still present due to the lack of a somatotopic sensory feedback. "*According to amputees, sensory feedback is amongst the most important features lacking from commercial prostheses*" [12].

Despite the high technological solutions adopted in the most recent devices, where the subject was allowed to control the movement of the artificial limb by will, patients still find the prosthesis not to be their own (*low embodiment*), they can't use it properly in terms of expressed force, and many times they decide not to use these neuroprosthesis at all [12].

Regarding *only* the improvement of gait patterns for amputees (which is also necessary to avoid osteoporosis in the residual stump and osteoarthritis in the intact limb [17]), some methods not involving a somatotopic feedback have been implemented [11]: either an alert was started (visual, auditory or vibratory) whenever a gait variable exceeded a range; or a variable (such as force) was encoded in a non-somatotopic, non-homologous feedback.

A somatotopic feedback would be easier to integrate in the walking patters, leading to an improvement in gait as well; this without excluding the various other positive effects of implementing such device: embodiment of the prosthesis, more trustfulness, force regulation.

These concepts have lead to some recent developments in the field: **real-time bidirectional sensory neuroprosthesis** [37], Figure 1.1. The idea implemented is the use of electrical stimulation concurrently with an active neuroprosthesis, to induce a somatotopic and homogeneous sensory feedback, while also moving the artificial limb. This restores the closed control-loop, which is a fundamental key point in our ability to "act towards a mean" correctly.

For upper limb amputees, both invasive and non-invasive methods of stimulation have been tested with successful results. The non-invasive method, which uses superficial electrodes, is somatotopic but provides paresthesia as sensation. As upside it doesn't present the many limitations that occurs in implanted electrodes (i.e., the invasive electrical stimulation), such as a surgical operation.

1.2 Diabetics

With some degrees of similarity these problems happen also to diabetic patients, whom may develop neuropathies: up to 50% of patients is affected by a neuropathy [5], the most common one being peripheral neuropathy.

Even though neuropathies developed in these patients are heterogenous, at least two major groups can be used to classify the pathology [42]: typical DPN (the most common one) and atypical DPN. Typical DPN seems to develop with longstanding hyperglycemia, and alterations of microvessels appear to be the cause of pathologic alteration of nerves; it affects mostly the toes, feet, or legs.

DPN may have the following symptoms [3]:

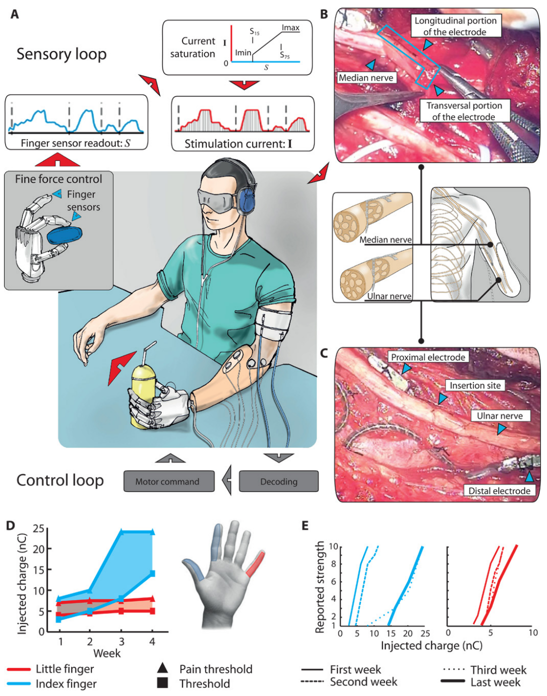


FIGURE 1.1: Bidirectional control of hand prosthesis and characterization of neural stimulation. **(A)** Shows how the sensory-feedback is driven by pressure sensors. **(B & C)** Displays the implantation of electrodes in the nerves. **(D & E)** Time courses of the injected charges and intensity felt during the weeks of testing [37].

- **Negative neurophatic sensory symptoms (N-NSS)** – decreased or absent function of sensory receptors: inability to feel tactile stimuli, mechanical displacement or movement of hair, skin, or other anatomical parts; decreased feeling of cooling, warming or noxious stimuli.
- **Positive neurophatic sensory symptoms (P-NSS)** – sensory experiences arising spontaneously, such as tingling, burning, or aching pain.
- **Both NNSS and PNSS**.

Thus these kind of patients may lose foot sensitivity, removing the sensory feedback used to walk correctly: indeed patients with DPN have been reported to have abnormalities in gait and an increased risk in falling [1]. Moreover, excessive pressure on some part of the foot are due to the missing sensory-feedback and may lead to ulcer [44], which is the most common cause of amputation in diabetics [2]. *"Once ulceration is present, the gait may alter further, leading to a self-perpetuating situation of ulcer formation and lack of healing"* [1].

Just like amputees, DPN patients suffer from the absence of sensory feedback in the foot, leading to some common problems; furthermore these problems may themselves lead to amputation. This absence may be restored through accurate superficial electrical stimulation over the nerves. This is a novelty, so far electrical stimulation has only been used as a therapeutic method [31]. An haptic feedback, driven by a sensorized insole, has been tested using pneumatically controlled baloon actuators [14].

The concept is evidently quite similar to the one for amputees: the "missing" touch in the foot should be somehow restored by electrical stimulation. Of course this electrical stimulation should occur only when the foot needs to sense something, in the appropriate position, obtaining a somatotopic feedback.

This thesis will focus mainly on lower limb amputees and DPN patients, providing the basis to elicit somatotopic feedback with non-invasive stimulation.

1.3 Conclusion of the introductory context

For upper-limb amputees, studies about the perception of evoked sensations (implanted/superficial electrodes) have already been made [16], [20], [6], [38]; including adaptation after prolonged stimulation [19]. The consensus is that both methods can evoke sensations, and that these elicited perceptions are well integrated in a movement control-loop.

In this thesis a non-invasive stimulation method will be proposed as implementation of a somatotopic sensory feedback, for both lower limb amputees and DPN patients. So far no study has yet explored this field, not even the characterization of evoked sensations for lower limbs using superficial stimulation.

This thesis work has been done at the Neuroengineering Lab in ETH Zurich[1], under the supervision of Professor Stanisa Raspopovic and PhD Francesco Petrini and in collaboration with the Sensory-Motor System Lab.

1.4 Concept: a closed-loop neuroprosthesis for lower limb amputees and diabetics

In order to break through the problems of currently available prosthesis, either these be passive or active, an implementation that restores the sensory feedback must be developed. This should be somatotopic, leading to an highly intuitive integration of the sensation in the control-loop, with consequentially increased gait symmetry, walking speed, less cognitive burdensome, higher embodiment of the prosthesis, reduction of PLP, more confidence towards the prosthesis and generally speaking the stump. This has all been already tested and proven to be true with invasive electrical stimulation, using implanted electrodes in the sciatic nerve.

The problem of losing sensations is also common in diabetic people. Just as amputees, they walk slowly, incorrectly, and for this they tend to develop ulcers, possibly ending up with amputations.

The solution to these problems is to find a way to artificially elicit sensations during gait. The sensation should be felt in the foot (somatotopic feedback) and ideally it should be touch (homogeneous feedback). The first consideration could be assumed to be actually more important from a functionality point of view, while the elicited feeling can be vibration/tingling or generically paresthesia.

The intensity of the feeling should be driven by a sensorized insole that sense force, with its signal properly transduced to an electrical stimulus.

This restores the loop-control in the brain: a movement is produced, a feedback is sensed, is integrated in the mental control scheme and used as a control variable.

[1]Swiss Federal Institute of Technology in Zurich.

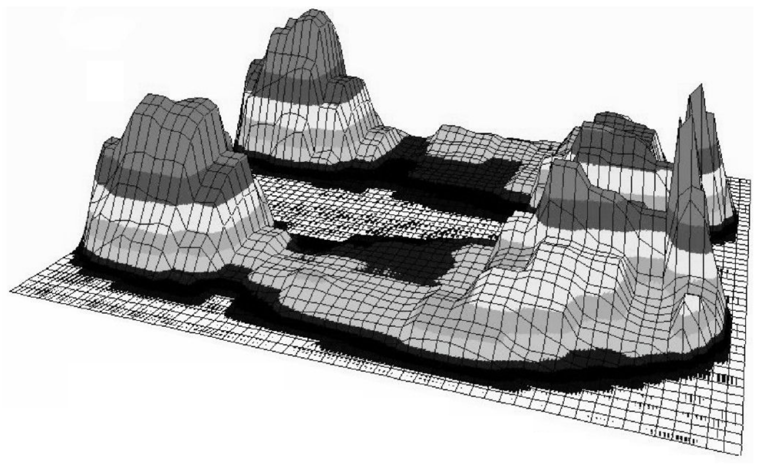


FIGURE 1.2: Pressure map of the foots [26].

1.5 Implementation of the closed-loop system

At least three main blocks are necessary:

1. Sensorized Insole
2. Processing unit
3. Stimulator

The insole should have force sensors in known and useful positions: these can be found evaluating foot pressure map during gait, an example of such can be found on Figure 1.2; together with an electronic circuit to prepare the signal to be sampled (i.e.: polarization circuit). A commercially available pressure mapping system is the **Tekscan's Digital Footprint Technology** [41], while a densely sensorized insole (commercially available) is the **PPS's Foot Pressure Mapping System** [34]. Alternatively literature already provides useful information [26].

The sampled signal would be then analyzed by a **Processing Unit**: an algorithm should decide how and where to electrically stimulate the patient. This is an important consideration: a mapping step is clearly needed, associating the available electrodes positions and the evoked sensations.

This will be further discussed in few chapters, but a simple example would be: assuming to have a known electrode position that elicits a sensation on the heel, and a force sampled on a close position, a linear mapping between force and stimulation could be made. If the stimulation is modulated by pulsewidth (see the Chapter 7 for considerations about the best stimuli modulation), the transfer function would then be:

$$PW_x[\mu s] = PW_{min} + \frac{Fr_x - Th_{Fr}}{Fr_{max} - Th_{Fr}}(PW_{max} - PW_{min}) \tag{1.1}$$

This formula will be discussed in Chapter 6.

The last block is the **electrical stimulator**: its software should allow remote control of the provided stimuli, allowing at least the settings of:

1. Pulsewidth
2. Amplitude

3. Start of the stimulation
4. Stop of the stimulation

It's important to notice that the delay of the various blocks should be known, or at least estimated, and should be the lowest possible as to have a concurrency with the events, and avoid lowering the system performance [12]; ideally it should be kept lower than 100ms [15]. For an evaluation of the delay in the developed system please refer to Chapter 6.

1.5.1 TENS to evoke somatotopic sensations

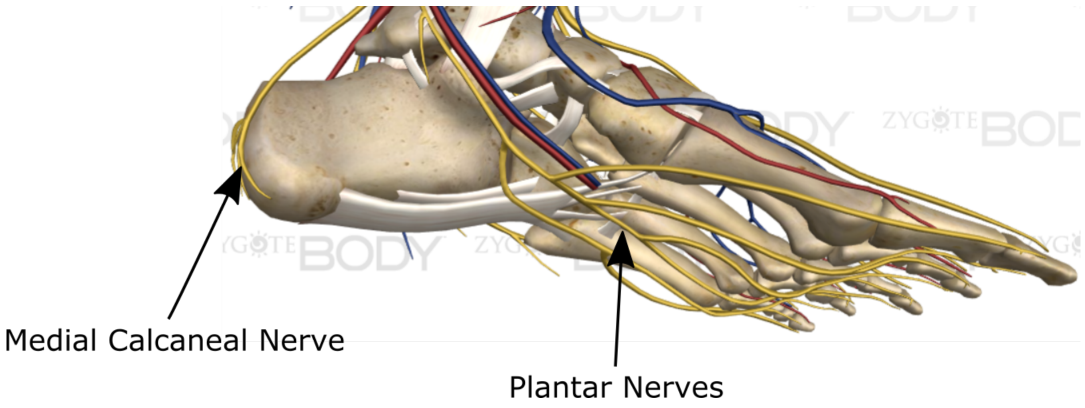


FIGURE 1.3: 3D Representation of Tibial Branches in the foot.

The stimulation used in this thesis is non-invasive, meaning that superficial electrodes are used. **TENS** stands for Transcutaneous Electrical Nerve Stimulation which summarizes the concept: the nerves are stimulated using electrodes placed over the interested nerves.

For the preliminary results obtained in this thesis, the stimulation was made occurring at the ankle level, with the idea to induce spiking in the Tibial Nerve, and its subdivisions (Plantar nerves and Calcaneal nerve). These nerves are associated with sensations under the foot, thus matching the aim of this project.

The method has already been used previously to induce somatotopic sensations in upper-limb amputees [12], and some evaluations about the science mechanisms behind this type of stimulation have been made [39]. Phantom finger sensation hand has been induced [30] in upper-limb amputees, and evoked tactile sensation in forearm amputees have been characterized [7]. No study so far has tried to characterize somatotopic sensations induced in lower-limbs, or to evaluate the possibility to actually obtain distal sensation using superficial electrodes. The assumption made (and confirmed by this work) was that the ability to induce spiking in the nerves should also be present placing the electrodes close to the tibial nerve, over the ankle.

1.6 Fast view of the implemented blocks

The implemented parts will be shown and analyzed in the various chapter, while an evaluation on the whole system will be presented on Chapter 6.

As preliminary introduction, the following blocks were already existing and were used in this thesis (see the next chapter for further details):

- **FSR**[2] **sensorized insole**, calibrated and made by coworker Oriella Gnarra, with 7 sensors placed at keypoints
- **Capacitive**[3] **sensorized insole**, calibrated and made by coworker Oriella Gnarra, with 7 sensors placed at keypoints
- Existing **electronic board** for FSRs conditioning and data sampling, communicating via bluetooth to a rapsberry microcomputer
- Existing **C++ code** for displaying read values on a remote console (running on a Raspberry)

While the new developed parts are:

- **Real-time GUI**[4] to display remotely the data read, and control the software running on the raspberry (e.g.: changing the stimulation parameters). See Chapter 3 for a description and discussion of the design choices
- **Mapping Software** to associate the used electrode positions, the evoked sensations and the pain. See Chapter 5 for a description and discussion of the design choices; see Chapter 7 for various tests on healthy volunteers, results and conclusion.
- The code on the **system-controller**[5] to implement a closed-loop sensory neuroprosthesis.

[2] force sensitive resistor

[3] sensitive to force

[4] Graphical User Interface

[5] The code running on the Raspbbery Pi 3, then replaced with an Odroid. This replacement will be introduced for simplicity in Chapter 6.

Chapter 2

Starting point and missing parts

2.1 Initial development

From the concept of the importance of sensory feedback in amputees, a project with the aim to implement a sensory-feedback neuroprosthesis for lower-limb amputation was started, and a prototype was developed. The sensory feedback was provided through implanted electrodes in the residual tibial nerve, the intensity of the charge injected was driven by force sensors positioned on the prosthesis with a sensorized insole. The evaluation of many different factors on the patient provided a clear evidence of its usefulness, namely on the ability to walk faster, higher confidence on the prosthesis, less energy consumption, less brain burdensome, lowering or almost complete suppression of PLP, higher embodiment of the artificial limb.

The existing project prototype, from which this thesis has been built upon, essentially provided:

- A sensorized insole with 7 FSR sensors placed at keypoint
- A biasing electronic circuit for the sensors
- An Arduino for the acquisition of data
- A Bluetooth module to connect the Arduino to a Raspberry Pi 3 device, allowing the transmission of read data
- A Software on the Raspberry, which continuously displays the read values on a console, and implements a stimulation on the patient
- **STIMEP** neural stimulator and *transversal multichannel intraneural implantable electrodes* (**TIME**)

An example of one of the insoles used and its design is shown in Figure 2.1 [4].

The insole is connected with a cable to a box containing the Arduino and the conditioning circuit, as the following images shows: Figure 2.2, Figure 2.3 [4].

The presence of the battery allows to run the system without any power-line cable.

A schematic for the electronics is presented in Figure 2.4 [4]: the configuration is a simple non-inverting amplifier, leading for each sensor a voltage output equal to:

$$Vo = \frac{Vref}{2}(1 + \frac{R_g}{R_{FSR}}) \tag{2.1}$$

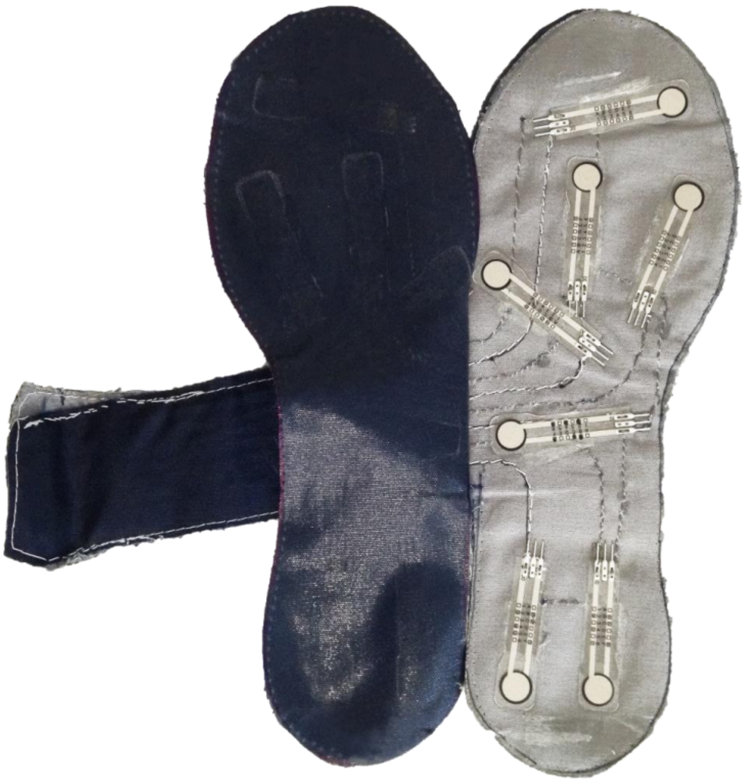


FIGURE 2.1: Opened sensorized insole

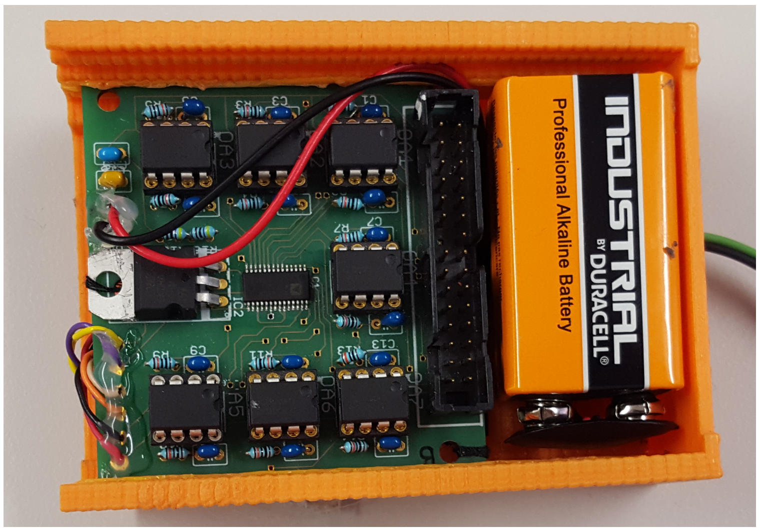


FIGURE 2.2: Open 3D-printed box

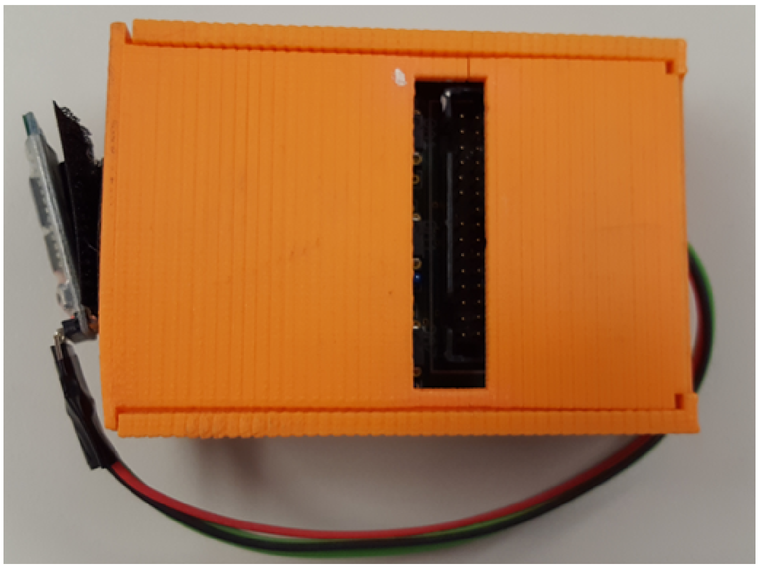


FIGURE 2.3: Closed 3D-printed box

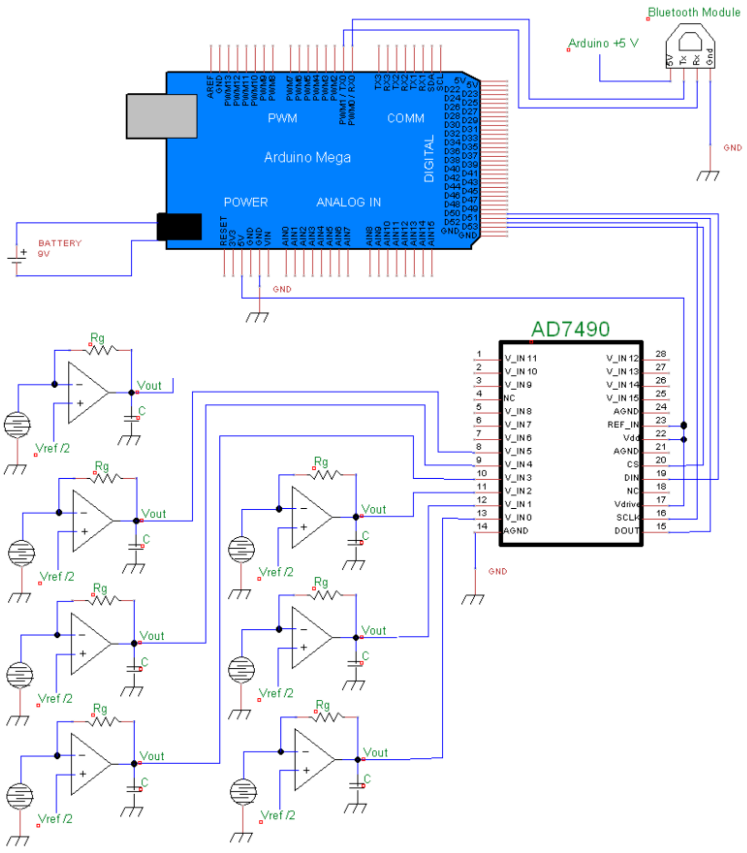


FIGURE 2.4: Initial Electronic Prototype

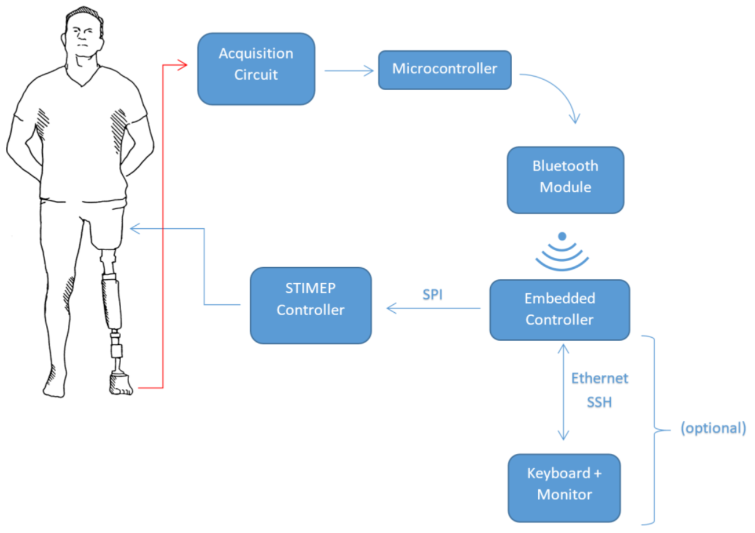


FIGURE 2.5: Existing Initial System

The output voltage of each sensor is then sampled with an external ADC, the **AD7490**. This provides the possibility of having more analogue inputs (thus, possibly, more FSR) and has a sampling speed much faster than the Arduino's ADC. Then the AD7490 sends the sampled value using **SPI**[1] to the Arduino.

Arduino uses the bluetooth module **HC05** to communicate to a Raspberry Pi 3 device and sends the sampled data. The communication is made through *RFCOMM* protocol, which emulates serial port over the *L2CAP*[2] protocol. This highly facilitate the development of the software, and makes the raspberry and Arduino code work also for a direct USB cable communication (that emulates the *RS232* typically with a FTDI chip [28]).

Once the data is sent to the Raspberry, an ad-hoc software displays the data read on a console and if STIMEP is connected, electrically stimulate the patient in the decided spot.

The Figure 2.5 shows the whole system and its links.

2.2 Problems and changes

As explained in the previous parts, this thesis has the aim to provide the basis to elicit somatotopic feedback with non-invasive stimulation, for either lower-limb amputees and diabetics. Firstly this means that the part for the stimulation has to be completely changed to work with a TENS[3] device. The one actually used is the **RehaStim**, deeply discussed in the Chapter 4.

Secondly, the stimulation paradigma has to be redone, to adapt to the new kind of stimulation, finding the best settings that allow for distal evoked sensations and lower in-situ[4] pain. Most importantly, the surface-electrodes' position most suitable for this tasks have to be found.

[1]Serial Peripheral Interface: a synchronous serial communication interface
[2]Logical Link Control and Adaptation Layer Protocol
[3]Transcutaneous electrical nerve stimulation
[4]Over the superficial electrodes

The Rapsberry has been replaced with an Odroid to increase the performace.

Also, the following subsections will elucidate over the existing problems needed to overcome.

2.2.1 Absence of a GUI

A first huge limitation of this system is the absence of a **GUI** (*Graphical User Interface*), which would control the whole application running on the raspberry: displaying data in real time, showing the current stimuli configurations and allow changing them, just to briefly name the most obvious additions (the functions provided by the developed software are listed on the corresponding chapter). Furthermore, being it a prototype it had several issues with the capability to run continuously without freezing. This meant an important re-visitation of the original code.

Not less important, the absence of such GUI rises limitations in both a research facility and a medical-care facility. Indeed, an instant evaluation of the execution, for both the stimulation and the inputs received, can clearly provide a fast assessment to check if everything is working correctly. While it obviously provides an useful tool during the design of a prototype, it's an unmissable part during the actual stimulation of a patient, either that be in a medical care facility or in a research setup. Furthermore, in the former case, a non-engineer staff is the normality, meaning that a highly intuitive and easy-to-use software must be provided.

For these reasons, a first task was to produce a GUI for the system, starting from the working code running on a Raspberry pi 3. From now on "server" will refer to the code running on the Raspberry/Odroid, while Client, or Client-GUI, or GUI, will refer to the Real-Time software developed, usually running on a laptop.

2.2.2 Mapping Software

Another important missing part was the Mapping GUI: a software designed to associate different stimuli, locations of the surface electrodes, and evoked sensations.

The stimulus produced may vary[5] in terms of:

- Pulsewidth
- Current Amplitude
- Frequency of the burst

The placement of the surface electrodes has no boundaries or rules, and neither has the polarity of stimulation.

All these factors affect the evoked sensation, which is the ultimate goal, that has to be recorded, so to be able to make decision over the position of the stimulation and the stimulus settings. Eventually allowing the semi-closed loop implementation.

Thus the Mapping step is crucial and mandatory, and cannot be avoided to obtain a usable sensory-feedback neuroprosthesis. Moreover, it's also needed to provide a first characterization of the evoked stimuli and their properties, since no scientific literature has yet proposed this kind of work for lower-limb (may the subject be

[5]for the RehaStim device

amputee, diabetic or sane). The found results will be explained and displayed on Chapter 7; even if the subject pool is not large, and the experiments are a few, they still provide a first insight in the topic.

