UNIVERSITÉ DE MONTRÉAL

ROBUST PRE-CLINICAL SOFTWARE SYSTEM FOR REAL TIME NIRS AND EEG MONITORING

MAHYA DEHBOZORGI

INSTITUT DE GÉNIE BIOMÉDICAL

ÉCOLE POLYTECHNIQUE DE MONTRÉAL

MÉMOIRE PRÉSENTÉ EN VUE DE L'OBTENTION

DU DIPLÔME DE MAÎTRISE ÈS SCIENCES APPLIQUÉES

(GÉNIE BIOMÉDICAL)

DÉCEMBRE 2014

UNIVERSITÉ DE MONTRÉAL

ÉCOLE POLYTECHNIQUE DE MONTRÉAL

Ce mémoire intitulé:

ROBUST PRE-CLINICAL SOFTWARE SYSTEM FOR REAL TIME NIRS AND EEG MONITORING

présenté par : DEHBOZORGI Mahya

en vue de l'obtention du diplôme de : Maîtrise ès sciences appliquées

a été dûment accepté par le jury d'examen constitué de :

M. COHEN-ADAD Julien, Ph. D., président

M. SAWAN Mohamad, Ph. D., membre et directeur de recherche

M. POULIOT Philippe, Ph. D., membre et codirecteur de recherche

Mme CHERIET Farida, Ph. D., membre

DEDICATION

To my wonderful and loving family…

ACKNOWLEDGEMENTS

The work accomplished in this thesis would not have been possible without the support and encouragement of many. I would like to express my sincere gratitude to my supervisor, Professor Mohamad Sawan, for providing me the opportunity to be part of his research team and the multidisciplinary project which met my interest for biomedical applications. His expertise, guidelines and motivations along the way made this journey a pleasant experience.

My appreciation also extends to my co-supervisor, Dr. Philippe Pouliot, for his valuable advice, patience and timely suggestions at every stage. His constant availability and positive feedback always clarified the path and was a constant source of motivation.

Special thanks to the collaborators of the Imaginc project, in particular Mrs. Phetsamone Vannasing and Ali Kassab at Sainte-Justine Hospital. They have provided valuable feedback and suggestions on my work and offered a unique opportunity to gain insight from a medical perspective. This work has been funded by the Heart and Stroke Foundation and Canadian Institute of Health Research.

I would also like to take this opportunity to thank my various colleagues and staff members at PolyStim who have provided a memorable and cheerful experience during my time. Special thanks to Jerome Le Lan, for his collaboration and help all throughout the project. Thanks to Nadia, Ghazal, Sina and Elie for the friendly and stimulating lab environment. It has been a pleasure working with all of you.

My deepest love and appreciation goes to my family and friends for their continuous love and support. Their confidence in me and has been invaluable to the completion of this thesis, and have always been there for me in moments of doubt.

RÉSUMÉ

Ce mémoire présente la conception et l'implémentation d'un logiciel dédié au support d'un système bimodal NIRS et EEG d'imagerie cérébrale en temps réel. En effet, l'accès à l'information en temps réel concernant l'activité cérébrale est un facteur important permettant la détection de tout changement au niveau cortex du cerveau à un stade précoce. Or, les logiciels actuellement disponibles comparables à celui présenté ici n'offrent qu'une possibilité d'ajustement limitée des paramètres en temps réel ainsi que peu de fonctionnalité permettant l'analyse rapide et efficace des données.