Considering the double application of this thesis, a Mapping Software specifically designed for amputees and one for diabetics has been developed, where the difference is only in the window to report the sensation: the application for diabetics is more user friendly and made for being used also by non-technician (the patient himself); the version for amputees provides also the possibility to indicate a proprioception sensation and a specific muscle contraction, while also allowing more freedom in the expression of the felt sensation (see the Chapter 5 for more details).

Chapter 3

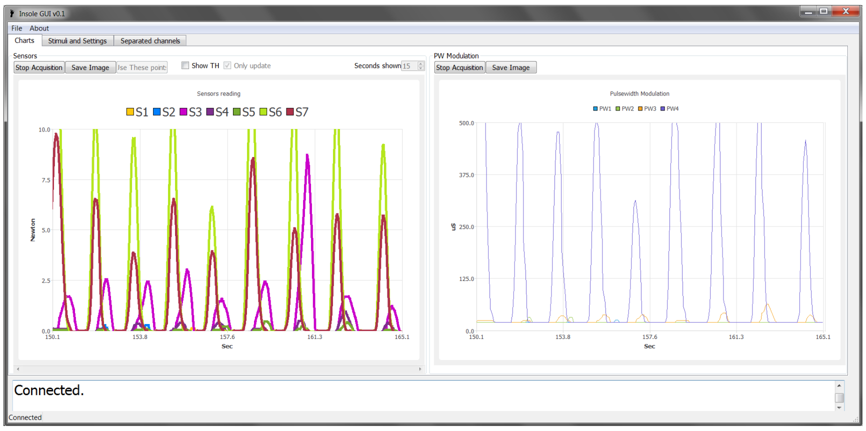
Real-Time GUI for insole-based applications with weight-balance controller

3.1 Concept

A first step was to develop a GUI to display the values read by the insole (eventually calibrated) in real-time. This is needed to facilitate prototyping and to evaluate the correctness of the system, and it's obviously mandatory in case of use of the system by a non-engineer staff, for example in a medical care facility. The main window of the developed software is shown on Figure 3.1.

Moreover, by an ad-hoc interface, the software allows to control the stimulation settings, such as, for each electrode: minimum and maximum stimulus, active sites (if multiple), minimum pressure, maximum pressure, as it can be seen in Figure 3.2.

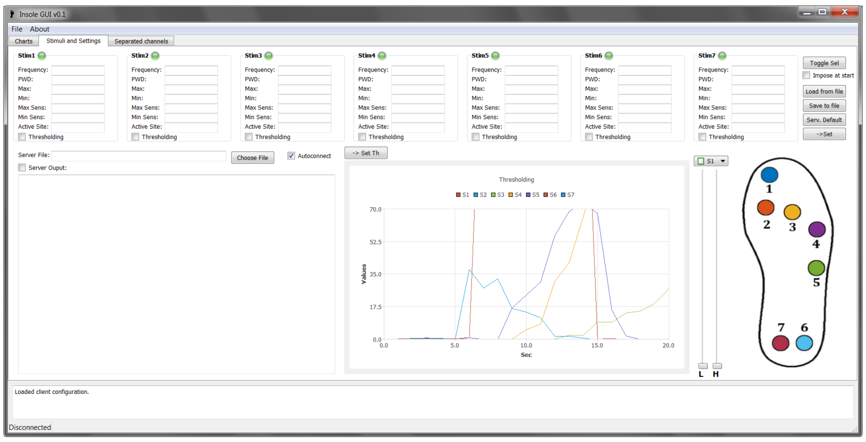
It was also decided to implement an additional feature to the GUI software, a **weight-balance controller**. As already explained in the introduction chapter, patients with



Connected.

Connected

FIGURE 3.1: Real-Time GUI, connected to the server, displaying read values from gait.



Loaded client configuration.

Disconnected

FIGURE 3.2: Settings window of the Real-Time GUI.

leg amputation tend to walk asymmetrically or with a non-ideal gait pattern. Amputees spend more time on the sane leg, which can be easily seen by a low **SI** (*Symmetry index* [1]) [11], and this can cause osteoporosis in the residual stump and osteoarthritis in the intact limb [17].

The balance controller had to provide a feedback (audio) to the patient, making sure he could adjust his balance accordingly. The type of feedback used is a negative feedback towards error, which has been proven to improve the SI [45]. The algorithm will be showed and deeply explained in the proper section but its part can be seen in the bottom right part of the Figure 3.2, and in Figure 3.3. The algorithm can work both on the server "stand-alone", or from the remote GUI client; the configuration may be set from the Client to the server trough an ad-hoc graphical interface embedded in the GUI.

3.2 Requirements for the GUI

The GUI had to follow some main guidelines requested for the application:

- Provide a real-time display of the 7 sensors values
- Run from the Raspberry (at least)
- Save the acquired data
- Implement an audio method to help the patient balance his weight
- Be relatively easy to use (i.e.: no requirement of deep knowledge to understand it)
- Be relatively easy to implement it in a health-care facility

[1]Symmetry Index (also known as Symmetry Ratio), defined as

$$SI = \frac{ST_{affected}}{ST_{intact}} \text{with ST being the stance time}$$

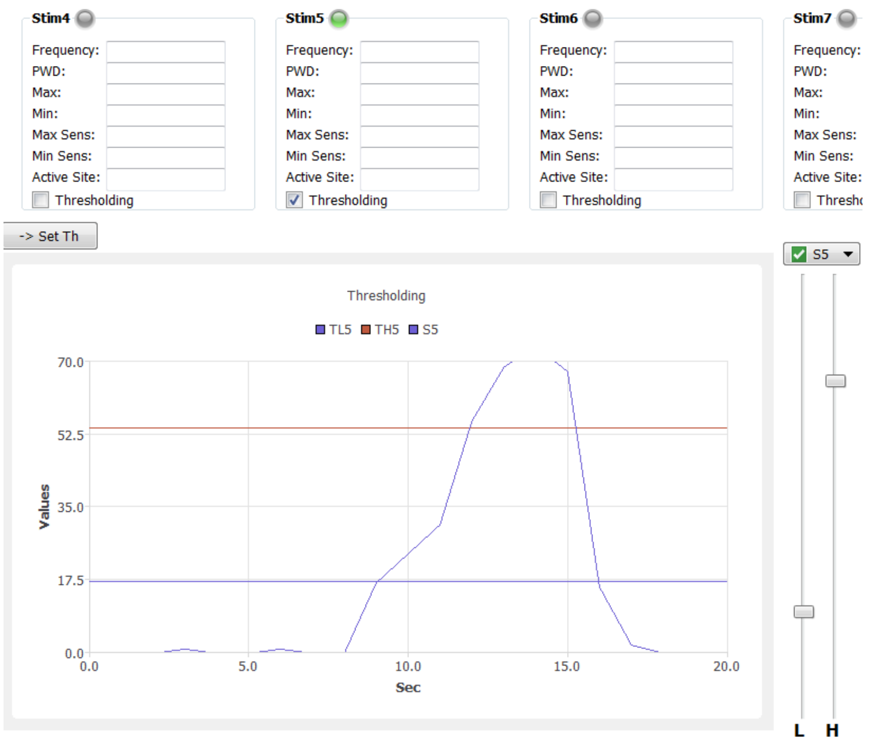


FIGURE 3.3: Settings of thresholds for balance control.

3.3 Design Choices

3.3.1 GUI as an external client

The GUI implemented can work either as simple *remote client*, which connects to the server (i.e. the raspberry, that runs as an ad-hoc WiFi hotspot) or locally, starting itself the server. The server is represented by the pre-existent code, opportunely modified in order to send network packets containing useful data.

The decision to separate the client program from the existing one is justified by many reasons:

1. The code on which the GUI has to be "applied" is quite nested and evidently sensitive to changes. Considering that it provides a neural stimulation, in order to be sure that it's still working 100 percent correctly, a "non-invasive" approach should be taken. The less invasive choice is to simply add a *networking part*, where the data read is sent as a **UDP**[2] packet.

2. The GUI needs to display a real-time plot acquisition, which may be burdensome for a "light" device like a Raspberry. This depends heavily on the resolution of the plot, on the rate of update, and on the saved points. Further discussion below.

3. Displaying a GUI running on a raspberry would require an HDMI cable and a monitor with HDMI input support, an USB mouse, and a USB keyboard. It can be argued that it's easier that the facility is equipped with a simple laptop or tablet, so both a remote and a local connection are available.

[2]User Datagram Protocol

4. The GUI-Client uses only *C++11* standard code and open-source libraries (**Qt5**[21]). This means that it can be compiled and executed also on the raspberry, in case that option is preferred. Not only that, but it could be ported easily on any device supported by gcc/Qt Libraries (e.g. Android Devices).
5. The GUI can run locally and connect to a local-running server self-started, as if it were a stand-alone program.

3.3.2 Libraries and programming language

In order to have a GUI, a *graphic library* associated with a programming language is needed. There are many different possibilities, so a selection must be made.

For the sake of maximum compatibility and support, it has been chosen only to use OpenSource libraries. The language used is C++11, which provides high efficiency and is supported on many devices. Furthermore, C++11 is used as language for the server, this provides the possibility to, eventually, join the two parts in only one program.

The graphic library should be at least compatible with C++11, be opensource, and have some sort of support for *charts* (needed for displaying the acquired data). For example, **wxWidget** could provide an easy library for displaying information, but it has no up-to-date chart support, thus would be useless.

The chosen one, **Qt**, comes with a Framework which covers everything needed. It's still constantly updated, and it has a wide compatibility with different OS, just to name few [23]: Windows 10, Windows 7, UWP[3], Linux, Android (form version 4 to 8), iOS, macOS. Even more useful, it has already a networking support and an UI-Design part.

The networking on the server, instead, will be handled using the **Asio library** [29] of the "**boost**" library pack [8]. This is a common and very well documented library pack, which was already used and included in the initial code-source files (i.e.: the reading and stimulation part running on Raspberry Pi 3), for timing applications.

To run tests on different devices from the Raspberry it's required to use the same version of the boost library, the *1.50*. This is quite important, since in newer versions some ASIO functions have slightly been changed. Instead, updating the library on the Raspberry would require the use of an unstable package-repository. This would in turn cause a massive update of many different packages, possibly leading to failures.

Since the actual sensor-board is powered by a 9V battery, many tests were run on a local server that simulated the presence of data. This server is again completely cross-compatible between different OS, but it had the essential constrain to use the above mentioned 1.50 boost library. The virtualization software is Oracle VM VirtualBox [32], using a Linux OS (Debian [36]).

[3]Universal Windows Platform

3.3.3 GUI Client running locally on the server

As stated before, the GUI can be ported on different architecture and OS; it was initially coded and designed on a Windows 7 system, but it has also been compiled and run on the Raspberry server, which uses a GNU/Linux OS and has also been tested on a Windows 10 laptop.

The modularity of the server/client obtains an independence of where the client must run, this is seen as an important consideration for future development and, potentially, modifications.

To run and compile the same source code on the Raspberry there is a requirement: The Qt version installed has to be >= **Qt5.4**. This is necessary [22] in order to use the Chart module (QChart), and provide a real time plotting. At the time of writing the latest version of Qt is 5.11, while the repository of pre-built software on the Raspberry gets up to 5.3.

For this reason, an updated version of Qt had to be compiled on the Raspberry. Compiling the whole Qt is not a light step, and it required about 8 hours [4]. It is required to skip some packages that are incompatible with Raspberry Pi 3 (such as qtlocation and qtwebengine) [40].

In the end the procedure resulted in the possibility to run the GUI also from the embedded Linux OS running on the Raspberry.

3.3.4 Modifications on the initial application: a non-blocking multithread implementation

The initial source code (executed on the Raspberry) provided only a written output on a console, containing the values read by the sensors, and everything run on the same thread. It has been decided to implement a *parallel thread*, dedicated only to the networking operations. As it will be shown, two more threads are added, one for the stimulation part, and one to implement the weight-balance control; obtaining 4 threads in total, leading to a multithread implementation.

Multithread it's a common way to develop a software, it can work either on different cores (for multi-core processors) or be a "switching of resources" on the same one. This provides the ability to wait for a network request (the connection of a client), while also print the output on the console, talk with the Bluetooth module and stimulate.

Non-blocking means that called functions will not block the server execution (i.e.: if it's required to write data on a network socket, this won't stop the execution of the code).

A non-blocking multi-thread approach is a necessity, with the use of **mutex** to increase the efficiency. Mutex, which stands for *mutual exclusion* avoids concurrent execution of critical parts of the code, which would either reduce the efficiency of the software (see next), or create **data races**.

Data races occurs whenever two (or more) different threads tries to access to the same memory without a specific "*hard-coded*" order. If at least one of the threads

[4]In our case, the version 5.9.6 was used.

wants to write over that memory, the execution is *non-deterministic*, leading to different results each run of the software [33]. A "safety" system is thus required, which in C++ is implemented with the use of the Mutex header[5] [10].

Another reason to use Mutex is to increase performance, this is done with the help of *condition variables* [9]. The idea is to "wake up" a thread only when needed. This server software is an easy example of why it's useful: the networking part has to send to the client (i.e., the GUI) a data-package only when a new set of read values is received from Arduino. The networking part is run on a different thread, and has to wait for a boolean[6] value that indicates the presence of a new data to send. If the CPU were to control over and over this variable, it would waste time. The thread can instead be "awakened" only once this value changes: this increases the performance, avoiding wasting CPU time.

The protocol chosen for the networking is **UDP**, since a fast and real time display of the values is required. UDP does not provide any built-in support for packet-read confirmation, thus in case of important communications (i.e.: not a single sensor read) a confirmation protocol has been implemented.

The server accepts only one connection at time as a security measure towards inconsistency. The networking implementation will be discussed in more details in the proper chapter.

Another important factor is the possible presence of an *increasing* lag between the Arduino and the server. In order to validate the absence of such problem, both the Arduino code and the server code must be changed. The former will be discussed in few sections; while for the latter, a packet-flush should be done *before* waiting for any new incoming packet. If no packet is ever flushed, it means that the server is fast enough to process and send the data, before a new one comes. For further details about delays evaluation, see Chapter 6.

Figure 3.4 shows the interconnection and signals between the various threads present in the server.

3.4 Implementation of the GUI

3.4.1 General concepts

The client is designed to have 3 different threads:

1. One for network communications (*UDP Thread*), which also implements:
   (a) A timer to send a "keep-alive" packet to the server[7].
2. One for the **balance-control**: *CTH Thread*.

[5]Headers are files that contain functions definitions that might be used in the software.

[6]A variable which value can only be either True or False.

[7]Keep-Alive packets do not contain any useful information. They are sent periodically to the server, to notify that the connection is still working. They are needed because UDP protocol does not use ACK (confirmation of read), thus in case of some network problem, there's no way for the server to know it. This way, if the client get disconnected for problems, the absence of Keep-Alive packets will inform the server.

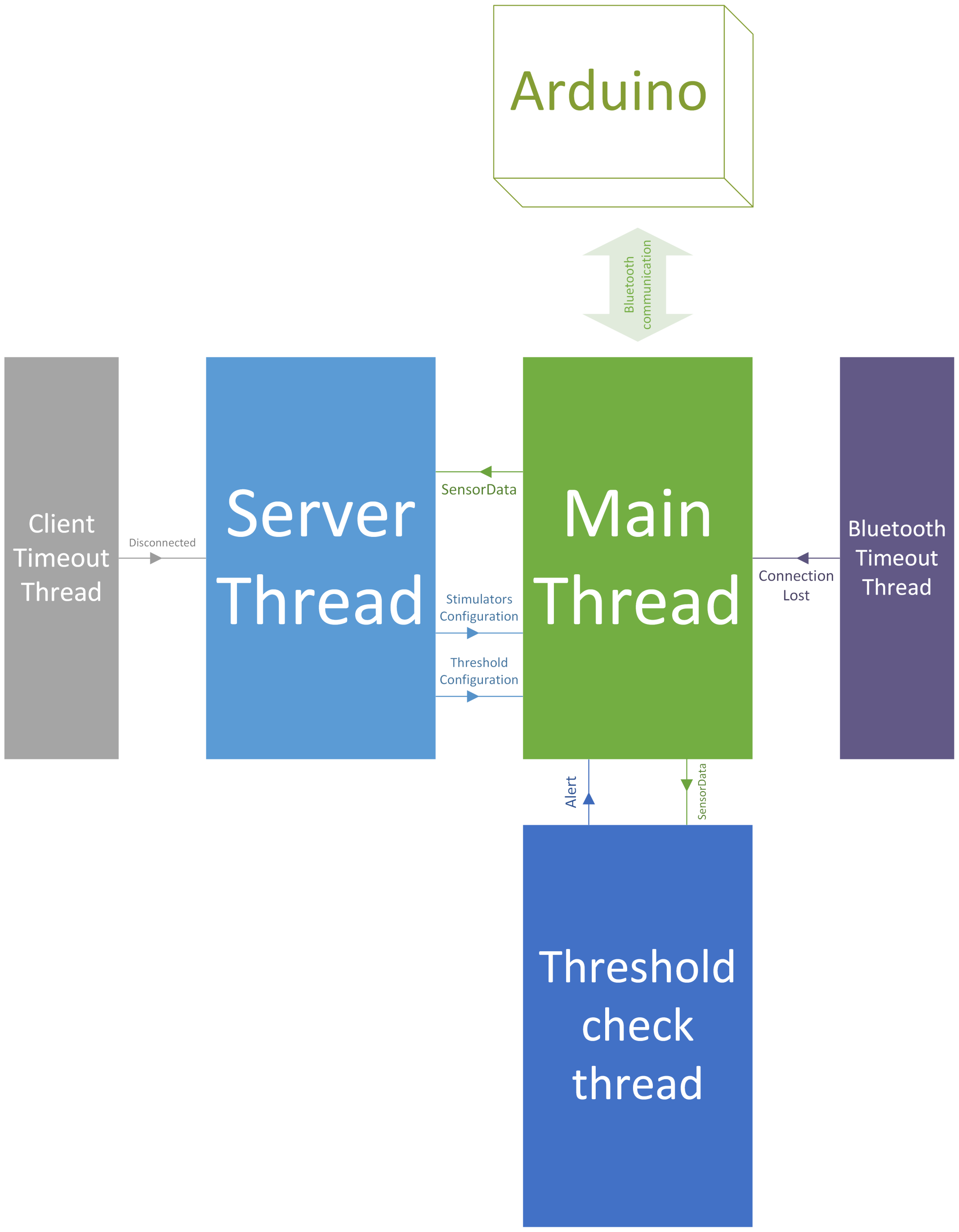


FIGURE 3.4: Signal used for communication between different threads in the server

3. One for the chart and visual update (*MainWindow*, - the main thread), which also implements:
   (a) A timer to update every 20ms the charts with all the new values.
   (b) A timer to check if we are still correctly connected to the server.
   (c) A timer to check for lag.

These threads must communicate through *signals* that can also carry data, and since they can have access to the same variables (as in the server-side code) mutex must be used to avoid race-conditions.

The used signals are[8]:

1. From the *UDP Thread* to the *MainWindow*:
   (a) **Append Text** – Used to add some strings to a GUI scroll-bar, which visualizes messages and statuses
   (b) **Write configuration window** – This notifies the GUI that a new stimulus-configuration has been read, and thus the GUI must be updated accordingly.
   (c) **Reset connection settings** – Once the connection has ended, the UDP thread notifies the MainWindow to prepare for a new connection (e.g.: clearing the chart, allowing a new connection, and so on).
   (d) **Start Timer** – Once a connection is established, we can notify the MainThread to start the "update-chart" timer (see next)
2. From the *CTH Thread* to the *MainWindow*:
   (a) **Alerted** – If the weight-control algorithm activates a sound alarm, it also notifies the MainWindow, this way it can potentially display the information on a chart, or save it for future references.
3. From the *MainWindow* to the *UDP Thread*:
   (a) **Disconnect UDP** – Initializes the procedure to disconnect the client from the server.
   (b) **Initiate UDP** – Start the connection to a given server.
   (c) **Send Configuration** – Sends a given stimuli configuration to the server.
   (d) **Send Threshold Configuration** – Sends a given Threshold Configuration (for the weight-balance algorithm) to the server.
4. From the *UDP Thread* to the *CTH Thread*:
   (a) **Control Values** – This makes the weight-control algorithm check the new sensors values

The Figure 3.5 shows the interconnection of signals.

[8]Excluding those for graphical user interactions, for example clicking on a button, which would be a large majority.

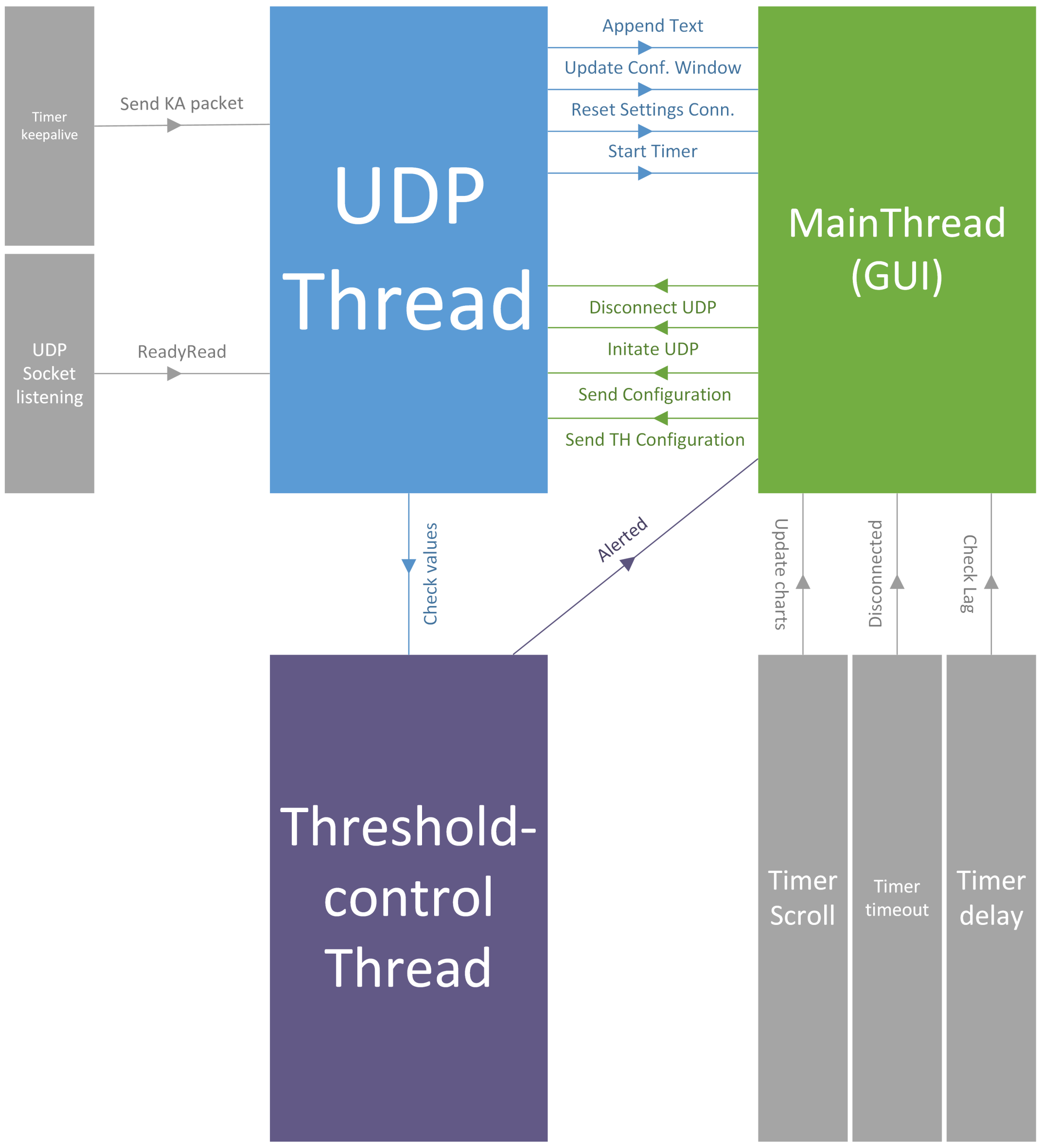


FIGURE 3.5: Signal used for communication between different threads in the GUI.

The thread with the highest priority[9] is the UDP, since no packet should be lost, and since the speed at which data is read should be faster than the speed of sent data (otherwise it's impossible to show real-time values).

3.4.2 Features implemented

The implemented features in the software are:

1. Remote network connection.
2. Local connection and automatic-server initialization.
3. Saving of the read data automatically to a new file.
4. Saving of the stimuli provided automatically to a new file.
5. Real-time display of sensors values.
6. Real-time display of stimulation.
7. Possibility to set the number of displayed seconds in the plot.
8. Possibility to pause/halt the update, and scroll the old chart values.
9. Possibility to "only update" the displayed chart with new values, to reduce memory usage. Old values are deleted from the memory.
10. A 2-threshold control-system for each sensor, used to create a negative-feeback towards low SI gait; with the possibility to use a recorded step as a reference for the configuration.
11. Display of the stimulation configuration.
    (a) Possibility to save it as an external file.
    (b) Possibility to set the wanted stimulation configuration.
    (c) Possibility to re-use the initial server configuration.
12. Possibility to activate/deactivate different sensors in the plot.
13. Display of the output of the local-server (i.e.: the output that would be on the console).
14. Automatic save/load of the configuration (path to the application, sensors activated, threshold imposed, . . .).
15. Time-out control to check the connection with the server.

3.5 Implementation of the balance-controller

The concept of the following algorithm is that patients walks asymmetrically because they trust less the prosthesis. The expected insole-pressure peak is thus lower, because it is assumed a brief, light, step on it. Under these assumptions, a step with values lower than a threshold starts an audio feedback. To reduce false-positives

[9]Meaning that it will have more allocated time to use the CPU

that may be generated by stance or non-gait pressures, the control is only initialized if a high variation in pressure is sensed (i.e.: a second threshold is used).

The balance-control algorithm has been implemented as follows.

For each sensor, and independently from the others, we can activate a thresholding algorithm. A first threshold (*TS1*) sets the minimum value that the sensor has to have, in order to "activate" the control. This avoids starting the algorithm in case of sitting, any non-walking activity, or presence of a noisy offset. A second threshold (*TS2*) sets a second minimum value (higher than the first one) that the sensor has to reach to be identified as a correct step. If the force applied is not enough, and the second threshold is not reached, an audio alarm (a loud beep) is emitted for 0.5 seconds.

Since any real signal will have a noise associated, a further control to avoid mis-activation is implemented: assuming that small superimposed variations to our signal could cause beeping, a third "threshold" (*SC*) is used.

If the signal has passed *TS1* but now it's lower, it will cause the negative feedback only if it's less than *TS1* by 10[10] (and, clearly, *TS2* has not been surpassed). This avoids triggering the control for small oscillations due to noise.

In case the value has already exceeded *TS1* by 10 (i.e.: the third threshold, *SC*) the feedback is triggered as soon as the signal is less than TS1 if, of course, TS2 has not been surpassed.

The flowchart in Figure 3.6 explains visually the concept.

3.6 Implementation of the networking

3.6.1 Server side

The server is implemented in the existing code, which runs on a Raspberry[11]. The library used for network operations is Asio, included in the boost library-pack for C++, which was already used in the initial code. The included boost version is 1.50, which is really important since the most recent version of boost have some differences in the functions included.

To assure the correct working of the reading/stimulation part of the program, a different Thread is created, which will perform the network actions. Communications between the two threads (the main one and the added one) will be performed through 'mutex' which allows a boost in the performance, with the cost of a bit of complexity.

Only one connection at the time is allowed, to assure consistency and reliability of the data.

The network and communication algorithm works as follows:

[10]This value refers to a digital value, with a calibrated insole it should be expressed in newton. The ADC runs at 12bits, leading to 4096 possible values, thus "10" is 0.2% of the fullscale range.

[11]Actually, the Raspberry has been replaced with an Odroid to improve the performance. This will be introduced in Chapter 6.