Le travail présenté dans ce présent mémoire a été réalisé au sein du groupe IMAGINC. Un groupe de recherche multidisciplinaire ayant comme objectif le développement d'un système d'imagerie cérébrale portatif, non invasif et sans-fil permettant d'imager le cortex entier en temps réel. Le module d'acquisition de données de ce système enregistre l'information de 128 canaux NIRS et 32 canaux EEG ainsi que différents accéléromètres et canaux analogiques le tout par l'entremise d'optodes et d'électrodes placées sur un casque d'enregistrement. Ces données sont ensuite envoyées par un lien de communication sans-fil au logiciel qui recueille et affiche l'état hémodynamique du sujet par l'entremise de son interface graphique. Il est ensuite possible de choisir différentes vues des cartes 2D du cerveau sur lesquelles les changements hémodynamiques sont présentés. La surveillance à distance de l'état du sujet est aussi possible puisque ces données peuvent être retransmises vers un autre ordinateur par un lien sans-fil. De plus, de par son interface graphique conviviale et intuitive, l'usager peut facilement ajuster différents paramètres de test tout au long de l'acquisition de données sans même l'interrompre. Dans le but d'optimiser les paramètres pour chaque sujet, une fonction de calibration automatique ajustant l'intensité d'illumination de chacun des émetteurs en quelques secondes a été implémentée. Pour faciliter le processus de test, il est possible de télécharger des fichiers (bipolaire et montage référentielle) contenant des paramètres de configurations préétablies pour le NIRS et l'EEG. Enfin, il est possible de faire une analyse automatique et rapide de l'état de tous les canaux NIRS durant les tests afin d'assurer une bonne connexion ainsi que la validité des données. Le système conçu est en mesure d'enregistrer et de traiter des données en temps réel sur une période de 24 heures. Les résultats obtenus ont été validés en utilisant des logiciels d'analyse de données NIRS similaires durant des tâches de finger tapping induisant un changement hémodynamique chez les sujets.

ABSTRACT

This master's thesis presents the design and implementation of a real-time software system to support a bimodal NIRS and EEG brain imaging device. Real-time information on brain activity is an important factor in early detection and diagnosis at the top level of the cortex of various brain disorders. Current software systems provide limited real-time parameter adjustment and automated features for quick and easy analysis.

The project presented in this master's thesis is part of the multidisciplinary IMAGINC research group, with the objective of developing a wireless, non-invasive and portable brain imaging system that allows imaging of the whole cortex in real time. The hardware system is capable of recording data from 128 NIRS and 32 EEG channels, as well as additional accelerometer and analog channels through the optodes and electrodes mounted onto the helmet. The software system acquires the real-time data from the hardware module using a wireless connection and displays the hemodynamic variations on the user interface. The change in hemodynamic activity is displayed on a 2D map of the brain, with selection of different views. Remote monitoring is also possible since the data can be transferred wirelessly to another computer. Through the user-friendly and intuitive user interface, the user can control and adjust various test parameters throughout the acquisition without any interruption. In order to achieve maximum illumination setting for individual subjects there is an automatic calibration function that quickly adjusts the illumination intensity for each of the emitters in just a few seconds. Previously defined NIRS and EEG configuration files (bipolar and referential montage) can be uploaded for easy testing. An automated analysis feature quickly analyzes and reports the status of all NIRS channels during the test to ensure good connection and valid results. The designed system can successfully record and process data for a continuous period of up to 24 hours. The results have been validated using similar NIRS data analysis software during figure tapping tasks and the hemodynamic variations were as expected.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS AND ABBREVIATIONS

ADC	Analog-to-digital converter
APD	Avalanche photodiode
BCI	Brain computer interface
CBF	Cerebral brain flow
CW	Continuous wave
DLL	Dynamic link library
DPF	Differential path length factor
EEG	Electroencephalography
fMRI	Functional magnetic resonance imaging
fNIR	Functional near infrared imaging
GUI	Graphical user interface
HbO	Oxy-hemoglobin
HbR	Deoxy-hemoglobin
HbT	Total hemoglobin
LED	Light emitting diode
MBLL	Modified Beer-Lambert law
MVC	Model-view-controller
NIRS	Near infrared spectroscopy
PC	Personal computer
PVF	Partial volume factor
SNR	Signal-to-noise ratio
TCP/IP	Transmission Control Protocol/Internet Protocol

UART	Universal asynchronous receiver/transmitter
UI	User interface
UML	Unified modeling language

CHAPTER 1 INTRODUCTION

The human brain is a complex structure which has been studied for the past centuries in order to understand the underlying mechanism and to better comprehend the chain of reactions. The human nervous system and the brain primarily consist of millions of neurons, organized in networks. All mental activity and motor tasks are the results of electrical signals passing between the axons of neighboring neurons. This message is passed from the brain to the desired organ and vice versa, in order to accomplish a task. In a healthy brain, this communication is performed quickly and automatically. However in the case of a neurological disease, the normal path of information is interrupted and can lead to devastating results. Over one billion people worldwide suffer from neurological disorders which includes degenerative diseases (such as Parkinson's and Alzheimer's disease), cerebrovascular diseases (such as such as stroke and vascular disorders) and convulsive disorders (such as epilepsy) just to name a few.

Monitoring and diagnostic devices are becoming increasingly important tools in detecting various health disorders and treating patients to prevent and minimize the side effects. They provide a valuable source of information for the investigation of developmental cognitive neuroscience and a potential factor in the advancement of our understanding of the brain's function and mal function. Of the various existing brain imaging technologies, such as MRI, CT and PET scan, EEG and TDC/ECD (transcranial/extracranial Doppler sonography), Near Infrared Spectroscopy (NIRS) imaging is gaining popularity due to its ability to continuously monitor cerebral activity in an unconstrained environment at the patient's bedside, while providing an inexpensive solution. It is a practical tool in both research and clinical settings, from psychology and neurology units to the treatment of cerebrovascular diseases, epileptic disorders and severe brain injury (Obrig, 2014). Our aim is to develop a brain imaging device that will help us to better understand the ongoing cerebral activity. The developed system consists of a signal acquisition prototype, a helmet which holds the optodes and electrodes, and the software system to support the device. The current hardware system is a multi-channel brain imaging system based on NIRS and Electroencephalography (EEG) techniques. Both of these modalities are portable and relatively inexpensive. NIRS is a non-invasive neuroimaging technique that measures changes in cerebral blood oxygenation associated with brain activity while EEG measures the electrophysiological changes on the scalp. The supporting user interface displays

the acquired information to the clinician for diagnostic purposes. In order to provide instant feedback, the acquired data is displayed in real-time, and allows for changes to the acquisition parameters during the recording. It is easy to use while allowing for control and modification of various parameters based on the specific test and subject. The combination of the hardware and software system will serve as a tool for long term clinical monitoring of patients with various neurological disorders. Subjects can easily wear the portable device on their belt while having mobility to freely move around. This can also help in reducing their physical and psychological stress that accompanies long term test sessions. A clinical neurologist can easily collect the data through the shared network and perform real time analysis from his/her personal computer.

The rest of this chapter concerns the theoretical aspects of NIRS and EEG technology and a brief overview of the software development stages and requirements that are the basis for this project.

1.1 Near Infrared Spectroscopy

1.1.1 Theory and principles

NIRS is a brain imaging technique that measures the physiological events associated with brain activity while performing various tasks. The primary component of red blood cells are the protein called hemoglobin. It is responsible for oxygen transfer from the lungs to the various organs, including the brain and vice versa. Hemoglobin is available in two main forms, oxygenated (HbO) and reduced (HbR). When an area of the brain is activated, resulting in neuronal activity, there is an increase in metabolic demand for oxygen. As a result, the cerebral blood flow (CBF) to the active region increases. There is an early decrease in HbO followed by a long lasting increase in oxy-hemoglobin. As a result there is a direct correlation between local changes of HbO and HbR concentration and brain activity.

The main concept underlying NIRS is the difference in absorption characteristics of oxygenated and deoxygenated hemoglobin when exposed to IR (infrared) and red light. Figure 1.1 shows the spectroscopic and absorptive properties of oxygenated and deoxygenated hemoglobin in the near infra-red region. In the optical window from 650 – 900 nm (NIR region), the absorption coefficients of HbO and HbR intersect at 805 nm, which is the isosbestic point (Koizumi et al., 2003; Zijlstra et al., 2000). The reason for choosing this optical window is that

NIR light can penetrate through skin and bones fairly easy in this range. Light below this range is mostly absorbed by hemoglobin, and light above this range is mainly absorbed by water. Using these principles, infrared light passes through the brain tissue and the diffuse reflectance is measured from the tissue a few centimeters apart from the emitting source(Ferrari et al., 2004; Ferrari et al., 2012; Giacometti et al., 2013; Wolf et al., 2007).