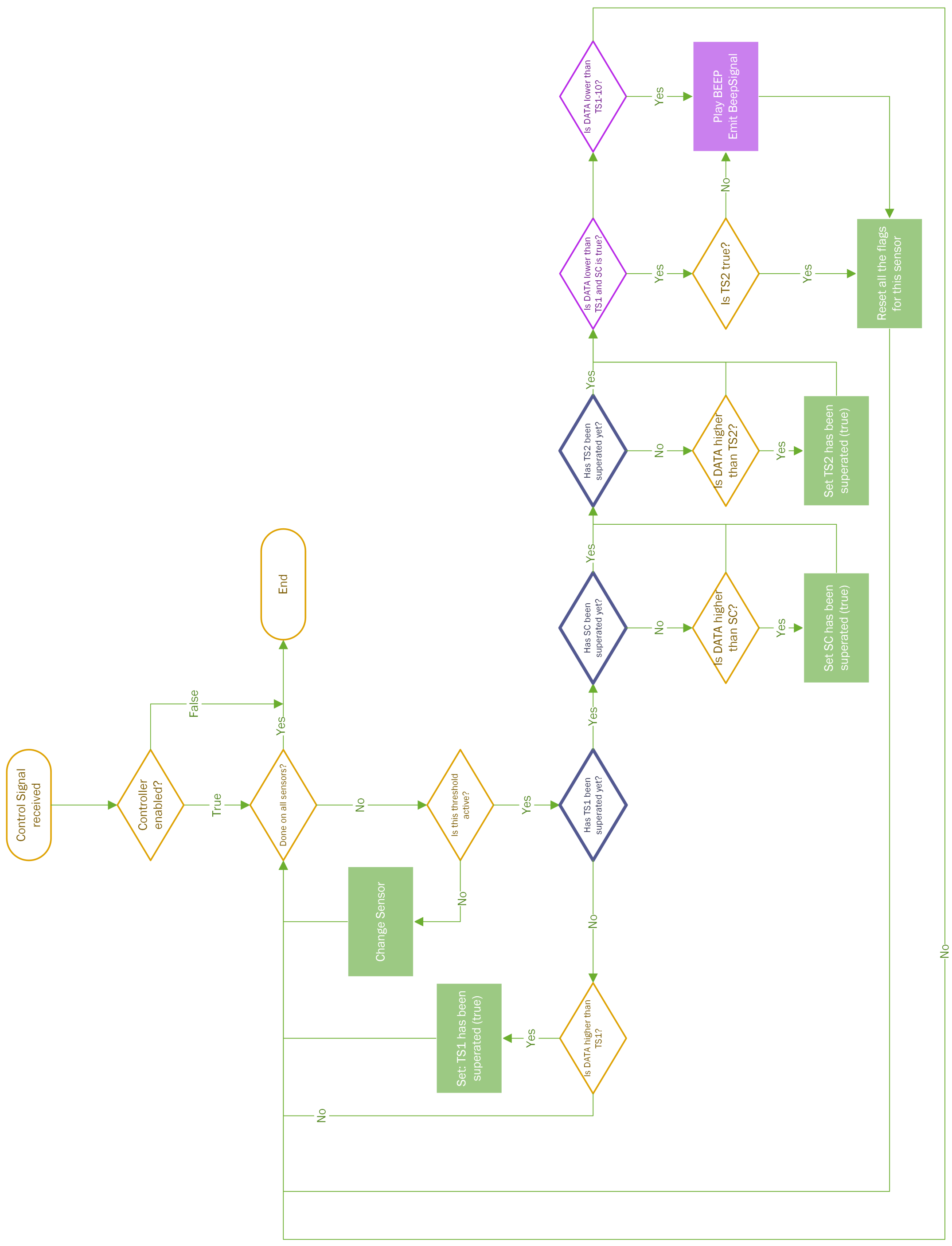

Control Signal received
Controller enabled?
True
False
Done on all sensors?
Yes
End
No
Is this threshold active?
No
Change Sensor
Yes
Has TS1 been superated yet?
No
Is DATA higher than TS1?
Yes
Set: TS1 has been superated (true)
No
Yes
Has SC been superated yet?
No
Is DATA higher than SC?
Yes
Set SC has been superated (true)
Yes
Has TS2 been superated yet?
No
Is DATA higher than TS2?
Yes
Set TS2 has been superated (true)
Yes
Is DATA lower than TS1 and SC is true?
Yes
Is TS2 true?
Yes
Reset all the flags for this sensor
No
Is DATA lower than TS1-10?
Yes
Play BEEP
Emit BeepSignal
No

FIGURE 3.6: Flowchart of the algorithm used in the weight-balance negative feedback control.

1. After an initialization of the socket[12], it waits for the main part of the program to have read the stimuli configuration from a file.
2. Once the configuration is read, it can start accepting a new connection. This wait is needed, because the first packet which will be sent is indeed the settings one.
3. Then the program awaits a connection from a Client, accepts one and sends a packet containing the configuration.
   (a) A packet of confirmation is waited for. In case something else is received, the configuration is sent again.
4. A loop is now entered, which will be run until the client disconnects, or errors are reported. In these cases the loop is exited, and the server gets ready for a fresh new client.
5. Once the connection has been established, depending on the packet received it will:
   (a) Set a new configuration[13].
   (b) Set a new threshold configuration[14].
   (c) Initialize disconnection[15].
   (d) Send again the configuration[16].
6. Sensor's data availability is waited for.

   "*Data availability*" consist in a single read of all the pressure sensors, and it signaled through a mutex[17]. Using a mutex means that the network thread will be locked until a signal is generated. This is useful to avoid waste of CPU resources which would just run continuously a *while()* to check a boolean variable.
7. Once the mutex receives a signal, and a boolean flag is active, it means data have been read, and a packet can be written on the socket. The packet is sent to the client.
8. The Thread is locked again.
9. Repeat form "point 5."

The Figure 3.7 displays with a flowchart the server network operations.

3.6.2 Client side

Since the GUI uses the *Qt Framework*, its network module can be used.

[12]That is, an UDP endpoint on the OS.
[13]*Packet with code: 03.*
[14]*Packet with code: 04.*
[15]*Packet with code: 99.*
[16]*Packet with code: 05.*
[17]Indeed the read happens on a different thread, dedicated to the communication via bluetooth to the Arduino.

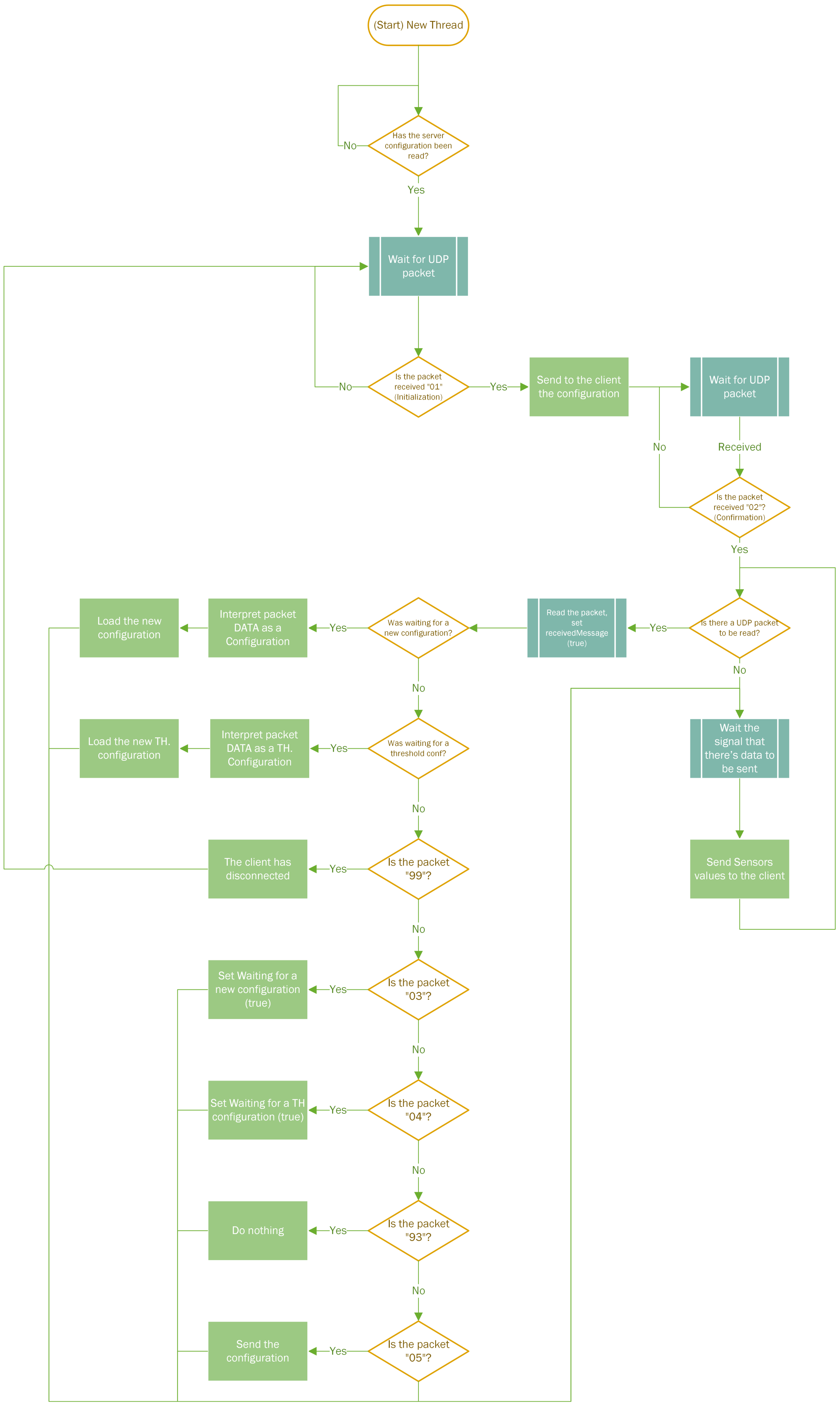

(Start) New Thread
Has the server configuration been read?
No
Yes
Wait for UDP packet
Is the packet received "01" (Initialization)
No
Yes
Send to the client the configuration
Wait for UDP packet
No
Received
Is the packet received "02"? (Confirmation)
Yes
Is there a UDP packet to be read?
Yes
Read the packet, set receivedMessage (true)
Was waiting for a new configuration?
Yes
Interpret packet DATA as a Configuration
Load the new configuration
No
Wait the signal that there's data to be sent
Send Sensors values to the client
Was waiting for a threshold conf?
Yes
Interpret packet DATA as a TH. Configuration
Load the new TH. configuration
Is the packet "99"?
Yes
The client has disconnected
Is the packet "03"?
Set Waiting for a new configuration (true)
Is the packet "04"?
Set Waiting for a TH configuration (true)
Is the packet "93"?
Do nothing
Is the packet "05"?
Send the configuration

FIGURE 3.7: Network flowchart of the server.

The Network is controlled in a stand-alone thread, different from the main one. Indeed the Main one provides the graphical interface; thus, once the user has chosen to connect to a specific server, a signal is emitted towards the *UDP Thread*.

When the UDP Thread receives the *IniateUDP signal*, the following happens[18]:

1. A local UDP instance will start listening on port 9777 for incoming messages[19].
2. A packet with *initiate-connection* code is sent to the server.[20]
3. A packet is received from the server[21].
   (a) If the configuration was yet to be read, the packet-size is controlled:
      i. If it's not equal to the expected (i.e.: the configuration size) a packet is sent to request a new configuration[22].
      ii. If it's the correct size the configuration is read.
4. A packet is sent to confirm the reception[23].
5. The signal *startTimer* is emitted to the *MainWindow*. **Timers** are needed while packets are received, but they are controlled by the MainWindow thread.
   (a) If the configuration was set, the packet is assumed to be a new sensors read, then the following happens:
      i. The *CTH Thread* is notified to control the new values with its weigh-balance algorithm.
      ii. If allowed by the Mutex, the data is added to the list of points.
      iii. The new values are written to a file, for saving purposes.
      iv. The received packets during this operation are flushed[24].
6. If the signal "*Send Configuration*" (emitted by the MainWindow) is received, a packet with code '03' is sent to the server, followed by the configuration.
7. If the signal "*Send Threshold Configuration*" (emitted by the MainWindow) is received, a packet with code '04' is sent to the server, followed by the new configuration for the weight-balance algorithm.
8. If the signal "*Disconnect UDP*" (emitted by the MainWindow, upon user request) is received, a packet is sent with code '99'[25].

The Figure 3.8 shows the typical network flow and packets: first a connection is initialized by a packet with *code 01*. Then the CONF (the packet with the configuration) is received, and it's confirmed with a *code 02*. After that each part enters a loop:

[18] Assuming it was not already connected
[19] Indeed the client will expect packets with data, and this has to happen on a different port from the server one (which is 9666).
[20] As any other "code" packet implemented, and used to communicate decision between client and server, its content-length are only 2 bytes, in this case “0x00 0x01".
[21] This causes an emission of an event, from the udp-socket instance to the Udp Thread.
[22] Code '03'
[23] Code '02'
[24] Disregarded.
[25] This is quite important, indeed in absence of an explicit packet, the server would never be notified of the disconnection, since UDP Protocol does not provide an ACK by design.

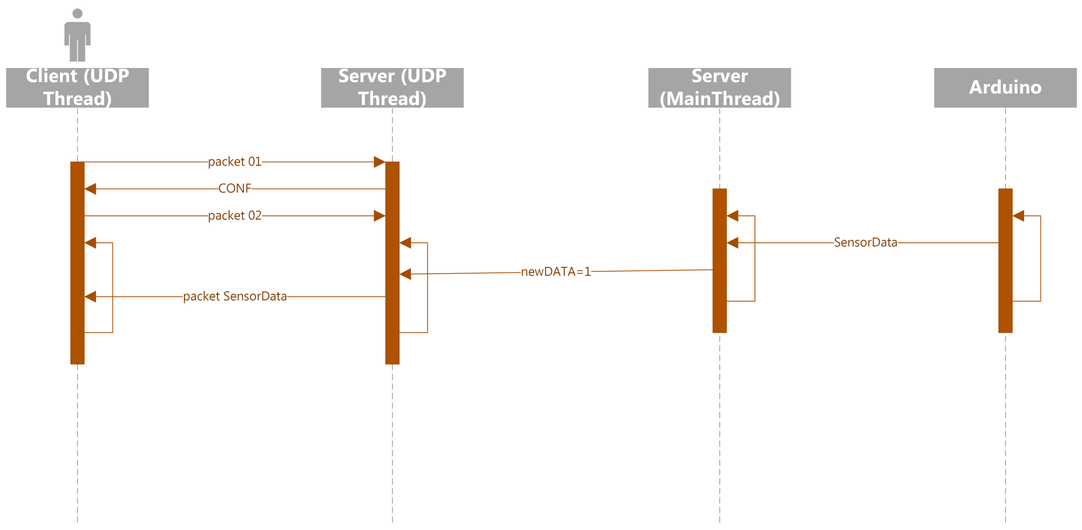


FIGURE 3.8: Typical network flow and packets of the various blocks.

- Arduino samples the sensors with an interrupt, and send a packet via bluetooth.
- Once the data is read by the server, newDATA is set to 1.
- The network thread on the server checks for newDATA, once it's 1 sends the sensors values to the client.
- The client receives a new pack of sensors values and shows them in the charts.

3.7 Arduino modifications

The original[26] Arduino code samples continuously the sensors values, and sends them via bluetooth. This is a problem for different reasons:

1. It doesn't use an interrupt to provide a certain sample time: this means that the period between each sample might vary substantially and randomly
2. It doesn't include a numerical index associated to the sample (i.e.: a counter). This leads to the impossibility to verify and check for packet loss
3. It doesn't transmit at a speed acceptable to the server. The server on the raspberry is not fast enough to read a packet, before a new one is sent. This creates a increasing lag over time.

The chart plots in the GUI display the time in the abscissa, which is the only useful way to provide information. To do so the Arduino code had to be changed because of point (1): since the sampling period was unknown, the only way to associate a set of sampled values with a time index was to check (on the server) the time at which the packet was read. This was really inaccurate and suffered from random unpredictable variations.

The modified version now samples at intervals of 30ms, using a Timer and an Interrupt. The sent packet now contains also an index that is increased at each sample,

[26]From the initial starting project

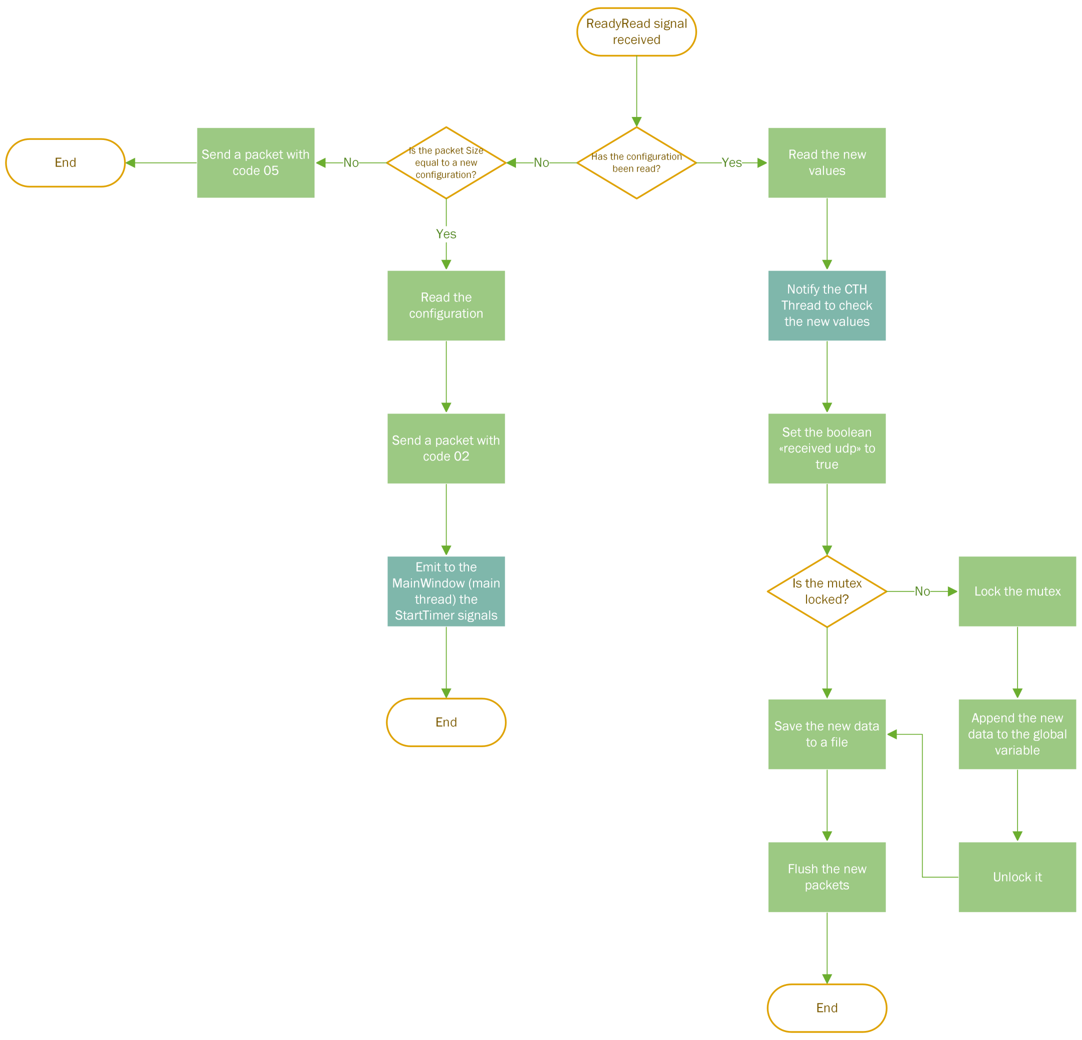

ReadyRead signal received
Has the configuration been read?
No
Yes
Is the packet Size equal to a new configuration?
No
Send a packet with code 05
End
Yes
Read the configuration
Send a packet with code 02
Emit to the MainWindow (main thread) the StartTimer signals
End
Read the new values
Notify the CTH Thread to check the new values
Set the boolean «received udp» to true
Is the mutex locked?
No
Lock the mutex
Append the new data to the global variable
Unlock it
Save the new data to a file
Flush the new packets
End

FIGURE 3.9: Algorithm used in the client after an UDP packet is received

providing the possibility to check for packet-drops. The latter consideration lead to the possibility to verify that 30ms is an acceptable sampling period, meaning that the next two steps (server on the odroid, and then Client-GUI) will not present an increasing lag due to their code being too slow. Refer to Chapter 6 for further discussion about delays.

Chapter 4

Integration of the TENS

4.1 Introduction

The RehaStim is a portable electrical stimulation device, produced by Hasomed, that generates impulses on up to 8 channels simultaneously, it can be used as a portable (it contains a battery) or stationary device [24].

For availability reasons, RehaStim1 has been the device used for electrical surface stimulation in this thesis. It's a common and famous TENS, used for many different purposes, included but not limited to:

- FES[1] .
- Relaxation of muscle spasm.
- Prevention or retardation of disuse atrophy.
- Increasing local blood circulation.

In the case of this project, the aim is to induce a distal (i.e.: far from the electrode) sensation, by activating the associated nerves, using this way a non-invasive stimulation.

The position on which to stimulate highly depends on the patient, and in the Chapter 7 the results will be shown and analyzed.

[1]Functional Electrical Stimulation

FIGURE 4.1: RehaStim device.

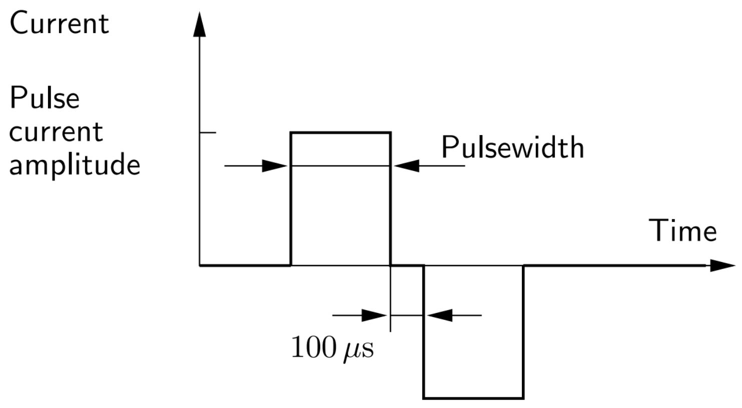


FIGURE 4.2: A generic stimulus generated by the RehaStim device.

4.2 Hardware

The technical specifications are [25]:

Size and Weight	
Length	13.5cm
Width	15cm
Height	7cm
Shipping Weight	5kg

Stimulator	
Display/Interface	Touch-sensitive LCD
Communications	USB-RS232
Maximum voltage output	150V
Maximum number of channels	8
Current output per channel	0-126mA, steps of 2mA
Type of pulse	Biphasic
Pulsewidth	0,20-500uS in 1uS steps

The device has a main **MSP430** microcontroller, and two stimulation blocks, each one controlled by a different MSP430: obtaining that the 8 channels are divided in 2 groups of 4 channels each.

Each module generate its own stimulation timing independently, thus concurrent stimulation of the two parts is possible[2].

4.3 Stimuli

The stimuli provided are biphasic square pulses, as seen in image 4.2.

In "*science mode*" (a working modality of RehaStim) it is possible to set the amplitude and pulsewidth of the stimulation, while having a fixed 100uS between the two phases. Depending on the configuration is possible to achieve 3 different type of generation:

[2]Actually, as described below, a delay between the two parts of a fixed 0.6ms will always be present.

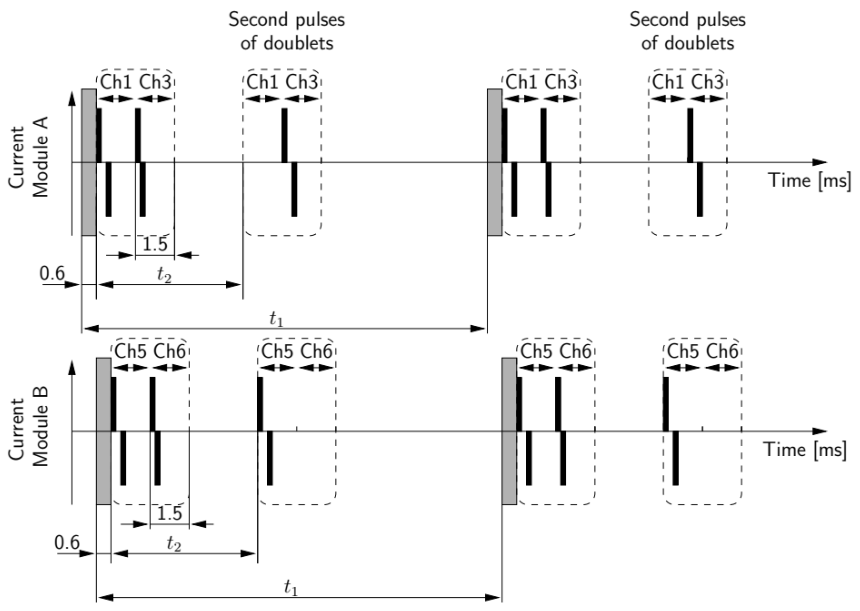


FIGURE 4.3: Continuous Channel List Mode, visual explanation.

Single Pulse – A command is sent from a device (e.g. a PC), and a single pulse, from a single channel, is generated. All the timing will be decided by the controlling device. This is the used configuration for the mapping step.

One Shot Channel List – This modality allows to set multiple-channel concurrent stimulation and executes it only once. This is the used configuration for the sensory feedback neuroprosthesis. The stimulation is not perfectly concurrent as there are some fixed delays between the two Modules, and each module can stimulate one channel at time. See the next chapter for an evaluation on the delays.

Continuous Channel List – The remote device can select the amplitude and pulsewidth of various channel, a main period time t2 of repetition of the stimuli, and a smaller inter period time t1 of repetition of triplets or doublets. As in the One Shot Channel List there's a fixed delay between channels of the same module, 1.5mS, while between the two modules there's a delay of 0.6ms. This way the PC only has to activate the stimulation and the device accounts for the repetition and the frequency. Another command to stop the stimulation has to be sent. See Figure 4.3 for a visual explanation. This modality is **not** used in this work.

Before each stimulation RehaStim checks the load resistance with a low intensity pulse, and if the value exceeds a range, the stimulus is not provided.

4.4 Communication

The communication works via RS232 (emulated over USB) with the following settings:

Depending on the stimulation mode, different commands are possible to be sent.

Single pulse generation command does not need any prior initialization, and simply requires:

Serial Settings	
Baudrate	115200
Parity	no
Data bits	8
Stop bits	2
Flow Control RTS/CTS	on

- **Channel Number** – The output channel for the stimulation.
- **Pulse Width** – Pulsewidth in uS of the two biphasic square waves[3].
- **Current Amplitude** – Provided current amplitude in mA.

While, for the **One Shot Channel List**, first an initialization packet is required (known as **Channel List Mode Initialization Command**[4]) providing:

- **N Factor** – This is used to define the ratio of the low-frequency channels. It's not used in this thesis.
- **Channels to Stimulate** – List of the channels to stimulate.
- **Channel low-frequency** – List of the channels that needs a low-frequency stimulation. It's not used in this thesis.
- **Group Time** – Defines the interpulse-interval t2, refer to the Figure 4.3 for an easier display. It's not used in this thesis.
- **Main Time** – Defines the main time period t1, refer to the Figure 4.3 for an easier display. It's not used in this thesis.

After this packet, to stimulate a **Channel List Mode Update Command** must be sent, containing:

- **Mode** – This sets the device to stimulate with either single-pulse (our case, for a OSCL) or to generate doublet, or to generate triplets.
- **Pulsewidth** – Pulsewidth of the stimulation for the X's channel, in uS.
- **Current Amplitude** – Current Amplitude of the stimulation for the X's channel, in mA.

Each time a command is sent to the device, RehaStim answers with an acknowledgement packet.

The software implemented sends a stimulation-burst for the desired amount of time, with a choosable frequency. The frequency used in all the experiments was 80Hz, this provided a more natural feeling to respect a low-frequency mode[5], while avoiding encountering problems with higher frequencies. Using higher rates indeed might be a problem for multi-purposes Operative-Systems like Windows 7 or Linux.