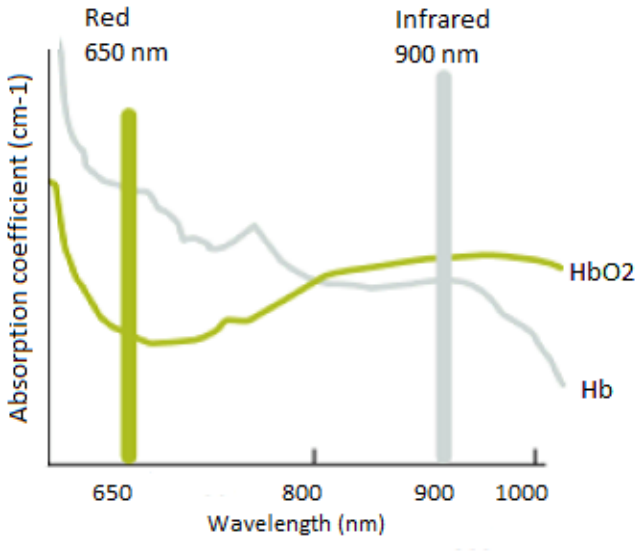


Figure 1.1: Absorption spectra of HbO and HbR in the near infra-red region

By passing light at different wavelengths through the brain tissue and based on the difference in the absorption spectrum of HbO and HbR, we can measure the relative changes of hemoglobin concentration based on the modified Beer-Lambert law (MBLL) (Delpy et al., 1988). In our system, we use two wavelengths of light, at 735 and 850 nm, in order to have one above and one below the isosbestic point to be able to differentiate the two chromophores. The recorded activity at the path between one emitter and one detector is referred to as one "channel". With multiple source-detector pairs covering the entire scalp, we can observe the changes simultaneously in all regions of the brain. In our system, there are 32 light sources and 32 detectors. Each light source can be coupled with 4 detectors, for a total of 128 channels at each of the two wavelengths.

Figure 1.2 shows a typical placement of emitters and detectors in a NIRS system. The light emitted from the source travels in a banana shape path through the tissue and reaches the detector. By increasing the distance between the source and detector we can obtain information

about the hemodynamic response in deeper tissues of the brain (Fukui et al., 2003). However this distance varies based on the initial light intensity and age of patients (adults vs. infants), as well as the amount of absorbed light (Scholkmann et al., 2013).

It has been shown that an emitter-detector separation of 2 cm is sufficient for imaging the infant cortex (Duncan et al., 1995; Taga et al., 2003). Various existing devices use an inter-optode distance of 3 cm for the adult cortex (Van der Zee et al., 1992). Similarly in our device we have set the emitter and detector separation at 3 cm. The light penetration depth is around half the physical distance, which is 1.5 cm.

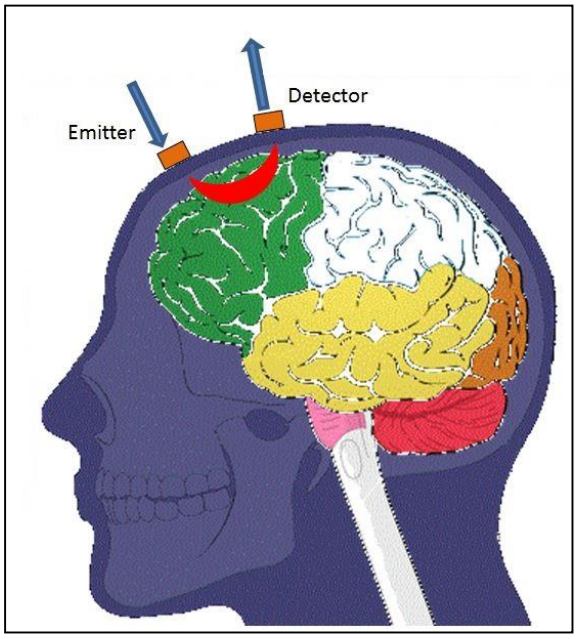


Figure 1.2: Light travel path between the emitter and detector. The light is emitted from the emitter, travels through the tissue and is received at the detector