[3]Note that both the positive and negative phases will be changed

[4]Note: this is the same for CCL and OSCL

[5]See Chapter 7 for an evaluation about stimulation settings and how they affect felt sensations.

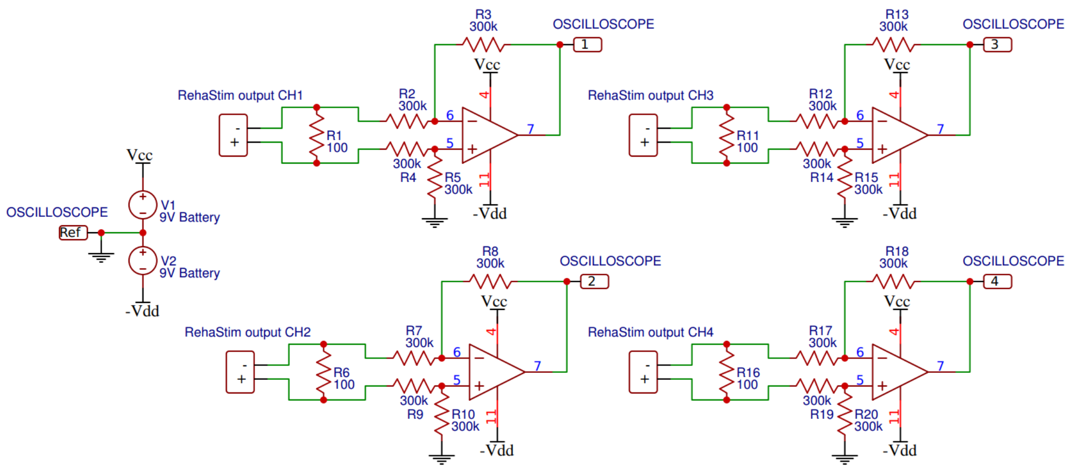


FIGURE 4.4: Electronic circuit used to test the system with an oscilloscope.

4.4.1 Implementation

The library that has been developed is in C++ and it's been used for both the Mapping Software and the sensory feedback running on the raspberry (see the next chapter for an evaluation of the obtained whole project).

The prototype testing was done on a linux virtual-machine and the output of rehastim has been checked with an oscilloscope before using it on human subjects.

To see multiple channels with the same oscilloscope, the small electronic circuit presented in Figure 4.4 is needed.

As it will be shown in the next chapter, a similar circuit has to be made to correctly estimate the delay time of rehastim.

The communication protocol has a strict bit-wise structure, meaning that each packet sent[6] has a specific role, and it's non-ASCII.

To implement the library, the *bit-field data structure* in C++ has been used. This allows to define data of the right size, depending on the packet, and to access each bit easier. Indeed one difficulty in programming this library is that some settings might be divided in different packets: for example, for the definition of a *SPGC*[7] four Bytes are needed; the *PulseWidth* is partially written on the bits of the Byte 2, and partially on bits of the Byte 3.

As an example of the method, the following C++ structure is used to generate a packet to control a single pulse:

```
struct SinglePulseGenerationCommand {
        unsigned char Pulse_Current : 7;
        unsigned char zero2 : 1;
        // Byte 4
        unsigned char Pulse_Width_L : 7;
        unsigned char zero1 : 1;
        // Byte 3
        unsigned char Pulse_Width_H : 2;
        unsigned char : 2;
```

[6]8bit packets.
[7]Single Pulse Generation Command

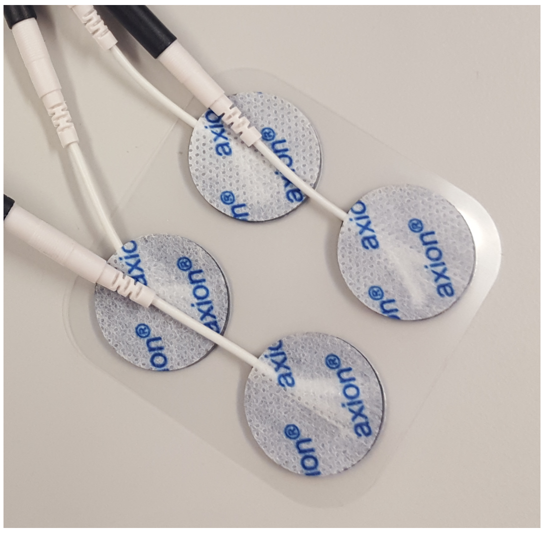


FIGURE 4.5: Circular electrodes used.

```
        unsigned char Channel_Number : 3;
        unsigned char zero0 : 1;
        // Byte 2
        unsigned char Check : 5;
        unsigned char Ident : 2;
        unsigned char one : 1;
        // Byte 1
};
```

Depending on the computer architecture, which can be either Big Endian or Little Endian, the order of the bytes can be the opposite. This is the most common case, the Little-Endian.

4.5 Electrodes

The electrodes used are produced by Axion, they are circular with a diameter of 2cm. For our purposes this was the best match of the ones available, being the smallest. As a reference, by reducing their size manually it was found that square electrodes of dimension 1x1cm still elicit distal sensation.

This leads to the assumption that in future works matrix-arrays electrodes could be used, obtaining more spatial freedom and eventually accuracy in the results. Of course this would need a different type of TENS, capable of providing more than 8 concurrent channels.

Chapter 5

Mapping Software

5.1 Reasoning

The concept of electrical stimulation in order to elicit somatotopic sensations, linking them to force sensors, has already been tested and proved working for upper limb amputees; both in non-invasive [12] and invasive [37] ways.

The implementation in lower-limb amputees and diabetic people, with the specific intent of restoring a controlling loop, is a novelty. In this case a sensorized insole will be the source signal by which drive the stimulation.

Since the sensation has to be somatotopic to help the patient reacquire the missing control-loop functionality, the perceived feeling must present itself in a known and defined location. This location has also to be the same as the position of the force sensors, in the insole, or close as possible to them.

In invasive stimulation, electrodes are directly inserted in the nerves; while in non-invasive stimulation surface electrodes are used. In both cases an a priori knowledge of the elicited sensation is impossible; of course depending on which nerve is stimulated it can be estimated at least the region of the perception, but more accuracy is clearly required.

Thus, in both cases, a mapping step is needed, associating:

- Electrodes location.
- Stimulus configuration (namely: PW, frequency, amplitude); see next chapters for details about the stimulation.
- Type of feeling elicited and its intensity.
- Location of the felt sensation.

This has to be done for each electrode position, for each active site, and the result is highly patient dependent due to the highly variability subject-to-subject. Figure 5.1 visually explains the concept. In case of implanted electrodes their position is fixed, instead in the non-invasive case, such as the one of this thesis, the mapping can be repeated moving the electrodes until a preferable result is obtained.

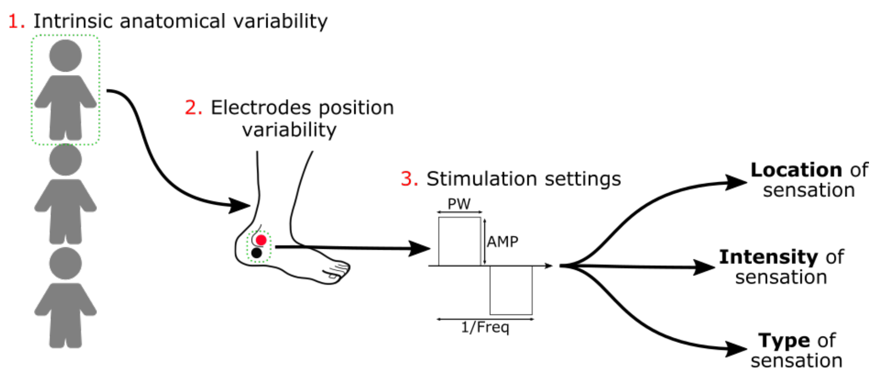


FIGURE 5.1: Concept of mapping: variability associated with the stimulation.

5.2 Requirements

Conceptually the project covers both diabetic people with DPN, and lower limb amputees; either with implanted/non implanted electrodes.

For this reason the request for the Mapping Software was to be as broadly usable as possible: meaning that it should be possible to use it in any case, with the least possible modification in terms of code.

The software should first electrically stimulate the patient, and then ask him to characterize the sensation felt - if any. The characterization should include:

1. A list of sensations.
2. Location of the distal (i.e.: on the foot) sensation.
3. An estimation of the intensity of the sensation.
4. An estimation of the pain felt under the electrodes.

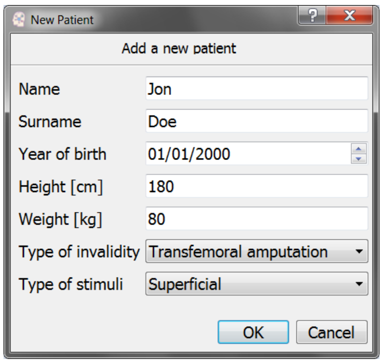
While for amputees it should also include

1. Muscles felt twitching.
2. Ankle proprioception.

5.3 Features implemented

A quick summary of the implemented features is:

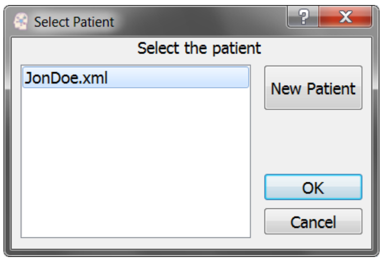
1. Patient database: the possibility to save/load a patient configuration file. Shown in Figure 5.2 and 5.3.
2. Real-time graphs showing the current electrical stimulation.
3. Possibility to change between Rehastim and Bonestim stimulators.
4. Possibility to add/remove new electrodes.
5. Possibility to add/remove the sensations selectable by the patient.
6. Possibility to choose between a pre-fixed "square" division for the location, and a freehand method (i.e.: a circle of selectable radius).


New Patient

Add a new patient

Name	Jon
Surname	Doe
Year of birth	01/01/2000
Height [cm]	180
Weight [kg]	80
Type of invalidity	Transfemoral amputation
Type of stimuli	Superficial

FIGURE 5.2: Mapping GUI: window to add a new patient.


Select Patient

Select the patient

JonDoe.xml

FIGURE 5.3: Mapping GUI: window to load a patient and its configuration.

7. Possibility to automatically stimulate with:
   (a) Pulsewidth modulation.
   (b) Frequency modulation.
   (c) Amplitude modulation.

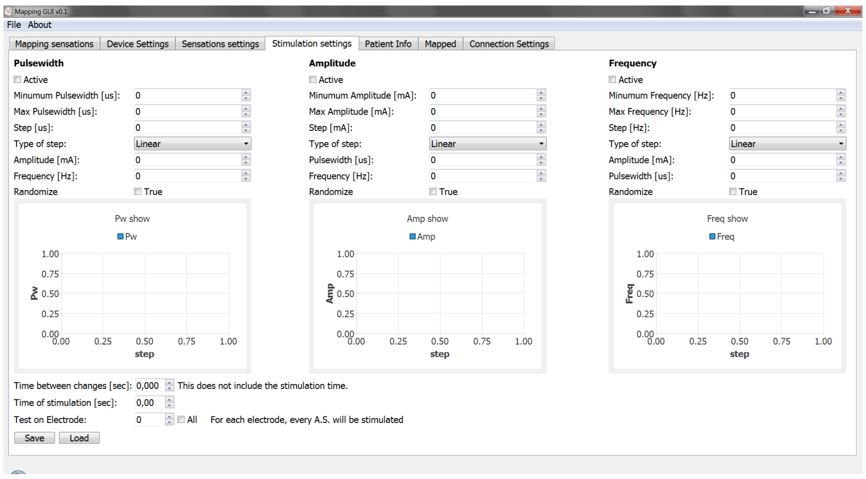
   for which a range of min/max with linear step can be set
8. Randomization of the stimuli order.
9. Possibility to set the duration time of the stimulation.
10. Discrimination test.
11. A mapping visualizer.

The main software windows are shown in Figure 5.4 for the settings of the stimuli, and Figure 5.5.

5.4 Software Architecture Design

Like the Client-GUI shown in Chapter 3, the Mapping Software is a multithread solution as well, and uses Qt libraries with C++. Three threads are used:

- One main thread that shows the graphical interface and interacts with the user: – "*MainThread*".
- One thread for communications with the electrical stimulator: – "*CommThread*". *Time-critical priority*.



Pulsewidth

Active	☐
Minumum Pulsewidth [us]:	0
Max Pulsewidth [us]:	0
Step [us]:	0
Type of step:	Linear
Amplitude [mA]:	0
Frequency [Hz]:	0
Randomize	☐ True

Amplitude

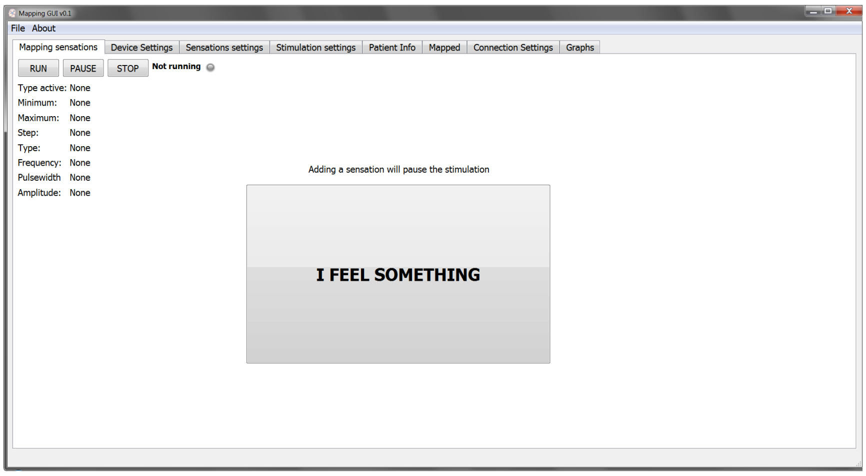
Active	☐
Minumum Amplitude [mA]:	0
Max Amplitude [mA]:	0
Step [mA]:	0
Type of step:	Linear
Pulsewidth [us]:	0
Frequency [Hz]:	0
Randomize	☐ True

Frequency

Active	☐
Minumum Frequency [Hz]:	0
Max Frequency [Hz]:	0
Step [Hz]:	0
Type of step:	Linear
Amplitude [mA]:	0
Pulsewidth [us]:	0
Randomize	☐ True

Time between changes [sec]: 0,000 This does not include the stimulation time.
Time of stimulation [sec]: 0,00
Test on Electrode: 0 ☐ All For each electrode, every A.S. will be stimulated

FIGURE 5.4: Mapping GUI: settings for the stimulation.



Not running

Type active:	None
Minimum:	None
Maximum:	None
Step:	None
Type:	None
Frequency:	None
Pulsewidth	None
Amplitude:	None

Adding a sensation will pause the stimulation

FIGURE 5.5: Mapping GUI: main user window.

- One thread to generate stimuli: – "*StimGen*". *High priority*.

To communicate between the treads "*signals*"[1] are used, since the memory space of different threads is separated. The signal implemented are:

1. Generated by **StimGen**:
    (a) To MainThread: "Update Output Graphs". Once a stimulation settings has been generated, it's being sent to the MainThread so to update the charats.

    (b) To CommThread: "Stimulate At Freq X for Y". Once a stimulation settings has been generated, it's being sent to the communication thread to initialize a stimulation that lasts for a decided amount of time, with a decided frequency.

    (c) To CommThread: "Abort Stimulation". This abruptly stops the current stimulation.

    (d) To MainThread: "Stimulation Finished". This notifies the MainThread that the set of stimuli pattern is over.

2. Generated by **MainThread**:
    (a) To CommThread:

        i. "Send Unit Test". Makes the device stimulate with the lowest possible intensity in the first channel. This is required to be run prior actually stimulating any subject, as a security measurement to check everything is working.

        ii. "Open Port for Communication". Makes the software initialize a serial communication with the saved serial port.

        iii. "Set Port And Device". Changes the selected Device[2] and the communication port.

    (b) To StimGen:

        i. "Start generating stimuli". Initialize the generation of stimuli patterns.

        ii. "Pause generating stimuli". Pause momentary the generation of stimuli patterns.

        iii. "Resume generating stimuli". Resume the paused generation of stimuli patterns.

        iv. "Stop generating stimuli". Ends the current set of generation of stimuli patterns.

3. Generated by **CommThread**:
    (a) To MainThread: "Port opened correctly". Indicates that the operation to take ownership of the COM port have been successful.

Figure 5.6 shows the signals between different threads.

[1]They are a Qt Framework ad-hoc implementation.
[2]Currently either RehaStim or Bonestim.

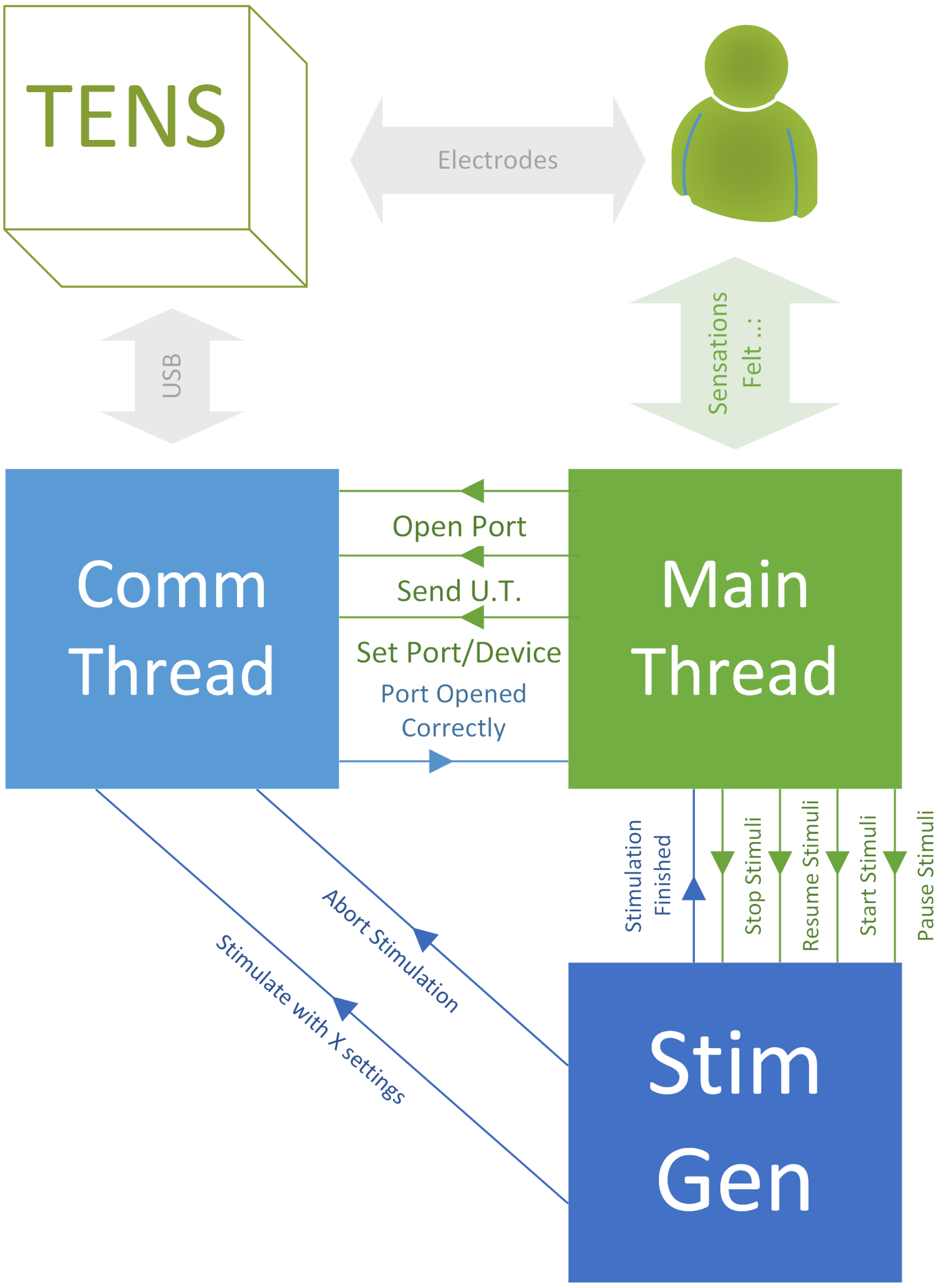


FIGURE 5.6: Mapping GUI: signals used for multi-thread communications.

5.5 Implementation

In the following section the most important parts of the software will be analyzed and described.

5.5.1 Stimulation

In order to map correctly a subject for each electrode position, there must be a sweep in terms of stimulus: varying one stimulation setting while keeping the others fixed, recording the effects.

The stimulation can be chosen to be applied as **Pulsewidth modulation**, **Frequency modulation**, or **Amplitude modulation**. For each of these possibilities a minimum, a maximum, and a step value can be set. Stimulus' duration is also to be set.

There are four ways of stimulation:

- **Simple non-random sweep** – the variable of interest, for example Amplitude, is made increasing from the minimum to the maximum, using the chosen step. The stimulation occurs every X chosen seconds, and lasts for Y chosen seconds. If requested, a pop-up window lets the subject map the sensations.
- **Simple Random sweep** – It's the same as the previous case, except the stimuli order is randomized. This is useful to map a subject without inducing bias.
- **Weber** – A couples of stimulus are provided each time: a reference and an increasing one. After each iteration the subject is asked to tell if the second stimulus was higher, equal, or lower. In case higher is chosen, the stimulation stops.
- **Discrimination test** – A couples of stimulus are provided each time: a reference and a variation. The variation can be lower, equal, or higher than the reference, and sweeps the whole range decided in the configuration window. The stimulation ends when all the range has been used as variation.

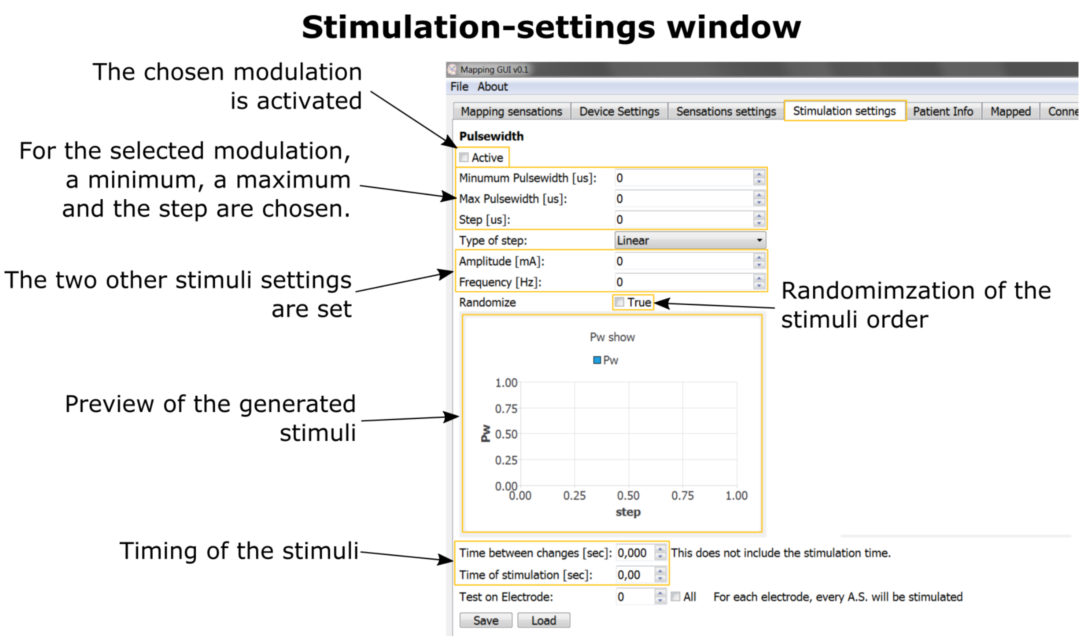
Each time a stimulation is started it can always be stopped or paused.

5.5.2 Mapping

Once the stimulation has begun, by clicking *RUN* in the main window, if neither *Weber* or *Discrimination* has been selected, the Mapping will start.

The order of stimuli by which the person is subjected to depends on the settings. Stimuli are prepared by linear steps, from the minimum to the maximum, and in case "*Randomize*" has been selected, they are shuffled. The shuffling is useful to remove the bias from the patient, while a non-shuffled (i.e.: simple linear increase) can be used to find thresholds for the minimum stimulus and maximum one. The protocol by which they are used will be explained in Chapter 7.

Each time the patient feels something the stimulation can be stopped (by clicking on "I feel something") and a predefined window allows to compile the felt sensations.


Stimulation-settings window
The chosen modulation is activated
For the selected modulation, a minimum, a maximum and the step are chosen.
The two other stimuli settings are set
Preview of the generated stimuli
Timing of the stimuli
Randomimzation of the stimuli order
Mapping GUI v0.1
File About
Mapping sensations
Device Settings
Sensations settings
Stimulation settings
Patient Info
Mapped
Conne
Pw show
Pw
step
Save
Load

Pulsewidth

Active

Minumum Pulsewidth [us]: 0

Max Pulsewidth [us]: 0

Step [us]: 0

Type of step: Linear

Amplitude [mA]: 0

Frequency [Hz]: 0

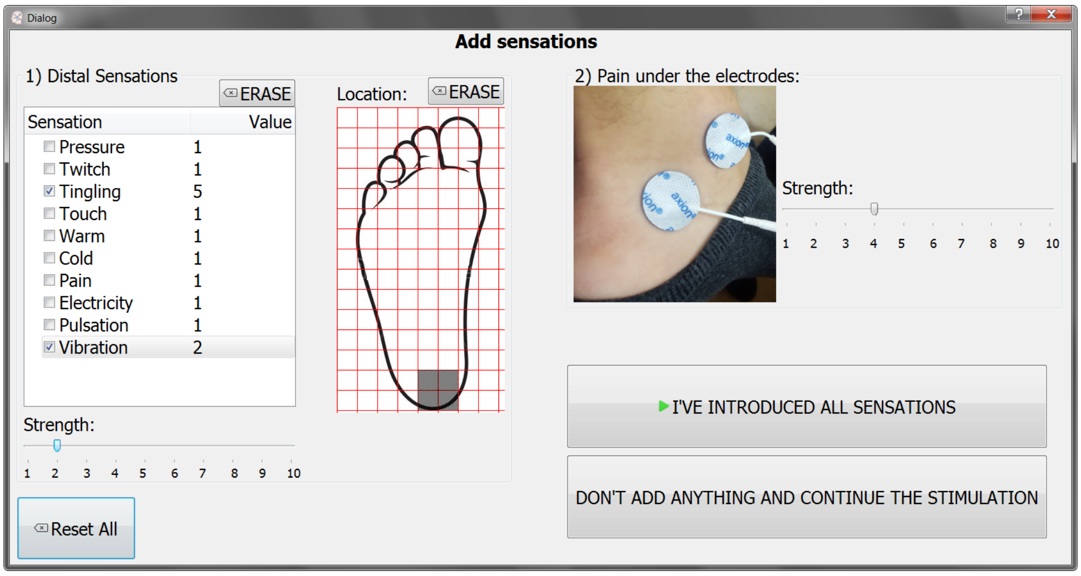
Randomize True

Time between changes [sec]: 0,000 This does not include the stimulation time.

Time of stimulation [sec]: 0,00

Test on Electrode: 0 All For each electrode, every A.S. will be stimulated

FIGURE 5.7: Mapping GUI: stimuli-settings.