The brain consists of several layers of tissue, which absorb part of the illuminated light. Therefore only a small portion of the injected light reaches the detector and is conveyed to the measuring device. Light attenuation is due to absorption and scattering within the tissue, both for the entering and exiting light. Figure 1.3 shows the different layers of the human head that light must penetrate through.

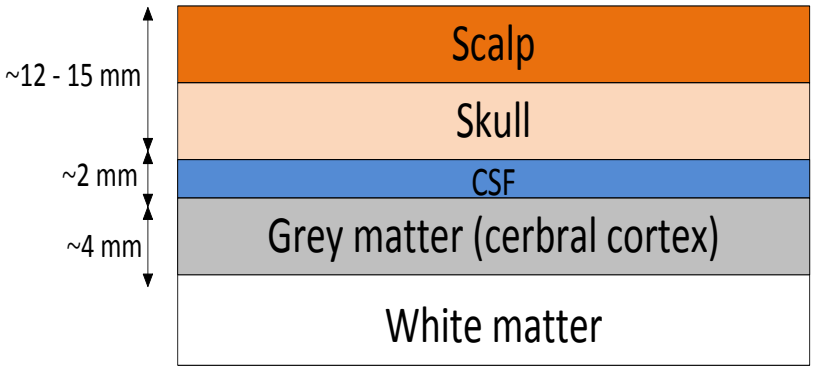


Figure 1.3: Various tissues of the human scalp

It was first demonstrated in 1977 by Jobsis that by measuring the light intensity observed at the detector, we can calculate the changes in hemoglobin concentrations using the Beer-Lambert law (Jobsis, 1977). The Beer-Lambert law relates the light intensity of emitted and detected light with changes in hemoglobin concentrations (Boas et al., 2004; Duncan et al., 1995). Assuming that the material is a homogenous substance, we can determine the amount of light absorption using the following equation:

$$A = log_{10}\frac{I_0}{I} = \alpha \cdot c \cdot l\ , \quad \alpha = 4\pi k/\lambda \tag{1.1}$$

where A is the attenuation, I_0 is the incident light intensity and I is the measured light intensity at the detector. In other words, the attenuation is proportional to the absorption coefficient of the chromophores, which is hemoglobin, the physical distance l between the source and detector, as well as the concentration of the substance, c. The absorption coefficient α is measured using the extinction coefficient at the specific wavelength λ which is determined from the literature. While this law provides a simple means of relating the light absorption to the chemical concentration of the substance, it only holds as long as the photons of the light source are either completely absorbed or transmitted in a straight line through the matter. However due to the highly scattering nature of brain tissue, there will inevitably be loss of light and the photons will travel a longer path length through the tissue. Thus it does not hold true for measurements of the brain tissue (Boas et al., 2001; Obrig et al., 2003).

In order to apply it to brain oxygenation, we must include an additive term to account for scattering losses, as well as the change in path length. The modified Beer-Lambert law is defined as:

$$A = \alpha \cdot c \cdot l \cdot DPF + G \tag{1.2}$$

where DPF is the differential path length factor, which accounts for the longer path length due to scattering. The value of DPF can be determined from experimentally derived studies in a non-homogenous solution based on Monte-Carlo simulations in a way which the key measurement is the change in transmitted light intensity (Duncan et al., 1995; Hemmati et al., 2012; Hiraoka et al., 1993). It has been found to be dependent on age, gender and wavelength of lights, and varies for different tissue (Essenpreis et al., 1993). The term G is an additive scalar term for the measure of the signal loss due to scattering and is unknown. In our application which is continuous wave (CW) NIRS we are interested in the changes in concentration, and not the absolute values. Therefore under this assumption of constant light scattering the term G cancels out when calculating the difference in attenuation for HbO and HbR. The changes in light absorbance can be calculated as follows:

$$\Delta A = \alpha \cdot \Delta c \cdot l \cdot DPF \tag{1.3}$$

When the tissue is illuminated at two different wavelength, λ_1 and λ_2, we can derive the following set of equations:

$$\Delta A(\lambda_1) = l \cdot DPF(\lambda_1)(\alpha_{HbO}^{\lambda_1}\Delta c_{HbO} + \alpha_{HbR}^{\lambda_1}\Delta c_{HbR}) \tag{1.4}$$

$$\Delta A(\lambda_2) = l \cdot DPF(\lambda_2)(\alpha_{HbO}^{\lambda_2}\Delta c_{HbO} + \alpha_{HbR}^{\lambda_2}\Delta c_{HbR}) \tag{1.5}$$

Hence we have a set of 2 equations with 2 unknowns, and solving for Δc_{HbO} and Δc_{HbR}, we have:

$$\begin{bmatrix}\Delta c_{HbO} \\ \Delta c_{HbR}\end{bmatrix} = \begin{bmatrix}\alpha_{HbO}^{\lambda_1} & \alpha_{HbR}^{\lambda_1} \\ \alpha_{HbO}^{\lambda_2} & \alpha_{HbR}^{\lambda_2}\end{bmatrix}^{-1} \begin{bmatrix}\frac{\Delta A(\lambda_1)}{l*DPF(\lambda_1)} \\ \frac{\Delta A(\lambda_2)}{l*DPF(\lambda_2)}\end{bmatrix} \tag{1.6}$$

Thus we are able to calculate the change in concentration for each of the HbO and HbR values in units of μmolars. Total hemoglobin concentration, Δc_{HbT}, is the sum of the previous two values:

$$\Delta c_{HbT} = \Delta c_{HbO} + \Delta c_{HbR} \tag{1.7}$$

This method is based on CW-NIRS instruments. The term continuous defines that the result is based solely on the light intensity measurement at the detector. Using this method, we cannot

determine the absolute concentrations and the results obtained are relative. If the value of DPF is determined, we can calculate the absolute concentrations. In contrast to CW NIRS, there are time and frequency domain techniques, which in addition to the light intensity also measure the time of flight and phase shift in order to determine the absolute value of hemoglobin concentration (Pellicer et al., 2011; Wabnitz et al., 2010). The details of these methods are discussed in the next section.

Time domain NIRS

In time domain NIRS technology, in addition to the intensity it also measures the time-of-flight, which is the time required for the injected light to travel through the tissue. Short pulses of light are injected into the tissue and the attenuation is measured by a single photon counting detector that measures the arrival time of the photons. As a result we can characterize the absolute optical properties of the brain tissue (light scattering and absorption coefficients), and calculate the absolute values of HbO and HbR. Other advantages include higher spatial resolution and penetration depth. However the drawbacks are the cost, lower sampling rate (temporal resolution in the sub-nanosecond scale), instrument size and weight.

Frequency domain NIRS

In frequency domain NIRS devices, the emitted light is modulated and sent through the tissue. At the detector the light intensity, phase shift and modulation depth is measured, and similar to time-domain NIRS, the corresponding time of flight is obtained. The advantages of frequency domain NIRS include the high sampling rate (temporal resolution at frequencies up to nearly 1 GHz) and absolute value calculation of the optical properties, as well as precise separation of absorption and scattering effects. However they have limited penetration depth, are more costly and less portable than CW NIRS devices.

Of the three mentioned techniques, time domain NIRS provides the most information about the photon migration through tissue, but at the same time comes with a lower temporal resolution and more complex technology. Figure 1.4 shows a visualization of the three different techniques.