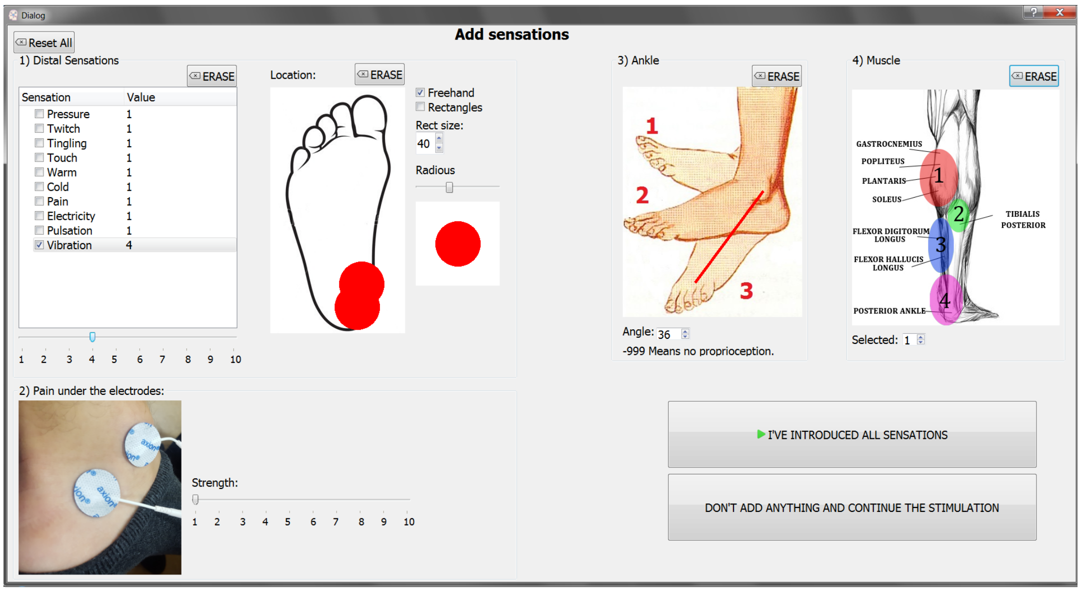

Dialog
Add sensations
ERASE
Location:
ERASE
2) Pain under the electrodes:
Strength:
1 2 3 4 5 6 7 8 9 10
I'VE INTRODUCED ALL SENSATIONS
DON'T ADD ANYTHING AND CONTINUE THE STIMULATION
Reset All

1) Distal Sensations

Sensation	Value
☐ Pressure	1
☐ Twitch	1
☑ Tingling	5
☐ Touch	1
☐ Warm	1
☐ Cold	1
☐ Pain	1
☐ Electricity	1
☐ Pulsation	1
☑ Vibration	2

Strength: 1 2 3 4 5 6 7 8 9 10

FIGURE 5.8: Mapping GUI: adding a new sensation for diabetics patients.



Sensation	Value
Pressure	1
Twitch	1
Tingling	1
Touch	1
Warm	1
Cold	1
Pain	1
Electricity	1
Pulsation	1
✓ Vibration	4

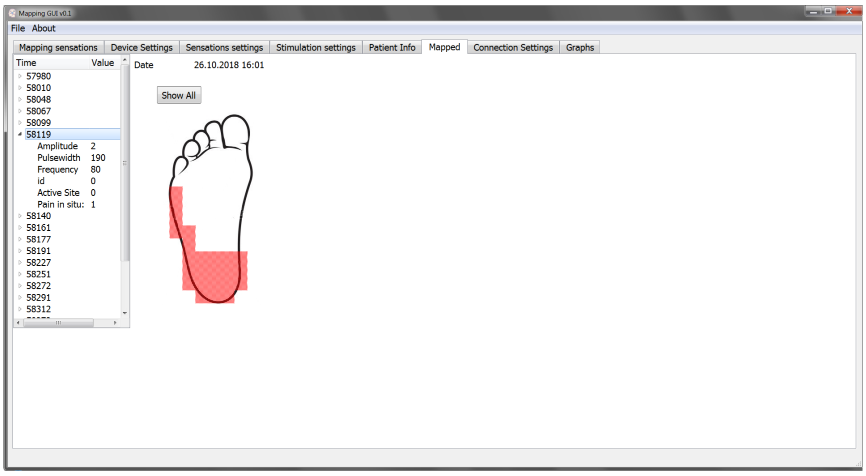
1 2 3 4 5 6 7 8 9 10

✓ Freehand
Rectangles
Rect size: 40
Radious

Angle: 36
-999 Means no proprioception.

Selected: 1

FIGURE 5.9: Mapping GUI: adding a new sensation for amputees.



Time	Value
57980	
58010	
58048	
58067	
58099	
58119	
Amplitude	2
Pulsewidth	190
Frequency	80
id	0
Active Site	0
Pain in situ:	1
58140	
58161	
58177	
58191	
58227	
58251	
58272	
58291	
58312	

Date 26.10.2018 16:01

FIGURE 5.10: Mapping GUI: Mapped window.

As stated before, there are two different mapping possibilities: one for diabetic patients and one for amputees, each one with its own "sensation"-window; they are shown in Figure 5.8 for diabetics, and Figure 5.9 for amputees.

In both cases he can choose the various feelings, expressing an intensity between 1 (absence) and 10 (maximum), then the pain over the electrodes. In the particular case of amputees he can also choose a contracting muscle and a proprioception feeling.

When all the stimuli in the range have been tested, the software will save the mapping, and automatically display it in the "Mapped" window. The mapping consist in the various associations between location of the electrodes, stimuli settings, and perception felt.

5.5.3 Mapped window

The *Mapped* window allows to load a saved Mapping showing the results, and after a *RUN* it automatically loads and display the mapping just finished. Each stimulus provided is shown, with the distal location expressed. An example is shown in Figure 5.10.

5.5.4 TENS connection

The TENS used has been Rehastim1, which will be presented in the following chapter.

It allows a serial communication over USB, with a defined protocol to control the stimulation called Science Mode. The stimulation is controlled one-stimulus-per-time by the Mapping Software, via the "CommThread".

The implementation allows a burst of a selectable frequency, for a selectable duration. To obtain a certain frequency and duration, timers are used; being Windows (as generally speaking, all the Desktop OSes) a non-real-time operative system, high frequencies might be unstable, depending on the system capabilities.

5.6 Conclusion

This Chapter has shown the developed Mapping Software. The result is a complete user-friendly interface, that at the same time provides an extensive control over the electrical stimulation. It can be decided to stimulate using pulsewidth modulation, frequency modulation, or amplitude modulation, even in a randomized way to eliminate bias from the Subject. The intervals during different stimulation and the duration of the stimulation itself can be imposed.

The system provides the possibility, at each point during the stimulation, to let the subject freely associate any felt sensation to the stimuli used. The sensation can be specified with an intensity from 1 to 10, and the distal elicited position is selectable graphically. These information, associated to the position of the electrodes, are essential to obtain a functional sensory-feedback neuroprosthesis.

Thus, the implemented platform provides a mandatory step in non-invasive electrical stimulation, essential for both research and characterization of evoked sensation, but also in the calibration of a neuroprosthesis.

Its functioning has been thoroughly tested with an oscilloscope. See Chapter 7 for validation of it on volunteers.

All the windows of the software are shown with higher details in Appendix B.

Chapter 6

Global system

6.1 Concept: closed-loop system

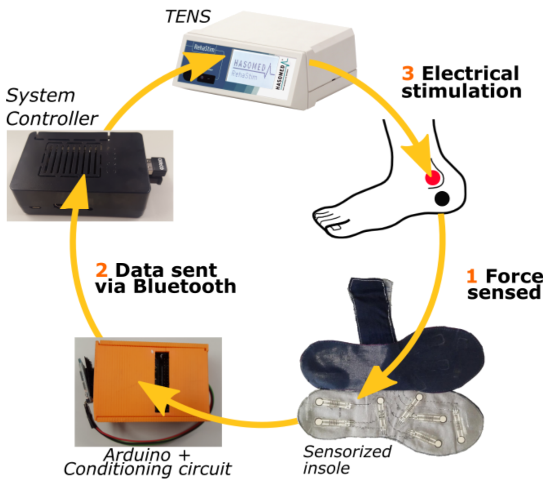


FIGURE 6.1: Concept of neuroprosthesis, example case of diabetic patient.

The final global system obtained is made by three main parts: the **electronic circuit** which samples with an interrupt every 30ms the force sensors, and sends the data via bluetooth to an **Odroid** (which has replaced the Raspberry, due to its higher speed). The software on the Odroid analyzes the data, and stimulate with a **TENS** accordingly.

In case a Client-GUI is remotely connected, the forces are sent over the network.

Figure 6.1 schematize further the concept. The loop symbolizes that the neuroprosthesis acts as a *closed-loop*: the force drives an electrical stimulation that will influence the Subject's action, and therefore the foot pressure itself.

Depending on the mapping results obtained on the Subject, each force sensor might be associated to different electrodes. This will be further discussed with examples in Chapter 7.

The Odroid[1] is a microcomputer that has replaced the Raspberry Pi 3, it still runs a Linux-based OS thus the server-code was not modified in order to be compiled and run. This new setup allows to have a better performance, which is translated to an higher software speed; thus maximum-frequency of data capability.

[1]Odroid C2

FIGURE 6.2: Odroid microcomputer inside its case.

6.2 Delays evaluations

As the aim of the project is to provide a sensory feedback, the system delay must be known, or at least to be estimated under a defined threshold. A delay of 100ms is considered to be the maximum allowable overall time in this field.

Various parts are critical:

1. The time that it takes to send the data read by Arduino over the bluetooth, and to be received by the software running on the Odroid.
2. The time it takes once the data has been received, to be used to change the stimulation parameters.
3. The time that it takes to send a stimulation configuration to RehaStim and to have the actual stimulation.
4. The time the software takes for data post-processing.

A graphical scheme of the delays is shown on Figure 6.3, where the numbers in red corresponds to the numbers of the above list.

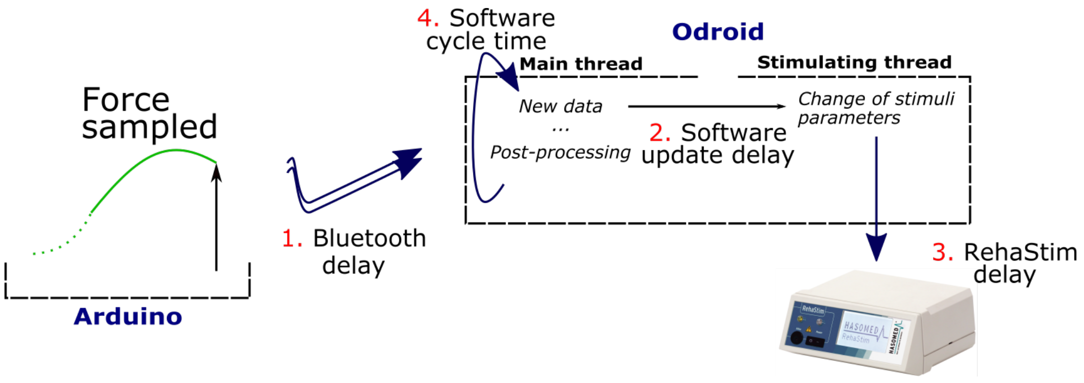


FIGURE 6.3: Scheme of the various delays.

The obtained overall delay is under 30ms; the following sections will explain the procedure used to extrapolate this value.

For consistency also the network delay between the Odroid software and the Client-GUI has been evaluated.

6.2.1 Delay evaluation for the Bluetooth transmission

To evaluate at which degree the Bluetooth transmission affects the delay, the following protocol was implemented:

1. The Odroid directly powered an Arduino board, which had a Bluetooth module connected.
2. A logical output of the Arduino was connected to a GPIO[2] pin of the Odroid.
3. Once the bluetooth connection was initialized, the Arduino board waited for a trigger sent by the Odroid.
4. The Odroid sent the trigger, then started a timer.
5. Once the trigger was received by the Arduino, the device sent a full package as if it were an insole read.
6. After a full data read was accomplished by the Odroid, the timer was stopped and its value printed.
7. This process was repeated for various seconds.
8. The average was calculated.

The obtained values are:

$$\text{Average Bluetooth delay} : 13.1ms \pm 8.5ms$$

6.2.2 Delay to update the stimuli with the new data

The reading of the bluetooth packets and the electrical stimulation work on different Threads. More information regarding the Software architecture can be found on Chapter 3.

Hence, once the new insole data has been received, this information has to be "forwarded" from the Bluetooth Thread to the Stimulation thread. To assure a correct evaluation of the overall delay of the neuralfeedback, this has been measured with software timers, obtaining an average less than 0.01ms, which is thus disregarded.

6.2.3 Delay evaluation for TENS

The time that it takes to send the stimulus settings from the Odroid to RehaStim, and for the stimulus to be actually done, has been estimated to be:

$$\text{Average TENS delay} : 3.6ms \pm 0.2ms$$

This is the delay between Odroid/RehaStim

To obtain this value, the following procedure was implemented:

1. The RehaStim output was connected to the electrical circuit shown in Figure 6.4.

[2]General Purpose Input/Output.

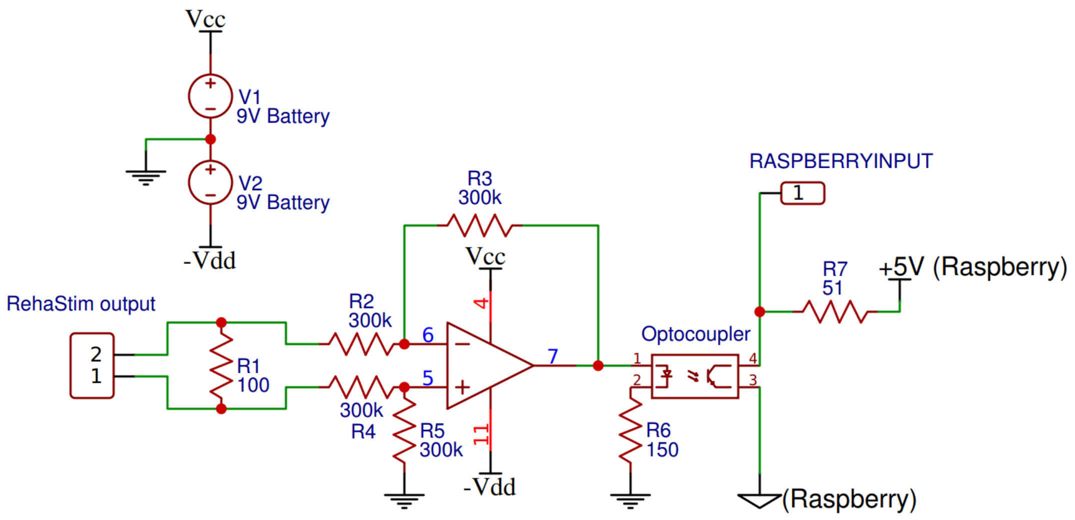


FIGURE 6.4: Electronic circuit used to evaluate delay.

2. The connection between Odroid and RehaStim was initialized.
3. A packet with a pre-defined stimulus was prepared.
4. An ad-hoc software made for the Odroid sent the packet, and started a timer.
5. A GPIO[3] port on the Odroid (connected as shown in the schematic) was read until the logical value changed.
6. Once the value changed, the timer was stopped and its value printed.
7. This process is repeated automatically for various seconds.
8. The average is calculated.

The electronic circuit is used to first obtain a differential signal from the output of RehaStim. It requires dual voltage supply (the pulse is biphasic) and this is obtained by two 9V batteries connected as shown.

Then, to read the value by the Odroid, an optocoupler is used to pass from one reference (the 9V-9V middle reference) to the the Odroid supplier one.

6.2.4 Software speed evaluation

The speed of the software-reading part has been evaluated indirectly: the sampling frequency on the Arduino (and thus, the frequency at which data is being sent) has been increased, and the number of dropped packets measured. The following results have been obtained:

This evaluation obtains also the usable sampling period; the chosen one is 30ms, to ensure no packet loss.

The evaluation is possible because the software, before waiting for a new packet, flushes the input data-stream and saves how many packets it has flushed. This is necessary in order to avoid an increasing lag.

[3]General Purposes Input Output

Sampling Period	**Dropped packets**
10ms	100%
14ms	54.56%
15ms	88.39%
16ms	47.95%
17ms	5.99%
18ms	1.44%
20ms	0.28%
25ms	0.17%
30ms	0%

6.2.5 Client-GUI network delay

To justify that the software is Real-Time two assumptions have to be made:

1. The software must not accumulate lag.
2. The time between the generation of new data (on Odroid) to the reception by the Client should be acceptable.

The network communication is discussed in Chapter 3; the fundamental concept is that data is sent via UDP packets, and the physical layer can be for example WiFi, or Ethernet cables, or local packet forwarding on the same system. Depending on such configuration, this delay time will change. The worst case scenario is to use WiFi connection, which is the slowest one and for this reason the one evaluated.

The first consideration is confirmed by implementing a flushing of the UDP buffer[4] prior starting waiting for a new packet. If no packet is flushed, it means that the Client is always ready to receive a new data, and thus it does not add an increasing lag. If packets are flushed, it means that the software is disregarding old data that was not fast enough to read to receive; thus not accumulating lag.

The number of flushed packet for a typical session are less than 0.3%, indicating that the system works correctly.

For the second problem, an evaluation has been made using the *Wireshark* software to analyze network packets. Using the exact same Client and server software, the following modification has been added: the server, before sending a packet, awaits for a request by the Client.

Then measuring the time difference between the *request packet* sent by the Client, and the *data packet* answered, an evaluation on the UDP network latency can be made, obtaining:

$$\text{Average Client-GUI network delay} : 1.0ms \pm 0.6ms$$

This is an acceptable value, non perceivable by a user.

[4]The OS implements a buffer for the networking: received packets but not read fills such buffer.

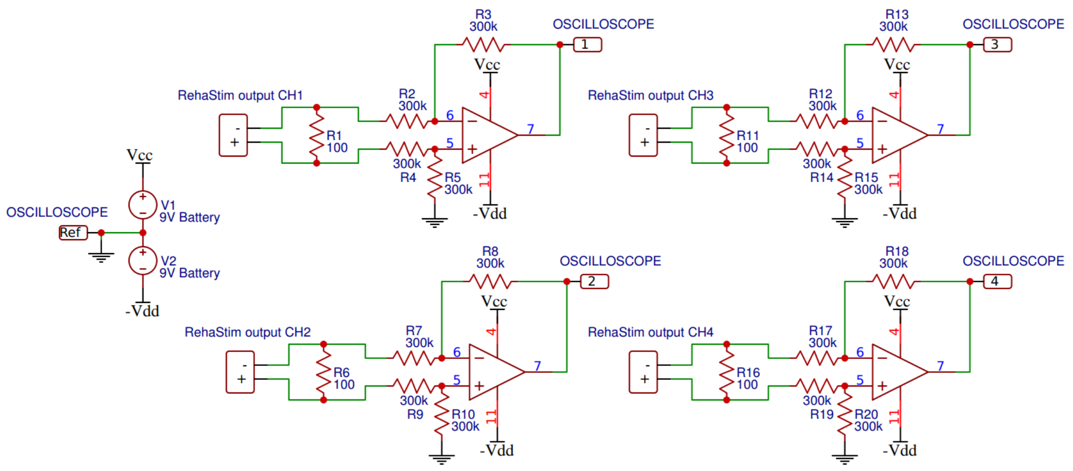


FIGURE 6.5: Electronic circuit used to test the system with an oscilloscope.

6.2.6 Preliminary test

To evaluate the correct functioning of the system, the implementation has been tested with an oscilloscope, and then with a sane subject walking. In the Appendix A some images of the tests can be found, with annex links to the complete videos.

The oscilloscope in the laboratory had up to four different channels displayed concurrently, for this reason four different RehaStim channels for stimulation have been used: four force sensors drove each a different pulsewidth modulation.

To be able to display the values on the oscilloscope, the circuit in Figure 6.5 has been used.

This is a necessity, because the reference for the oscilloscope has to be the same for each channel.

6.3 Drive of the stimuli: transfer function

Once the force is sampled, a system controller (Odroid) uses that value to drive an electrical stimulation that elicits in the patient a distal-sensation, that ideally should be in proximity to that sensor.

The more the "stimulation", the more the felt intensity of the artificial sensation. The intensity of the stimulation can be expressed in terms of Current amplitude, Frequency, or Pulsewidth of the pulses.

Pulsewidth-modulation has been chosen as the way to encode the intensity of the feedback: as it will be further discussed in Chapter 7, frequency modulation is hard to discriminate, meaning for example that stimuli from 80Hz to 300Hz are felt to be the same one. Modulating Current amplitude, instead, achieves discomfort in the patient after only 3 steps from the minimum value.

Pulsewidth modulation provides a stimulation that is discriminable and has a large swing before creating too much discomfort. For these reasons, is the method used to encode information, and to artificially induce a sensory-feedback.

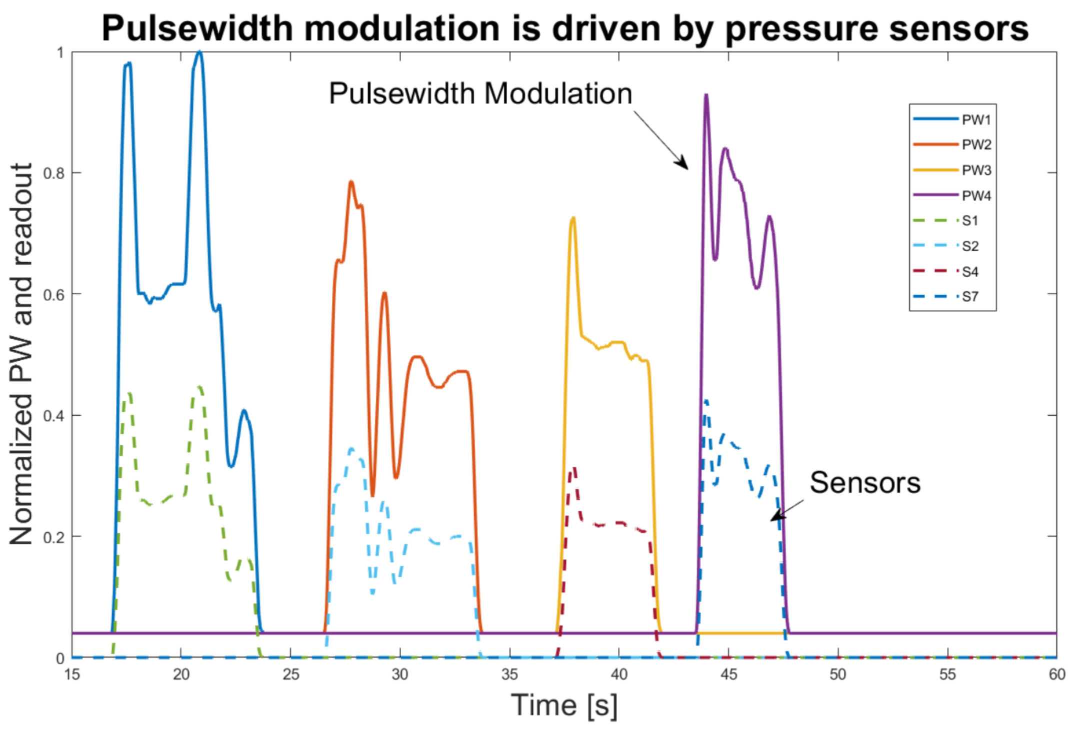


FIGURE 6.6: Data saved by the GUI shows the modulation of various channel applying forces over the sensors. This, as the case shown in the Appendix A, keeps the minimum pulsewidth at 20us. It was done to more easily visualize the real-time modulation on the oscilloscope, for a multi-channel modulation.

Thus, the force sampled should drive the pulsewidth modulation, and this is achieved with the following transfer function:

For $Th_{Fr} \geq Fr_x \leq Fr_{max}$

$$PW_x[\mu s] = PW_{min} + \frac{Fr_x - Th_{Fr}}{Fr_{max} - Th_{Fr}}(PW_{max} - PW_{min}) \tag{6.1}$$

For $Fr_x < Th_{Fr}$

$$PW_x[\mu s] = 0 \tag{6.2}$$

For $Fr_x > Fr_{max}$

$$PW_x[\mu s] = PW_{max} \tag{6.3}$$

where:

- PW_x is the pulsewidth of the stimuli for the x couple of electrodes, associated with the x force sensor.
- PW_{min} is the pulsewidth at which the Subject starts feeling something distal.
- Fr_x is the force read and sampled for the x sensor.
- Th_{Fr} is the force threshold to overcome to start the stimulation.
- Fr_{max} is the maximum force sensed during a subject natural stride.

- PW_{max} is the maximum pulsewidth that the Subject can accept before feeling too much discomfort.

If the force is less than a minimum threshold, the stimulation is not activated, meaning that the pulsewidth is 0. This is useful to avoid stimulating permanently the subject and also the possible noise present in the data.

If the force is between a minimum and a maximum (i.e.: the force swing range that the Subject generate during a natural gait) then a linear mapping between the sensed value and the Pulsewidth is applied. The Pulsewidth increases from the minimum to elicit a distal-sensation, up to the maximum that is still acceptable by the Subject.

After this maximum value, the exerted stimulation saturates. A visual representation is shown in Figure 6.7.

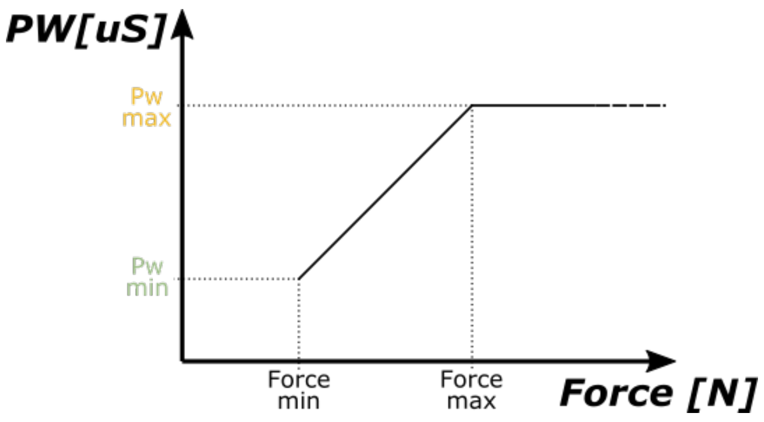


FIGURE 6.7: Transfer function between force sensed and pulsewidth exerted.

The maximum and minimum pulsewidth will be discussed in the next chapter, but it should be stated that they depend on the electrodes positions and thus each output channel should have its calibrated minimum/maximum.

Chapter 7

Preliminary validation of the platform

7.1 Introduction

The Mapping software has been tested on some volunteers: in the following subsection the various results will be presented and properly discussed.

The subjects were stimulated on the ankle with the aim to evoke distal sensations on the foot; despite them being sane, this still provides a first step into the characterization of elicited sensation by ankle-transcutaneous stimulation. To this student knowledge, no literature has yet explored this concept.

7.1.1 The basis of eliciting distal sensations by ankle stimulation

The first step has been finding a general proper zone to stimulate. To elicit sensations on the foot, the nerves to stimulate are the various branches of the *Tibial Nerve*:

- *Lateral Plantar* nerve.
- *Medial plantar* nerve.
- *Medial Calcaneal* branches.

The Figure 7.1 [35] displays the cutaneous innervation of the foot, where: **e** is the Medial plantar nerve, **f** is the Lateral plantar nerve, and **g** is the Medial Calcaneal Nerve.

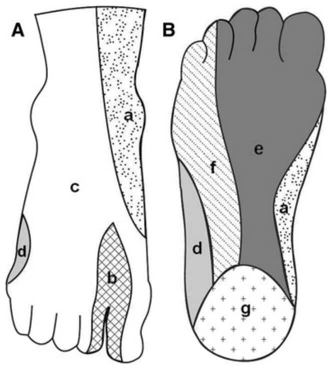


FIGURE 7.1: Cutaneous innervation of the foot, e is the Medial Plantar Nerve; f is the Lateral Plantar Nerve; g is the Medial Calcaneal Nerve.

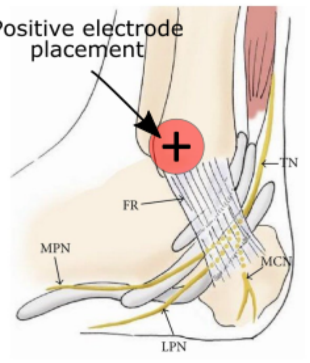


FIGURE 7.2: Branches of the Tibial Nerve (TN) near the medial malleolus [13]: Medial Calcaneal Nerve (MCN); Medial Plantar Nerves (MPN); Lateral Plantar Nerves (LPN).