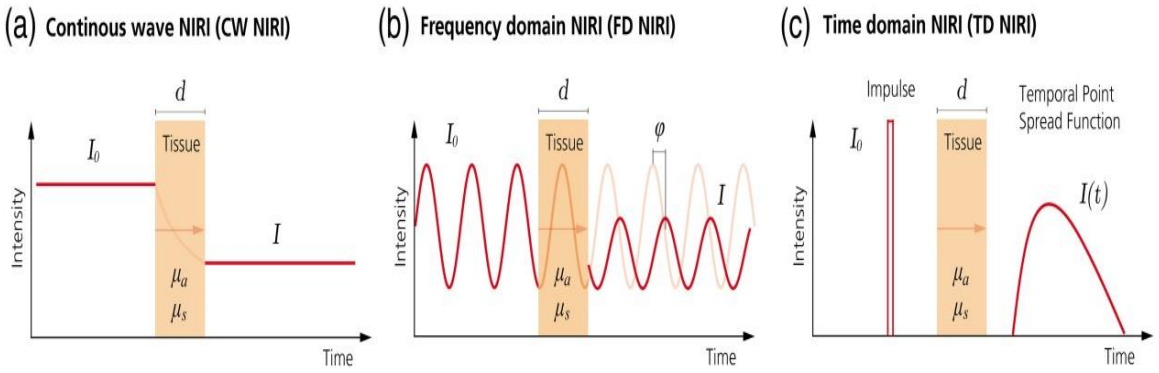


Figure 1.4: Illustration of the three different NIRI techniques from (Scholkmann et al., 2014)

1.2 Electroencephalography

Electroencephalography is a brain imaging technique that measures the neuronal electric activity by electrodes placed on the scalp, or in invasive methods in the cerebral cortex. It was first used by Berger in 1929 to successfully measure the EEG signals from the scalp surface (Haas, 2003; Nunez et al., 2006). The underlying source of this electrical activity is the polarized neurons of various membranes. Constant ion exchange creates an ionic current. When this reaches the electrodes on the scalp, it creates a voltage fluctuation that is measured by the multiple electrodes and sent to the recording device. The measured voltage is relative to a reference electrode which is at a different position than the recording electrode. The midline position is usually used since it does not amplify the signal differently in the two hemispheres. Another common reference point is at the earlobes or mastoids. The resulting measured EEG activity is the summation of the synchronous activity of millions of neurons within the same region; individual neuron activity is too small to measure and thus it is difficult to localize the source of activity.

EEG has a temporal resolution in the range of sub-milliseconds and can be used for real time monitoring applications. However the spatial resolution is generally low. This is due to the fact the underlying neuronal activity must pass through several layers including the skull, CSF and other layers of tissue between the cortex and electrodes. In conventional EEG sessions, the electrodes are fixed to the patients scalp using conductive paste to ensure direct contact with scalp. The electrodes are either embedded into a helmet, or individually wired. Figure 1.5 shows an EEG cap with multiple electrodes.

Figure 1.5: EEG sensing device from Neuroscan

Patient movement can generate artifacts that alter the signals. Sweating and loose electrode connections can also affect the EEG signal quality. If the ground electrode has a bad connection it can cause 50/60 Hz artifacts due to the power system's frequency. These factors need to be taken into consideration in order to obtain a stable and artifact-free signal. Typically the EEG signals are amplified and noise-reduction techniques are applied to minimize the unwanted effects, through high common mode rejection and low amplifier noise. In video EEG, the subject activity is also recorded as a video, to help in continuous monitoring so the staff can respond immediately if there are any abnormalities observed. Our system can be used with up to 32 EEG electrodes placed on various points on the scalp. The electrical activity is measured and sent to the prototype for further signal processing and noise cancelation.