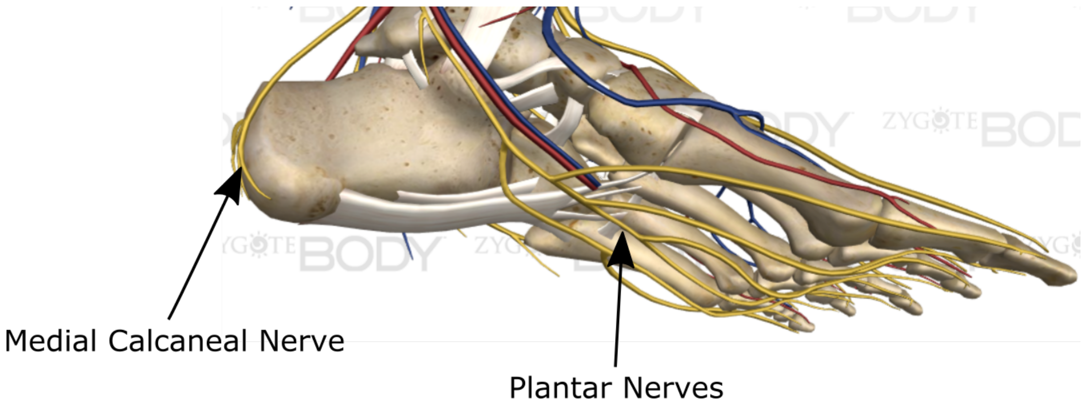


FIGURE 7.3: 3D Representation of Tibial Branches in the foot.

The positive electrode[1] was thus placed on the *medial malleolus*, while the negative has been moved over the foot to draw a first broad range of "useful positions".

An *useful position* was identified as one that elicited distal sensations and did not generate excessive in-situ[2] pain. On image 7.2 and 7.3 it can be seen an approximate position of the nerves that are underneath the tested positions.

Figure 7.4 shows the results of this step. The green zones are the ones identified as *possible* to use: they provide a distal sensation and the pain is relative low. They'll be candidates to be used in the actual mapping part with subjects. The red zones were either ineffective in obtaining a sensation, or as in most cases, they created too much discomfort even with the lowest stimulation intensity. As a comparison, the lowest intensity is usually not perceived in the green-zones, and a similar pain is usually not reached in the whole stimulation range used.

The stimuli settings used were:

Setting	**Value**
Frequency	80Hz
Current Amplitude	2mA
Pulsewidth	[20us to 300us] with steps of 10us

[1]As discussed in Chapter 4, circular electrodes with a diameter of 2cm were used.
[2]That is, under the electrodes

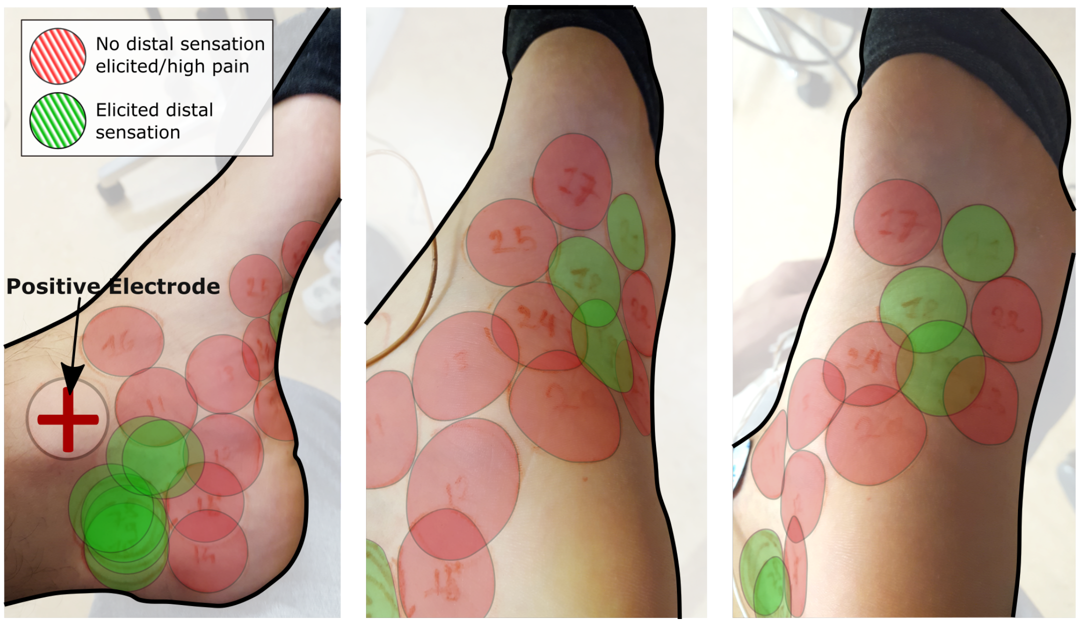

No distal sensation elicited/high pain
Elicited distal sensation
Positive Electrode

FIGURE 7.4: First step: broad evaluation of useful electrodes positions.

further details about the stimuli modulation and their effects are expressed in the following subsection.

During some first experiments it was discovered that moving the ankle resulted in changing, some times largely, the location of the sensation. Also the intensity undergoes a noticeable variation: this is probably justified by the relative movement of the fibers to respect the electrodes. For this reason, as it will be explained in the "*Protocol*" subsection, the evaluations of the mapping have been made for different ankle positions, and the results have been compared to evaluate the degree at which this might affect a neuroprosthesis.

7.1.2 Stimuli settings and their effects

As previously explained, the stimulator used, RehaStim, provided the possibility to change either the *Current Amplitude* or the *Pulsewidth* of the stimuli[3]. As a third variable, a burst frequency was implemented via software.

The final aim is to map a force value to a stimulation parameter, ideally evoking more sensation as the force/stimuli increases. Choosing what variable to use for the mapping, meaning, what variable will be modulated while keeping the other fixed is, in this case, quite easy. Firstly, the frequency modulation is insufficiently discriminable, reducing the possible swing of it, and thus, its usefulness. Secondly, amplitudes higher than 2mA quickly achieve pain or discomfort in the subjects. Modulation of the stimuli by the pulsewidth is instead a feasible way to control the intensity of the evoked sensation: it's discriminable and provides a range in which pain is not felt, assuming that a correct placement of the electrodes has been made.

[3]More information can be read on Chapter 4

For these reasons, during the mapping 2mA will always be used for the Current Amplitude. Higher values would also reduce the swing of the pulsewidth, obtaining discomfort in the subject quickly.

The frequency chosen for all the mapping tests has been 80Hz. Lower Frequencies evoked more unnatural sensations, up to being described as a "rhythmical knock". Higher frequencies are indistinguishable, thus avoided to reduce the timing requirement of the software.

7.2 Protocol

After founding an initial location-reference, as seen in Image 7.4, four positions were chosen and used to map and characterize the sensations; these are displayed in Figure 7.5 and were found on Subject 1. For the others Subjects, position that were decided to be similar to these were used to map, but, due to high variations between people, position adjustment had to be taken to avoid pain.

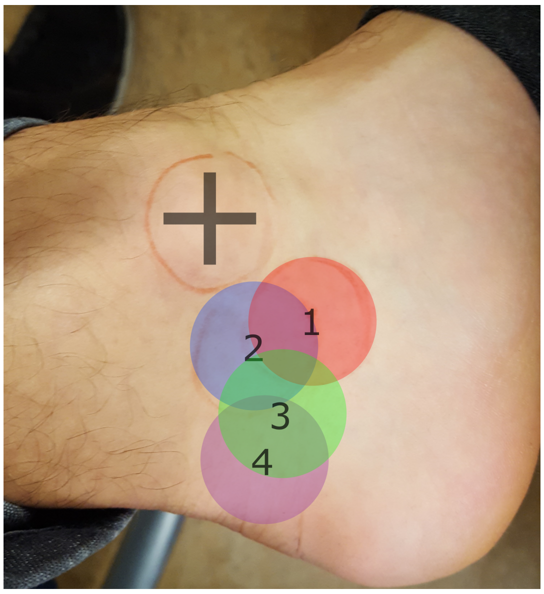


FIGURE 7.5: Chosen position to map, from Subject 1.

The protocol used to test the Mapping software on different subject is the follow.

The subject is required to wash his left feet, and if required, hair were clipped or shaved. Albeit the interested locations are way less affected by hair than others, the position on the medial malleolus might be partially covered. The positive electrode is placed on the malleolus while the negative is placed on one of the proposed four positions as seen in Image 7.5.

Once the subject is ready, a first initial low stimulation is done. This has the aim to make the subject aware on what's the feeling under the electrodes, and make him relaxed about the experiment after that the actual trial begins.

A linear increase in Pulsewidth modulation is used to stimulate the subject. The Frequency used will always be 80Hz, and the Current Amplitude 2mA, while instead the Pulsewidth starts from 20us (the minimum possible) and is increased by a step of

10us every 3 seconds. The stimulus lasts for 1 second, this means that for 2 seconds no stimulation will be done.

If the subject reports pain or discomfort before reaching 100us, the position is adjusted until this ceases.

Once a suitable position is found, and this usually takes only one trial, having already the starting map seen in Figure 7.4, the stimulation is made once again increase from 20us with steps of 10us. The subject is required to tell the experimenter whenever he feels something distal (i.e., a sensation not on the electrodes); this value will be used as lower limit for the mapping. The subject is also asked to report whenever he feels that the stimulation is too high, either that be simple discomfort due to muscle contraction or pain. This will be used as maximum in the mapping part (actually, using 20us less than this value). If no discomfort is felt up to 400us, then 400us is used as maximum stimulation.

After this initial preparation, the subject is trained about the "Adding sensation" window (shown again in Figure 7.6). It's explained to him that:

- He will have to report any distal-felt sensation, with a value from 1 (not felt) to 10 (maximum sensation).
- He's made read the list of possible distal sensations.
- He will have to choose where he felt the distal sensation.
- He will have to choose the pain/discomfort/sensation under the electrodes, from 1 (not felt) to 10 (maximum).

Then the mapping part starts: this time the stimuli will not be increasing, rather a random pulsewidth modulation (between the minimum and maximum, divided by steps of either 10us or 20us) will be used. This is done to avoid biasing the subject, which would know that the stimulus is increasing. The subject is also informed about the randomization order of the stimuli. Every time a new stimulus occurs, he's asked to compile the "Adding sensation" window.

As before, stimuli occurs every 3 seconds (the Adding Window pauses this process) and last this time for 2 seconds.

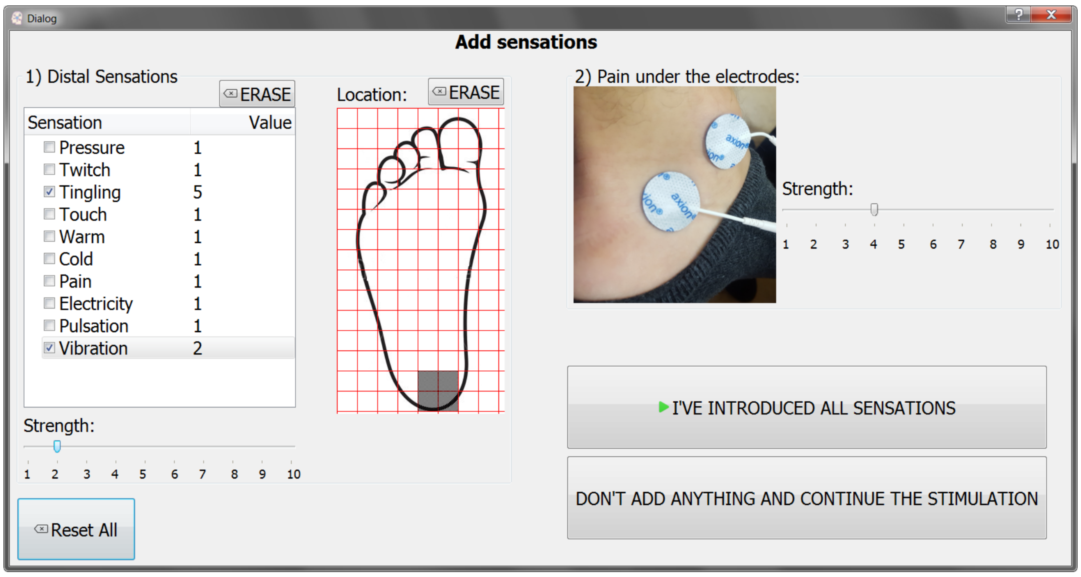
Once this is done for every step in the range, the stimulation is stopped and the choices are saved with their associated stimuli.

As stated before, the result depends on the ankle position. For this reason, this mapping is repeated for the following ankle positions:

- Lifted Foot.
- Heel-strike.
- Push-off.
- Stance.

7.2.1 Discrimination test

A subsequent test done is the **Discrimination test**. The software will alternate a reference stimulus (the middle point between the minimum and the maximum Pulsewidth



Sensation	Value
☐ Pressure	1
☐ Twitch	1
☑ Tingling	5
☐ Touch	1
☐ Warm	1
☐ Cold	1
☐ Pain	1
☐ Electricity	1
☐ Pulsation	1
☑ Vibration	2

FIGURE 7.6: Mapping GUI: "Add a new sensation" window used for the experiments.

is used) and a random stimulus (in the range). The subject is then asked if the two distal-felt sensations were:

- The second of Higher intensity
- Equal/Uncertain
- The second of Lower intensity

The two stimuli lasts 1 second each, and are intermediated by 100ms.

The subject is instructed before starting of the possible choices.

7.3 Mapping Results

Figure 7.7 shows the intensity perceived by different subjects, for different ankle positions, as the modulation increases. As stated, the values are extracted from randomized trials to avoid bias. Figure 7.8 displays the same information for the perceived pain in-situ. Both subjects are able to correctly identify an increase in the distal-intensity as the pulsewidth is increased. Interestingly, for a similar electrode-position and ankle angle, subject 1 and subject 2 reported similar intensities. While instead, with *stance*, Subject 2 perceived an higher intensity for all the pulsewidth modulation.

These first images clarify that the concept of driving a perception with pulsewidth modulation is possible.

The complete Mapping results for **Subject 1** is shown in Figure 7.9. As it can be seen the distal-location depends on both the electrode position and the ankle position.

Albeit some mapping results might seem similar (in terms of location), the comparisons in Figure 7.10 show how the perception is highly different. Position 3 has a reduced pulsewidth range, due to a painful distal sensation limiting the possible range. Position 4 shows a lower swing in intensity perception.

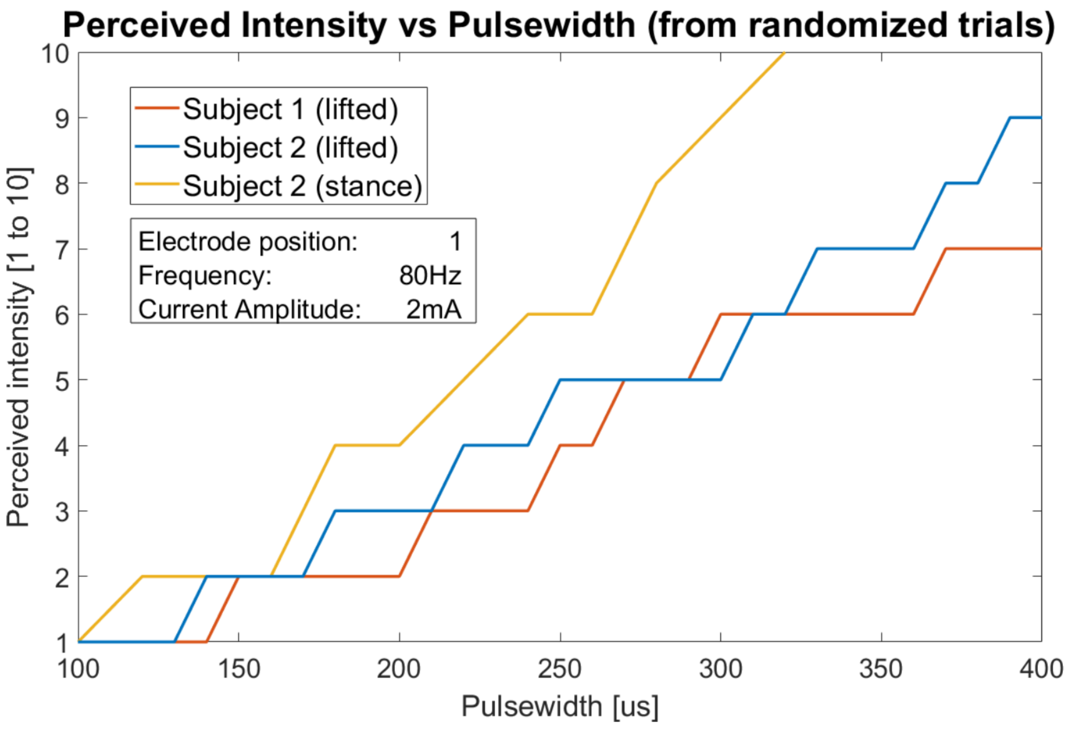


FIGURE 7.7: Perceived distal-intensity increasing the stimulation.

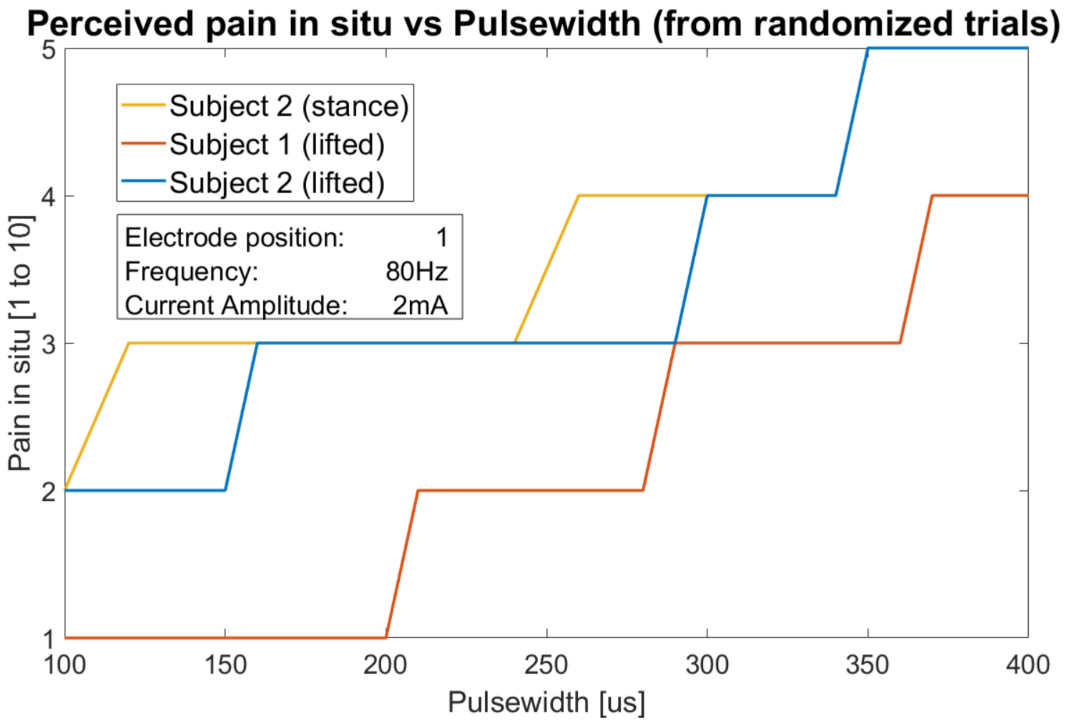


FIGURE 7.8: Perceived in-situ pain increasing the stimulation.

Figure 7.11 makes the same comparison evaluating in-situ pain. Even if Position 2 has a steeper increase after 320us of pulsewidth, it is still the preferable position, because in the most probable range of use (i.e.: far from the maximum) it is associated with an absence of in-situ pain.

The effects of increasing the stimuli over the distal-location can be seen on Figure

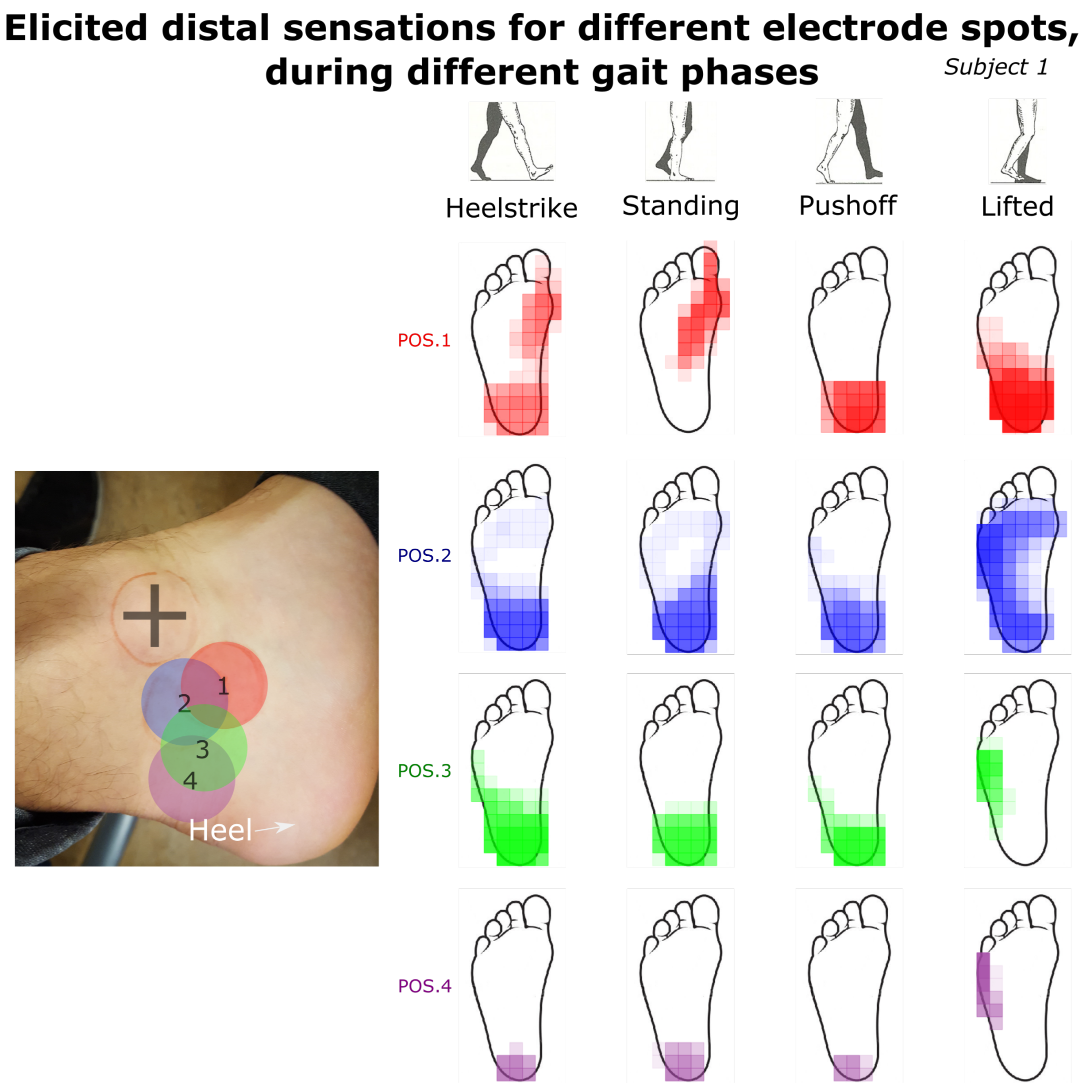


FIGURE 7.9: Complete mapping for Subject 1. The distal-location is affected by the ankle position and by the electrode placement.

7.12, the results indicate that the distal-location increases in size, but not to the point of reducing the usefulness of the system.

Subjects were also tested by inverting the electrodes polarity, while keeping the same electrodes placement and settings used to stimulate. The mapping results are compared in Figure 7.13. It's clear that the distal-location is not affected by electrode polarity.

Repeatability of the elicited distal-location can be seen on Figure 7.14 and Figure 7.15. As it can be seen, in these cases the results are highly repeatable.

Inter-subject comparison, for similar electrode placement is presented in Figure 7.16. To reduce variability the same ankle position are used in the comparison. The result shows that the distal-location is highly patient dependent.

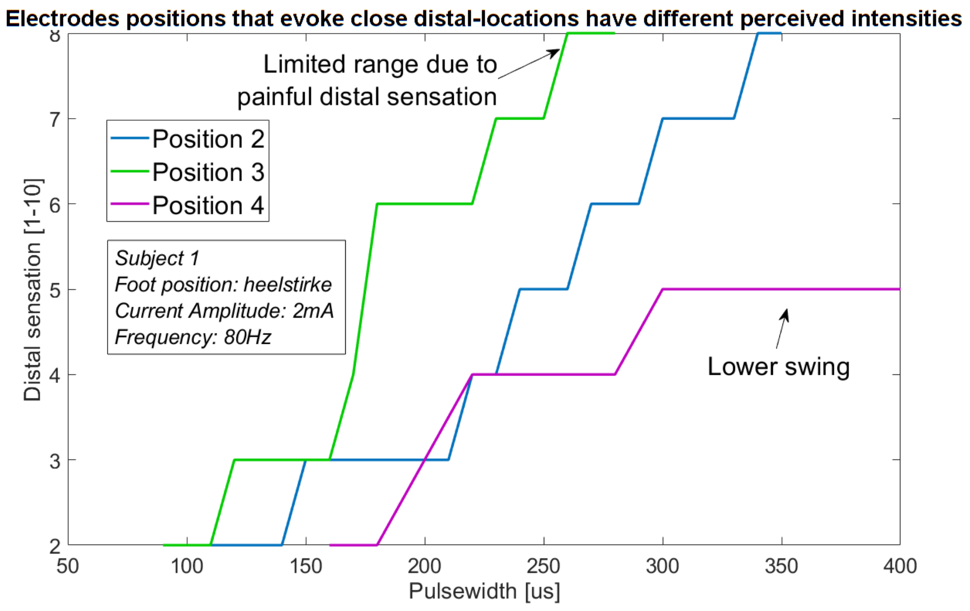


FIGURE 7.10: Perceived distal intensity for three different electrodes positions that evoke close distal-sensations. The plot is color-coded with positions shown in Fig. 7.5.

7.3.1 Discrimination results

The discrimination test evaluates what's the minimum increase (**JND**[4]) of pulsewidth from a reference, by which the subject can correctly identify a variation in the distal-sensation intensity.

By this absolute value, the Weber fraction can be calculated:

$$K = \frac{JND}{Reference}$$

for the same type of stimulation and same electrodes position this should be approximately a constant value.

This is an indicator of the feasibility of this system to provide correctly a sensory feedback, in a functional and useful way [20] indeed, if for example K were to be 10, then a stimulation with a pulsewidth of 200us would require 2000us more to elicit *the minimum* perceivable difference; it's clear that this would disrupt the efficiency and usefulness of the system.

The obtained values were, for pulsewidth modulation:

Subject 1, pos.4	**k**
Mode A	0.10 ± 0.01
Mode B	0.08 ± 0.01

Subject 1, pos.3	**k**
Mode A	0.10 ± 0.02

[4]Just Noticeable Difference.

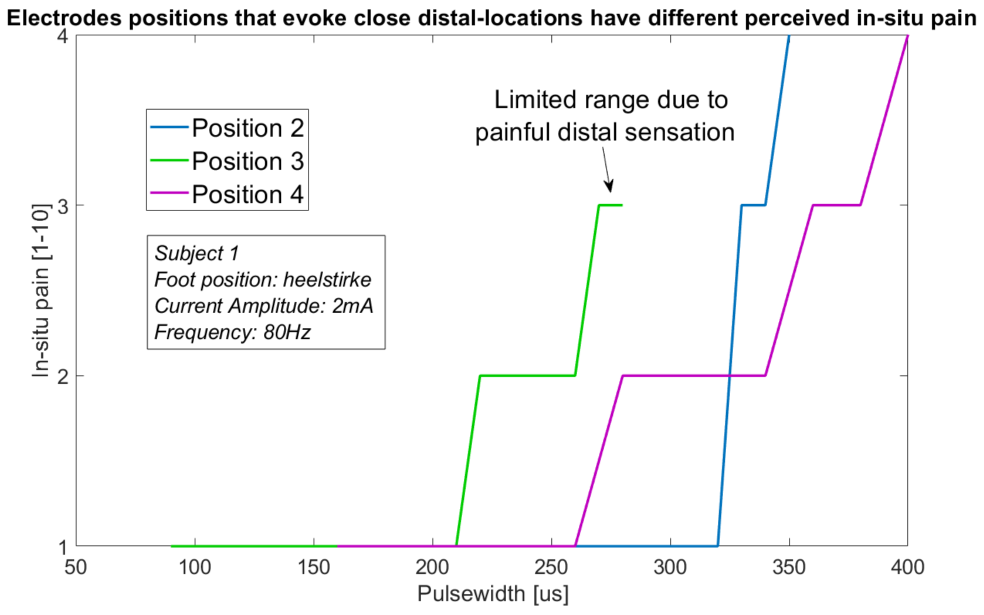


FIGURE 7.11: Perceived in-situ pain for three different electrodes positions that evoke close distal-sensations. The plot is color-coded with positions shown in Fig. 7.5.

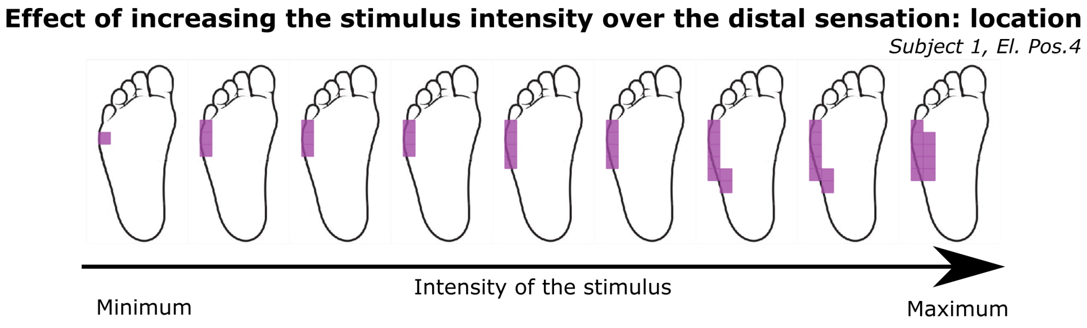

Effect of increasing the stimulus intensity over the distal sensation: location
Subject 1, El. Pos.4
Intensity of the stimulus
Minimum
Maximum

FIGURE 7.12: Effect over the distal-location evoked, increasing the pulsewidth.

7.4 Example of application

Using the results from Subject 1, an example of application can be provided. Assuming the possibility to have access to a portable TENS, then the force sensors positioned over the heels could be directly linked to the modulation of stimuli over position 2, which has been extracted as the position that allows the best dynamic range, both in terms of perception and in terms of pulsewidth modulation, keeping an acceptable in-situ pain. While standing, the stimulation could be done *also* on position 1, linking it's value to a force sensor under the metatarsal bones.

7.4.1 Conclusions

These preliminary results have been obtained with only four healthy subjects. The electrodes placement is the one suggested for Diabetics, and clearly for amputees

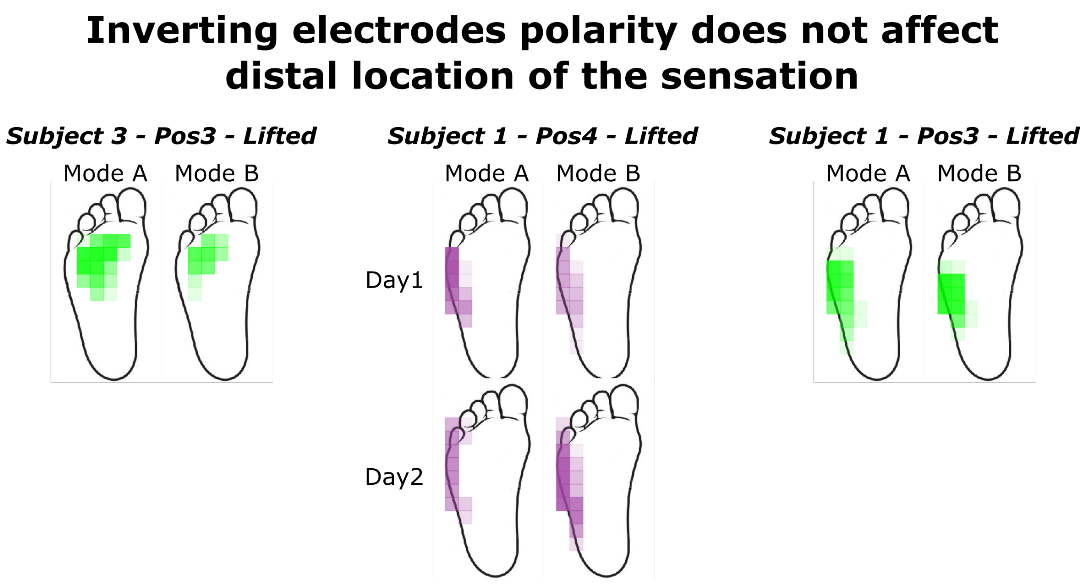


FIGURE 7.13: Comparison of mapping results inverting the electrodes polarity. Mode A: positive electrode on the medial malleolus; Mode B: negative electrode on the medial malleolus.

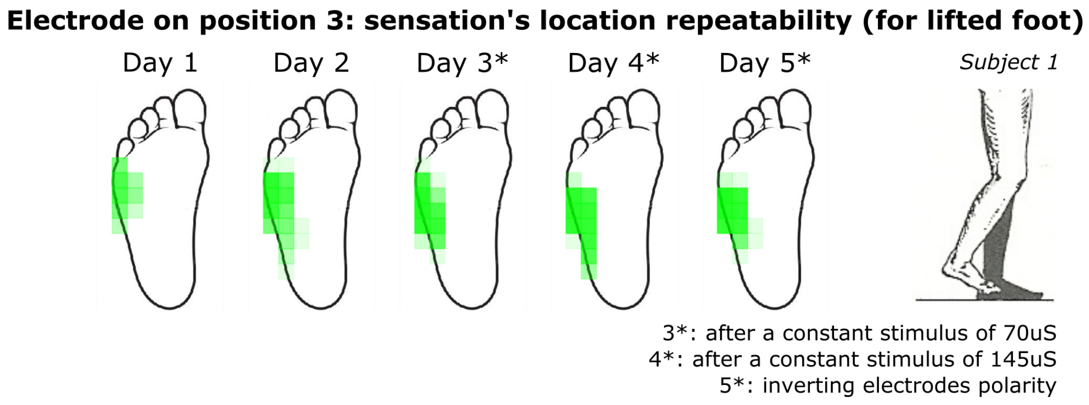


FIGURE 7.14: Comparison of the mapping results during different days, for the same electrodes positions (position 3 of the Subject 1). Day 3 and Day 4 also shows the results after prolonged stimuli. Day 5 shows the location affected after inverting electrodes polarity.

(unless they are partial foot amputees) the position should be carefully chosen. Nonetheless, the results have proven the implementation to be working, and most importantly the feasibility of stimulation of the Tibial Nerve via non-invasive electrical stimulation, evoking somatotopic sensations.

The various different location evoked in different patients, suggest that using smaller electrodes might result in the possibility to achieve a more accurate stimulation over the entire foot for every subject.

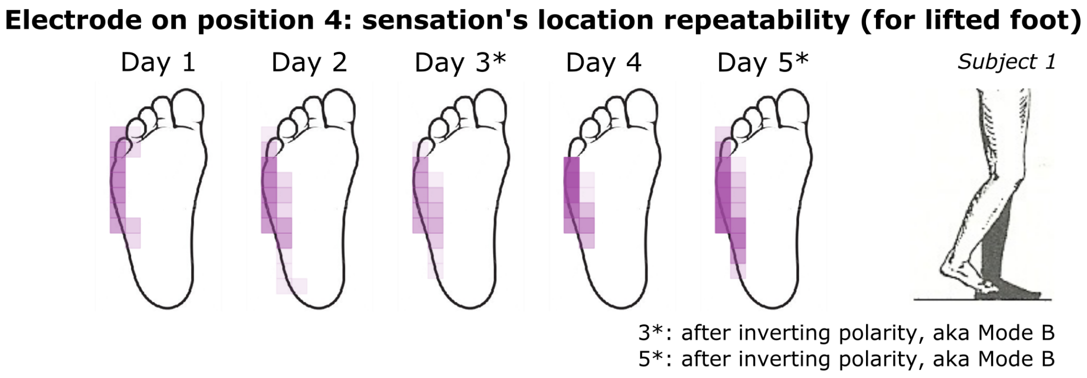

Electrode on position 4: sensation's location repeatability (for lifted foot)
Day 1
Day 2
Day 3*
Day 4
Day 5*
Subject 1
3*: after inverting polarity, aka Mode B
5*: after inverting polarity, aka Mode B

FIGURE 7.15: Comparison of the mapping results during different days, for the same electrodes positions (position 4 of the Subject 1). Day 3 and Day 5 also shows the results inverting electrodes polarity.

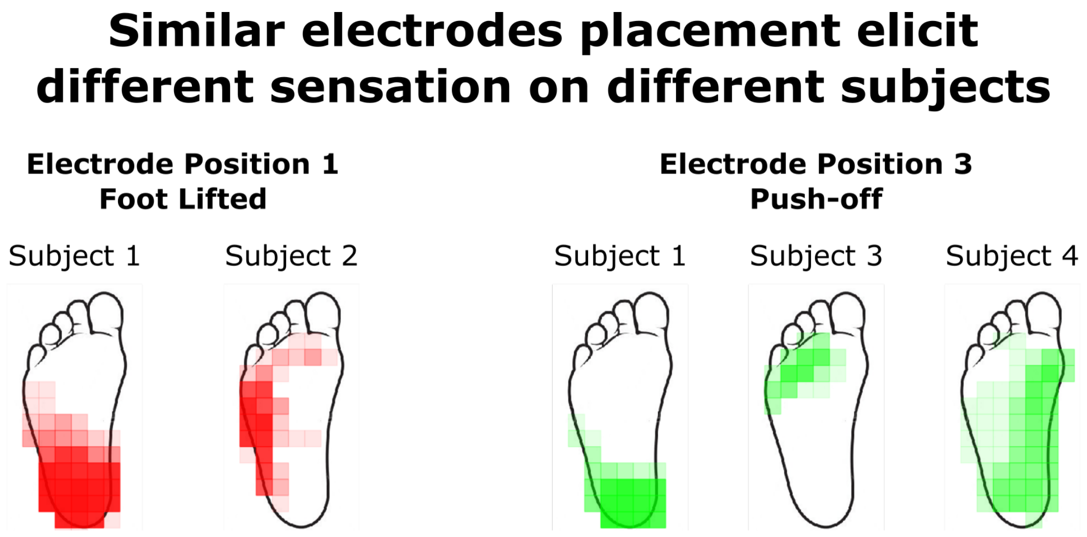


FIGURE 7.16: Comparison of mapping results for different subject, using similar electrodes positions. The distal-location differs from subject-to-subject.

Chapter 8

Discussion and future developments

In this Chapter the main parts of this thesis work will be evaluated, providing some insights in the current limitations, as well as future possible developments.

8.1 Real-Time Client GUI

This thesis has first presented a **Real-Time software** to display insole sensed data.

Particular attention was posed to optimize the software code and the network protocol, to overcome many problems that would otherwise create an increasing and cumulative lag between the actual data and the plot. This required using UDP for network communications, implementing flushing operations, optimizing the data plot using hardware acceleration and most importantly creating a multi-thread solutions.

The implementation of a multi-thread software complicates the software architecture heavily: different threads must *signal* events to each others, and the access to global variables must be regulated through the use of *Mutex*. Nonetheless, this allowed to obtain a Real-Time solution.

The software has been developed with C++ and Qt libraries, achieving the possibility to run the software on many platforms, such as Windows, Linux, iOS and Android.

The network delay has been evaluated to be around 1ms, thus not perceivable by the user; counting how many old UDP packets have been flushed, reported that the Client-GUI is fast enough to process data faster than the sampling period.

As an additional feature, a weight-balance has been added to the GUI, with the aim to increase gait symmetry in amputees. The method implements a negative audio feedback controlled independently for each force channel. The technological implementation of the algorithm has been tested and it's working correctly (simulating wrong steps), but no test on amputees has yet been made, and it's heavily required to evaluate its actual functionality.

As a way to improve the algorithm, first derivatives could be used as a threshold to better identify real gait from other false-positives. More advanced technique cold also implement a machine-learning approach to characterize the identification of a wrong stride from a correct one. Another improvement could be to use a different sensorized insole, placed in the shoe of the healthy limb, and to formulate an algorithm to extrapolate from the force data a symmetry evaluation.

8.2 Mapping Software

To associate different stimuli and electrodes position to the elicited sensation, a **Mapping Software** has been designed and created from scratch, and eventually validated on healthy volunteers.

Just like the Client-GUI and the server running on the system-controller, the Mapping software is also multithread. This was required as a security mechanism towards possible failures of parts of the system. This way the code for the stimulation part has only that aim, and it's less prone to have problems. The Software is developed using C++ assuring high performance and stability. Two slightly different versions for mapping the sensations for amputees and diabetics have been developed.

Not only the software achieves a useful part in obtaining a functional closed-loop system, but the highly intrinsic anatomical variability between subjects leads to the absolute necessity of such implementation.

The **Mapping Software** has been validated on four healthy volunteers, and the results clearly show its importance: even with similar electrodes placements, inter-subject variability is present up to the point of moving completely the affected distal-location.

While, for the same subject, in cases where different electrodes configurations elicited sensation in the same locations, the mapping results shows that the in-situ pain is different, and the possible swing for the perceived distal-intensity might be much lower. Thus obtaining a Mapping of the subject can clarify with a better accuracy the ideal usable spots.

Indeed, in the final application, the neuroprosthesis will require an initial accurate **calibration** (i.e.: the mapping) for each patient. After this time initially invested, if, as the obtained results suggests, the sensations will be repeatable and stable over time, this means an excellent usability for the patient, that will not have to waste time daily searching for a good setup.

For these reasons the developed software has a significant importance in developing effectively a sensory-feedback neuroprosthesis.

These preliminary results suggest the possibility to use a non-invasive method also for lower-limb amputees and diabetics, but testing with actual patients is strongly required. There's an important need to explore the various possible electrodes placements depending on the type of patient.

Future development includes also the necessity to evaluate smaller types of electrodes, to obtain an higher accuracy in the distal-location.

The use of matrix-electrodes with an electrical stimulator that provides more channel to use concurrently is another path to explore.

8.3 Closed-loop system

The software on the system-controller is ready to implement a closed-loop sensory-feedback neuroprosthesis. The implementation is multithread for security reasons, and the code is enough modular to easily allow the substitution of the stimulator with a different one.

A link to a video of its testing can be found on the Appendix. The video shows how pulsewidth is modulated in real-time, depending on the force read. The oscilloscope had four channel, and thus four different waves can be seen; of course in practice there's no such limitation, meaning that ideally even eight couples of electrodes could be used, which would be the total number of different channels provided by RehaStim. The video shows also the data being read and shown in real-time by the Client GUI.

From a practical point of view, first a mapping can be done on the patient to identify some electrodes placement that elicit useful sensations. Then the stimuli intensity of these positions can be mapped to be driven by force sensors placed inside an insole.

An essential required step is the validation with patients to fully explore the feasibility of the system, as from a technical point of view the system is ready to be tested.

Appendix A

A.1 Global system testing

The following images show the testing of the global system, displaying how the pulsewidth of the various channel is properly modulated accordingly to the pressure sensed by the insole.

Figure A.1 is a single frame extracted by **Video1**, which can be found here: Video1.mp4: https://drive.google.com/open?id=1sqsorD7C3QP9uUbm_jDx_xQR_QkgZTov

Figure A.2 is a single frame extracted by **Video2**, which can be found here: Video2.mp4: https://drive.google.com/open?id=1uUkktPUzJfjd4v-RPdhmk-_Q3dUNcrhy

It should be noted that in these videos the pressure/pulsewidth mapping was made starting from 0. Meaning that the minimum pressure threshold to overcome is not present. This is done in order to provide a better visual representation, otherwise there would be some issues with the oscilloscope triggering and syncing mode. Nonetheless, in a pratical implementation the transfer function used would be the one discussed in Chapter 6.

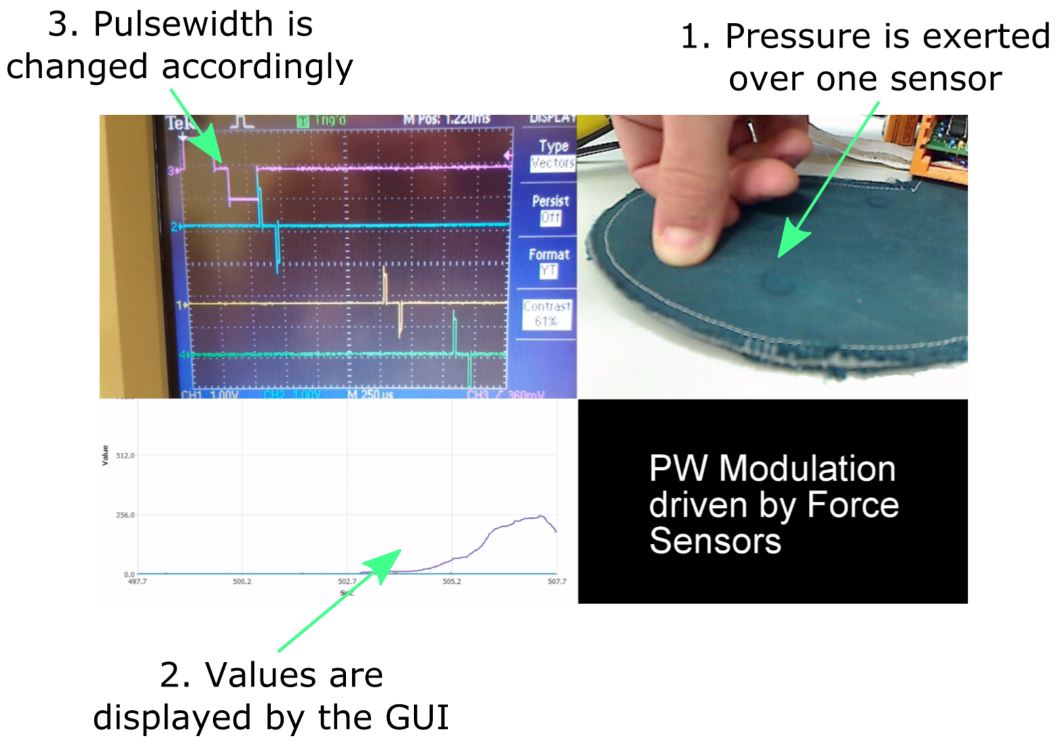

3. Pulsewidth is changed accordingly
1. Pressure is exerted over one sensor
PW Modulation driven by Force Sensors
2. Values are displayed by the GUI

FIGURE A.1: Testing of the modulation of pulsewidth driven by pressure sensors, with manual pressure. Single frame from Video1.

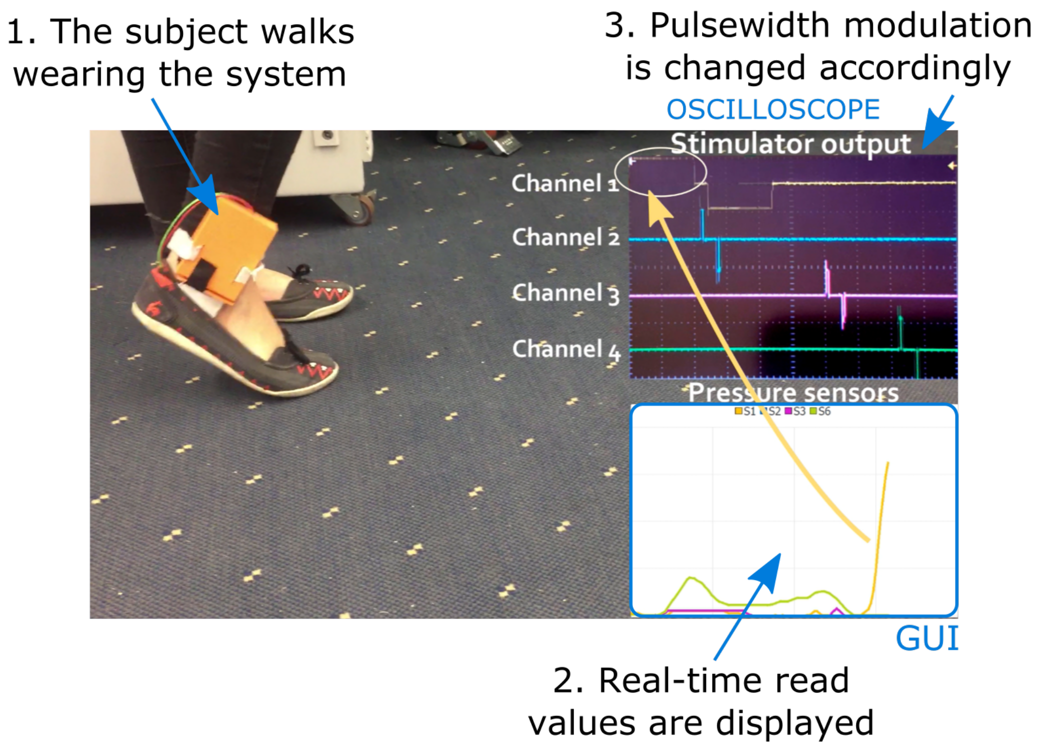
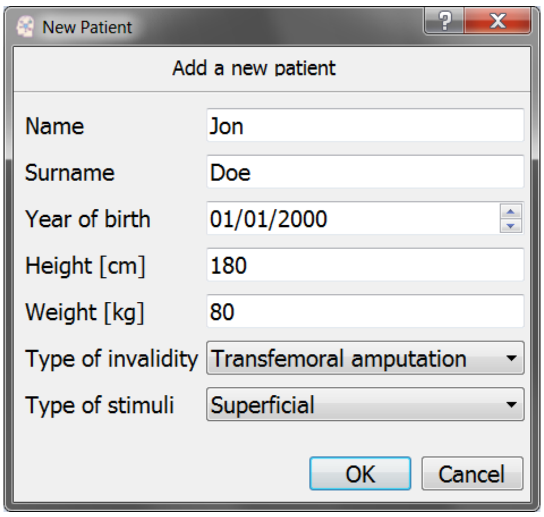

1. The subject walks wearing the system
3. Pulsewidth modulation is changed accordingly
OSCILLOSCOPE
Stimulator output
Channel 1
Channel 2
Channel 3
Channel 4
Pressure sensors
S1
S2
S3
S6
GUI
2. Real-time read values are displayed

FIGURE A.2: Testing of the modulation of pulsewidth driven by pressure sensors, with a walking subject. Single frame from Video2.

Appendix B

Appendix B

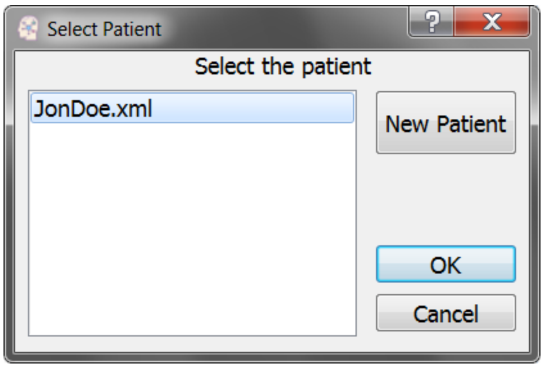
In this Appendix all the windows of the Mapping software will be shown.



New Patient

Add a new patient

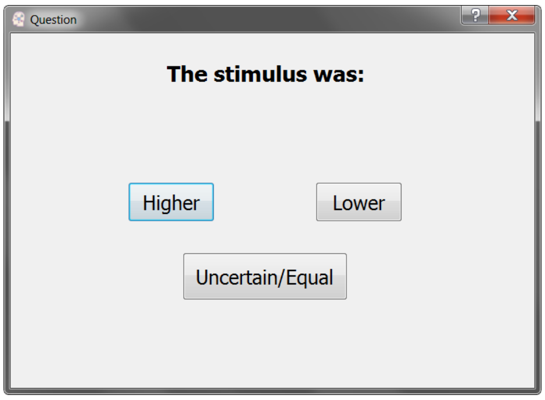
Name	Jon
Surname	Doe
Year of birth	01/01/2000
Height [cm]	180
Weight [kg]	80
Type of invalidity	Transfemoral amputation
Type of stimuli	Superficial

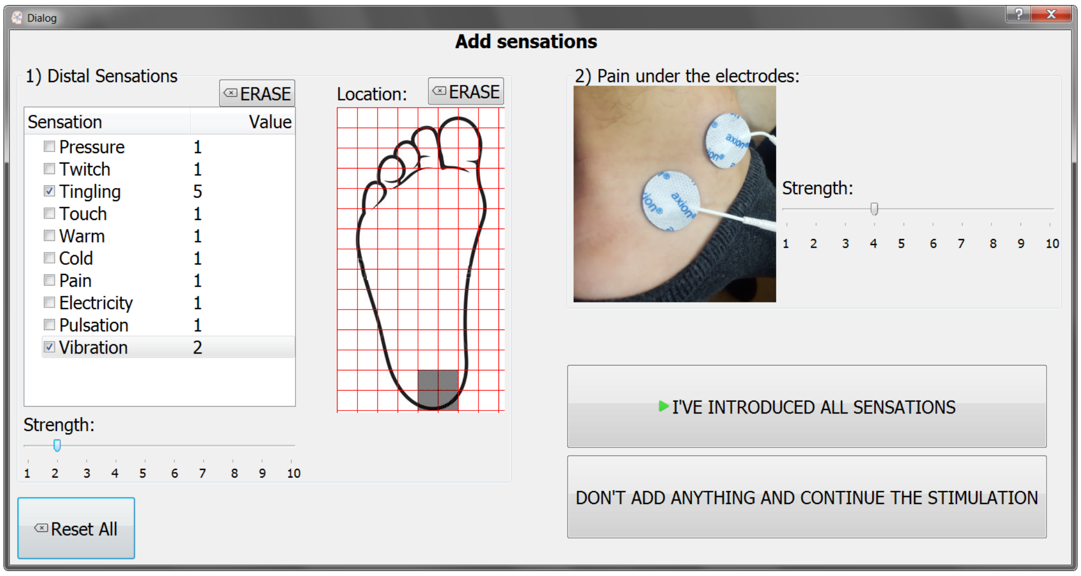


Select Patient

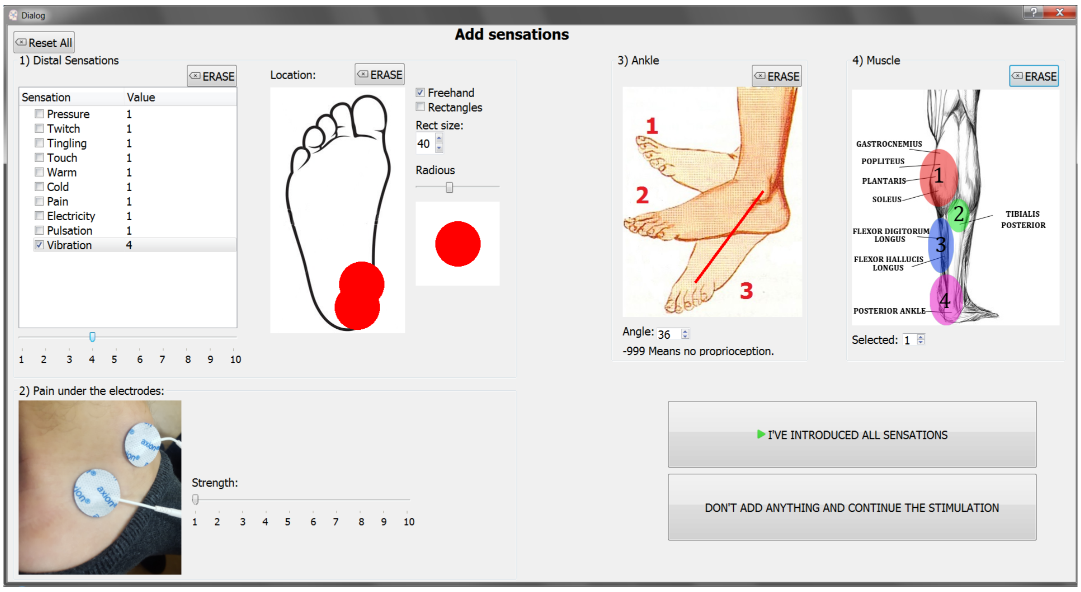
Select the patient

JonDoe.xml





Sensation	Value
☐ Pressure	1
☐ Twitch	1
☑ Tingling	5
☐ Touch	1
☐ Warm	1
☐ Cold	1
☐ Pain	1
☐ Electricity	1
☐ Pulsation	1
☑ Vibration	2

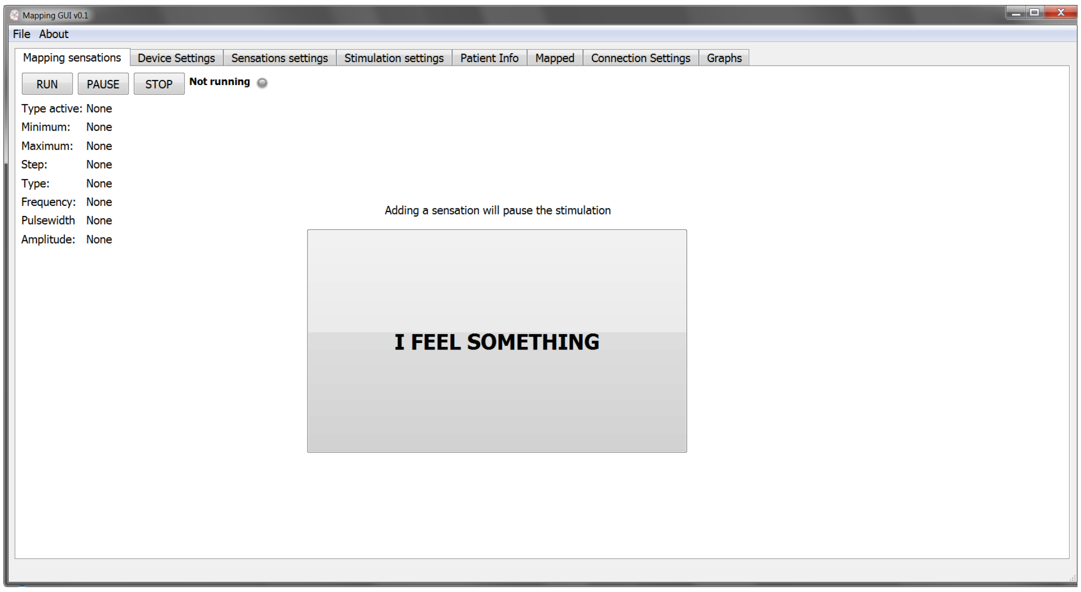


Sensation	Value
☐ Pressure	1
☐ Twitch	1
☐ Tingling	1
☐ Touch	1
☐ Warm	1
☐ Cold	1
☐ Pain	1
☐ Electricity	1
☐ Pulsation	1
☑ Vibration	4

☑ Freehand
☐ Rectangles
Rect size: 40
Radious

Angle: 36
-999 Means no proprioception.

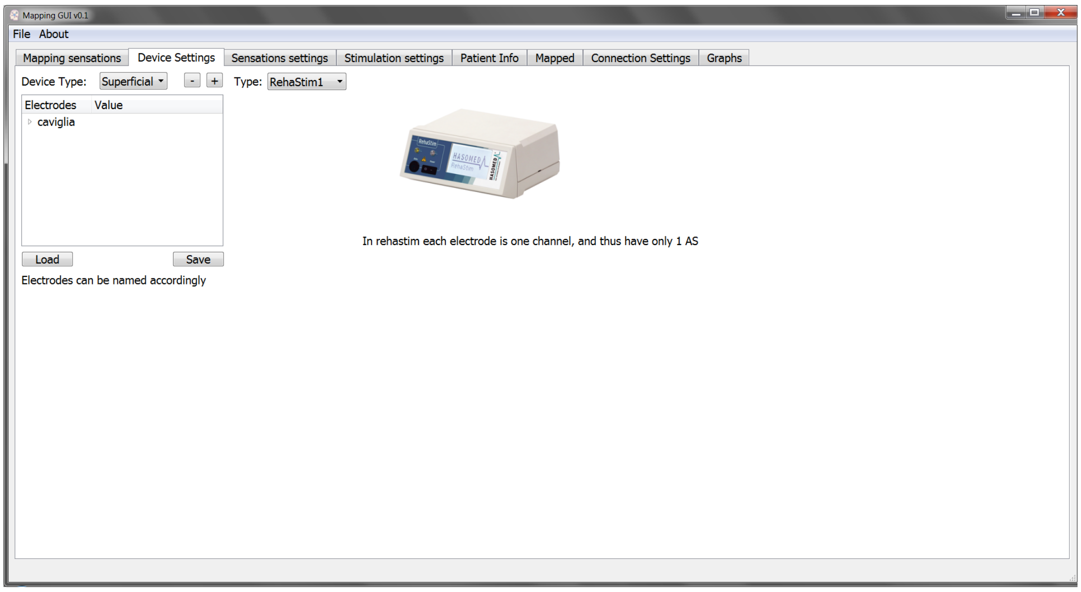
Selected: 1



Not running

Type active:	None
Minimum:	None
Maximum:	None
Step:	None
Type:	None
Frequency:	None
Pulsewidth	None
Amplitude:	None

Adding a sensation will pause the stimulation



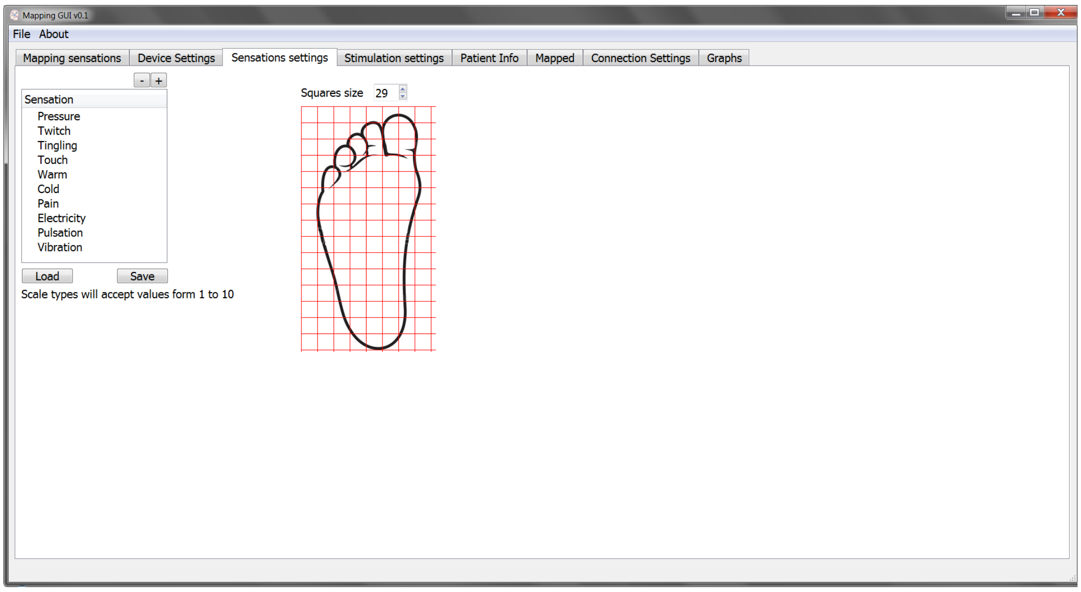
Device Type: Superficial

Type: RehaStim1

Electrodes	Value
caviglia	

Electrodes can be named accordingly

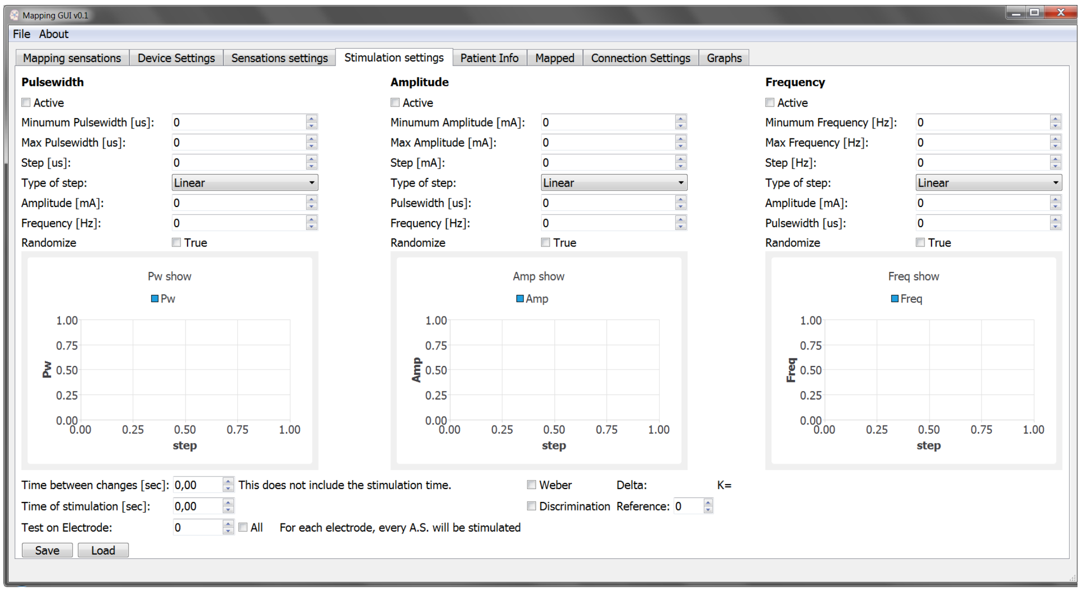
In rehastim each electrode is one channel, and thus have only 1 AS



Sensation
Pressure
Twitch
Tingling
Touch
Warm
Cold
Pain
Electricity
Pulsation
Vibration

Scale types will accept values form 1 to 10

Squares size 29



Pulsewidth

Active	☐
Minumum Pulsewidth [us]:	0
Max Pulsewidth [us]:	0
Step [us]:	0
Type of step:	Linear
Amplitude [mA]:	0
Frequency [Hz]:	0
Randomize	☐ True

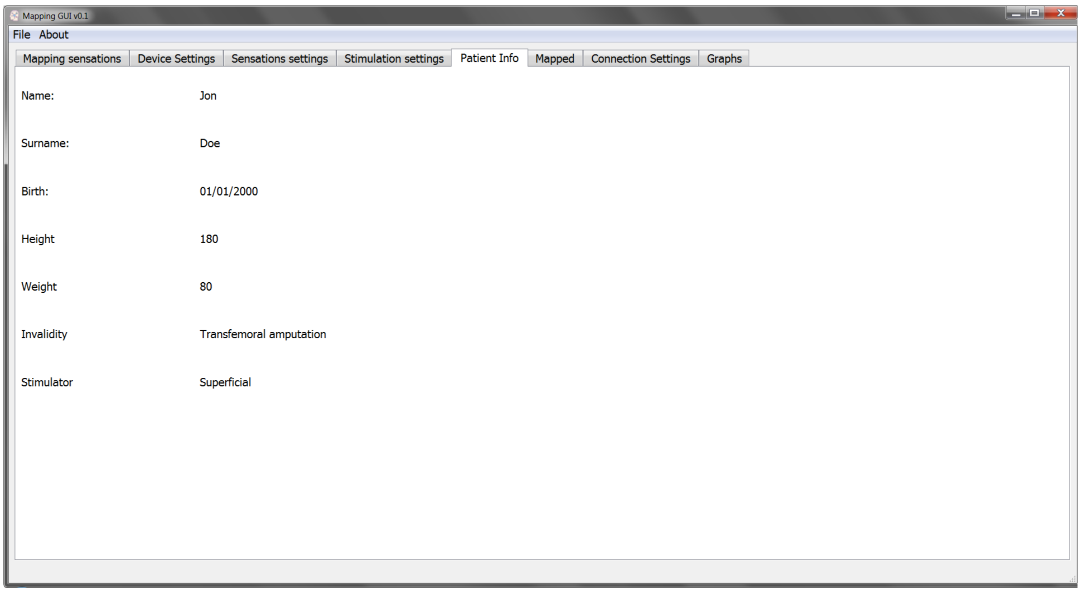
Amplitude

Active	☐
Minumum Amplitude [mA]:	0
Max Amplitude [mA]:	0
Step [mA]:	0
Type of step:	Linear
Pulsewidth [us]:	0
Frequency [Hz]:	0
Randomize	☐ True

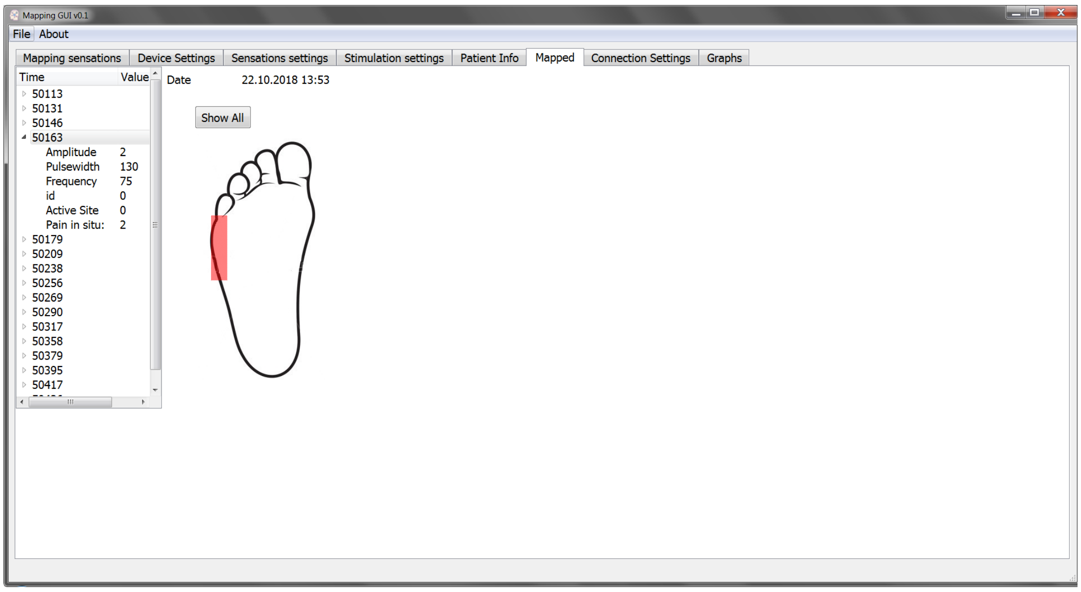
Frequency

Active	☐
Minumum Frequency [Hz]:	0
Max Frequency [Hz]:	0
Step [Hz]:	0
Type of step:	Linear
Amplitude [mA]:	0
Pulsewidth [us]:	0
Randomize	☐ True

Time between changes [sec]: 0,00 This does not include the stimulation time. ☐ Weber Delta: K=
Time of stimulation [sec]: 0,00 ☐ Discrimination Reference: 0
Test on Electrode: 0 ☐ All For each electrode, every A.S. will be stimulated



Name:	Jon
Surname:	Doe
Birth:	01/01/2000
Height	180
Weight	80
Invalidity	Transfemoral amputation
Stimulator	Superficial



Mapping GUI v0.1

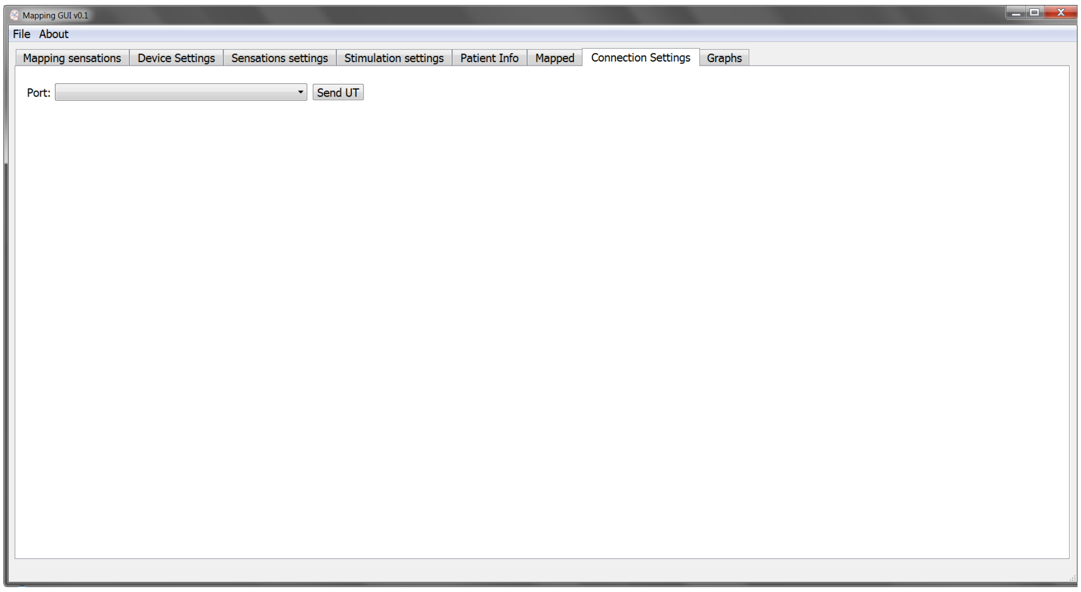
File About

Mapping sensations | Device Settings | Sensations settings | Stimulation settings | Patient Info | Mapped | Connection Settings | Graphs

Time	Value
50113	
50131	
50146	
50163	
Amplitude	2
Pulsewidth	130
Frequency	75
id	0
Active Site	0
Pain in situ:	2
50179	
50209	
50238	
50256	
50269	
50290	
50317	
50358	
50379	
50395	
50417	

Date 22.10.2018 13:53

Show All

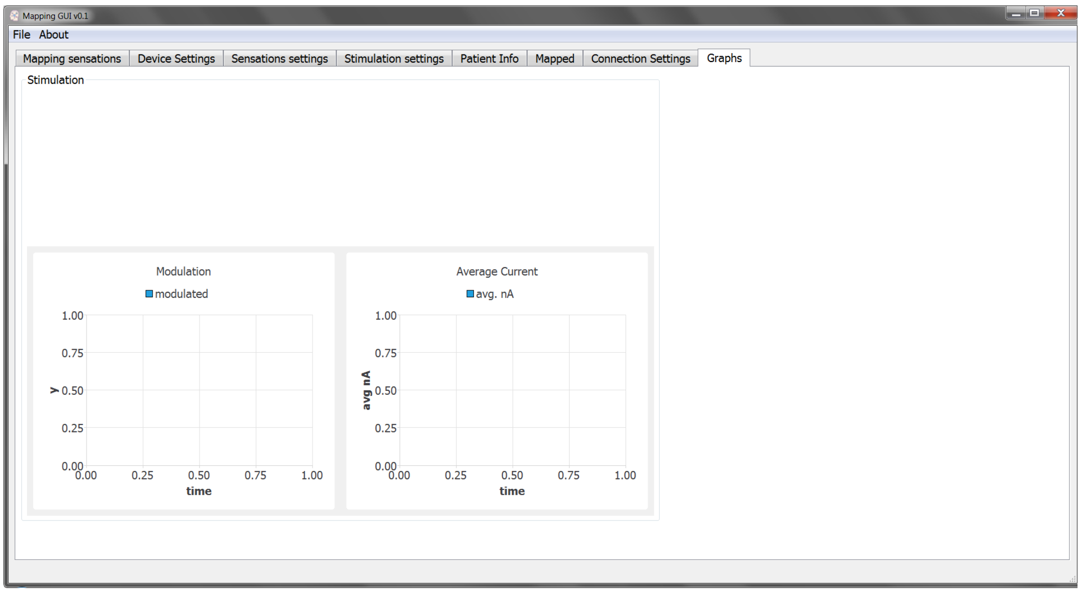


Mapping GUI v0.1

File About

Mapping sensations | Device Settings | Sensations settings | Stimulation settings | Patient Info | Mapped | Connection Settings | Graphs

Port: Send UT



Mapping GUI v0.1

File About

Mapping sensations | Device Settings | Sensations settings | Stimulation settings | Patient Info | Mapped | Connection Settings | Graphs

Stimulation

Bibliography

[1] Uazman Alam et al. "Diabetic neuropathy and gait: a review". In: *Diabetes therapy* 8.6 (2017), pp. 1253–1264.

[2] Alexandra Alvarsson et al. "A retrospective analysis of amputation rates in diabetic patients: can lower extremity amputations be further prevented?" In: *Cardiovascular diabetology* 11.1 (2012), p. 18.

[3] Stuart C Apfel et al. "Positive neuropathic sensory symptoms as endpoints in diabetic neuropathy trials". In: *Journal of the neurological sciences* 189.1-2 (2001), pp. 3–5.

[4] Federica Barberi and Dario Bortolotti. *Development and testing of a sensorized insole for a sensory neuroprosthesis*. 2017. URL: http://hdl.handle.net/10589/133252.

[5] Andrew JM Boulton et al. "Diabetic somatic neuropathies". In: *Diabetes care* 27.6 (2004), pp. 1458–1486.

[6] GH Chai et al. "Phantom finger perception evoked with transcutaneous electrical stimulation for sensory feedback of prosthetic hand". In: *Neural Engineering (NER), 2013 6th International IEEE/EMBS Conference on*. IEEE. 2013, pp. 271–274.

[7] Guohong Chai et al. "Characterization of evoked tactile sensation in forearm amputees with transcutaneous electrical nerve stimulation". In: *Journal of neural engineering* 12.6 (2015), p. 066002.

[8] Open Source community. *Free peer-reviewed portable C++ source libraries*. 2018. URL: https://www.boost.org/.

[9] cppreference. *The condition variable header*. 2018. URL: https://en.cppreference.com/w/cpp/thread/condition_variable.

[10] cppreference. *The Mutex header*. 2018. URL: https://en.cppreference.com/w/cpp/thread/mutex.

[11] Simona Crea et al. "Time-discrete vibrotactile feedback contributes to improved gait symmetry in patients with lower limb amputations: Case series". In: *Physical therapy* 97.2 (2017), pp. 198–207.

[12] Edoardo D'Anna et al. "A somatotopic bidirectional hand prosthesis with transcutaneous electrical nerve stimulation based sensory feedback". In: *Scientific Reports* 7 (Dec. 2017). DOI: 10.1038/s41598-017-11306-w.

[13] Qian Dong et al. "Entrapment neuropathies in the upper and lower limbs: anatomy and MRI features". In: *Radiology research and practice* 2012 (2012).

[14] Richard E Fan et al. "A haptic feedback system for lower-limb prostheses". In: *IEEE Transactions on Neural Systems and Rehabilitation Engineering* 16.3 (2008), pp. 270–277.

[15] Todd R Farrell and Richard F Weir. "The optimal controller delay for myoelectric prostheses". In: *IEEE Transactions on neural systems and rehabilitation engineering* 15.1 (2007), pp. 111–118.

[16] Johanna C Forst et al. “Surface electrical stimulation to evoke referred sensation”. In: *Journal of Rehabilitation Research & Development* 52.4 (2015), pp. 397–407.

[17] Robert Gailey et al. “Review of secondary physical conditions associated with lower-limb amputation and long-term prosthesis use.” In: *Journal of Rehabilitation Research & Development* 45.1 (2008).

[18] Erádio Gonçalves Junior, Rodrigo José Knabben, and Soraia Cristina Tonon da Luz. “Portraying the amputation of lower limbs: an approach using ICF”. In: *Fisioterapia em Movimento* 30.1 (2017), pp. 97–106.

[19] Emily L Graczyk et al. “Sensory adaptation to electrical stimulation of the somatosensory nerves”. In: *Journal of neural engineering* 15.4 (2018), p. 046002.

[20] Emily L Graczyk et al. “The neural basis of perceived intensity in natural and artificial touch”. In: *Science translational medicine* 8.362 (2016), 362ra142–362ra142.

[21] Qt Group. *Complete software development framework*. 2018. URL: https://www.qt.io/what-is-qt/.

[22] Qt Group. *Qt Charts*. 2018. URL: https://github.com/qt/qtcharts/blob/5.11/README.

[23] Qt Group. *Supported Platforms*. 2018. URL: http://doc.qt.io/qt-5/supported-platforms.html?hsLang=en.

[24] Hasomed. *RehaStim Operating Manual*. 2009.

[25] Hasomed. *ScienceMode Manual*. 2009.

[26] Mary Josephine Hessert et al. “Foot pressure distribution during walking in young and old adults”. In: *BMC geriatrics* 5.1 (2005), p. 8.

[27] ICF. *ICF Core Set for persons following an amputation*. 2017. URL: https://www.icf-research-branch.org/icf-core-sets-projects2/other-health-conditions/icf-core-set-for-persons-following-an-amputation.

[28] Future Technology Devices International. *FTDI Chips*. 2018. URL: https://www.ftdichip.com/FTProducts.htm.

[29] Chris Kohlhof. *Asio C++ Library*. 2018. URL: https://think-async.com/.

[30] Mengnan Li et al. “Discrimination and recognition of phantom finger sensation through transcutaneous electrical nerve stimulation”. In: *Frontiers in neuroscience* 12 (2018), p. 283.

[31] Bijan Najafi et al. “Using plantar electrical stimulation to improve postural balance and plantar sensation among patients with diabetic peripheral neuropathy: a randomized double blinded study”. In: *Journal of diabetes science and technology* 11.4 (2017), pp. 693–701.

[32] Oracle. *Oracle VM VirtualBox*. 2018. URL: https://www.oracle.com/it/virtualization/virtualbox/.

[33] Oracle. *What is a Data Race*. 2010. URL: https://docs.oracle.com/cd/E19205-01/820-0619/geojs/index.html.

[34] PPS. *Foot Pressure Measurement System*. 2018. URL: https://pressureprofile.com/foot-pms.

[35] Andri Primadi et al. “Neurologic injuries after primary total ankle arthroplasty: prevalence and effect on outcomes”. In: *Journal of foot and ankle research* 8.1 (2015), p. 55.

[36] Debian Project. *The universal operating system*. 2018. URL: https://www.debian.org/.

[37] Stanisa Raspopovic et al. “Restoring natural sensory feedback in real-time bidirectional hand prostheses”. In: *Science translational medicine* 6.222 (2014), 222ra19–222ra19.

[38] Henry Shin et al. "Evoked haptic sensations in the hand via non-invasive proximal nerve stimulation". In: *Journal of neural engineering* 15.4 (2018), p. 046005.

[39] Kathleen A Sluka and Deirdre Walsh. "Transcutaneous electrical nerve stimulation: basic science mechanisms and clinical effectiveness". In: *The Journal of pain* 4.3 (2003), pp. 109–121.

[40] TalOrg. *Building Qt for Raspberry Pi on Debian Jessie*. 2018. URL: `https://web.archive.org/web/20180413233208/http://www.tal.org/tutorials/building\_qt\_5\_for\_raspberrypi\_jessie`.

[41] Tekscan. *Tekscan's Digital Footprint Technology*. 2018. URL: `https://www.tekscan.com/application-group/embedded-sensing/foot-mapping`.

[42] Solomon Tesfaye et al. "Diabetic neuropathies: update on definitions, diagnostic criteria, estimation of severity, and treatments". In: *Diabetes care* 33.10 (2010), pp. 2285–2293.

[43] The-Amputee-Coalition. *Managing Phantom Pain*. 2018. URL: `https://www.amputee-coalition.org/limb-loss-resource-center/resources-for-pain-management/managing-phantom-pain/`.

[44] Robert Van Deursen. "Mechanical loading and off-loading of the plantar surface of the diabetic foot". In: *Clinical Infectious Diseases* 39.Supplement_2 (2004), S87–S91.

[45] L. Yang et al. "Utilization of a lower extremity ambulatory feedback system to reduce gait asymmetry in transtibial amputation gait". In: *Gait & Posture 36* (2012).