1.3 Combined EEG and NIRS

Each of the mentioned techniques measure different and independent parameters of brain activity, however the relationship between them is not well understood. EEG is a well-established technique that measures the electrical activity along the surface of the scalp. We can monitor electrical activity in the range of milliseconds, providing an excellent temporal resolution. However the spatial resolution is relatively low. NIRS measures local concentration changes of HbO and HbR. The spatial resolution depends on distance between the source-detector pairs, hence in the order of centimeters. In contrast with EEG which is relatively ineffective at

differentiating specific regions, NIRS provides a more precise localization of brain activity and a higher spatial resolution. Both techniques provide a temporal resolution in the order of milliseconds, with EEG providing a slightly higher resolution.

Thus EEG and NIRS provide complementary information on electrical and local hemodynamic brain activity, providing more comprehensive information. This multimodal analysis technique has many advantages in research and clinical settings. Since there is no observed interference with the combination of these two methods, it can be used in long-term monitoring sessions which require a compact and portable device (Wallois et al., 2010). Areas of application include prolonged monitoring in epilepsy, brain machine interface and neuropsychological studies. In addition, both techniques are portable and relatively inexpensive.

1.4 Other NIRS system factors

1.4.1 Helmet

There are many solutions for imaging the cortex, some devices use a patch which can be placed on the region of interest (for example frontal cortex) while other systems provide a full helmet that can be placed on the head. A greater number of channels results in a better spatial resolution, but we must also consider the physical space available for the attachment of the optodes, as well as the channel cross talk and interference. Depending on the region of interest for imaging, a patch may be sufficient but full cortex monitoring requires a helmet. Figure 1.6 shows an example of a patch and helmet design.

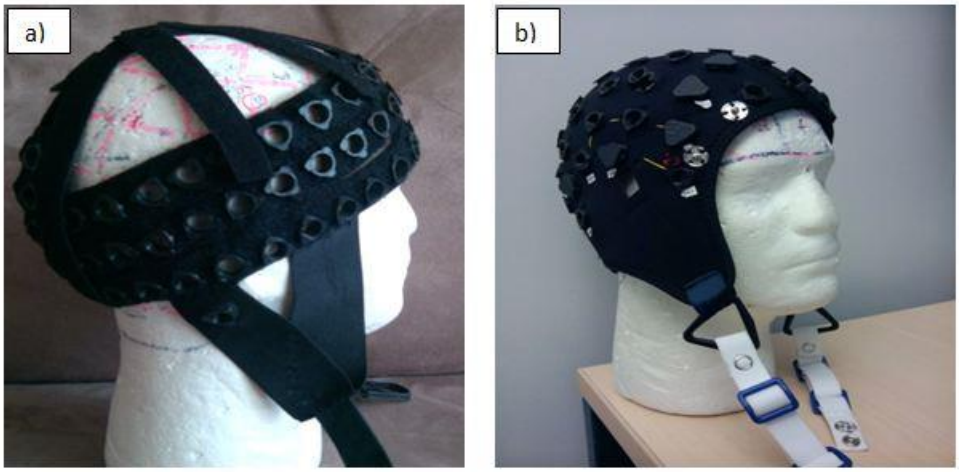


Figure 1.6: Different NIRS head caps. a) patch b) helmet. Images obtained from (Kassab, 2014)

The optodes must be in good contact with the scalp to achieve a good SNR and any hair must be cleared from the path. Another important factor is ambient light that can interfere with the injected light. Thus it is optimal for the helmet (or patch) to be made of an opaque material to minimize this effect. To accommodate various head sizes, the helmet material should have some degree of flexibility. Ideally it should fit the head tight enough to provide a stable connection to avoid movement artifacts without reducing the comfort. In addition, many tests involve monitoring sessions of long durations, thus it should be comfortable and breathable to wear for long periods of time.

1.4.2 Wavelengths and light source

As seen in equation 1.4, in NIRS applications the MBLL equations are solved at two or more discrete wavelengths (Delpy et al., 1988; Kocsis et al., 2006). At any wavelength, the emitted light must be selected from the near-infrared region of the spectrum. As mentioned before, in this optical window most of the emitted light can be detected by the detectors, with minimum absorption by tissue. Thus it is optimal to have maximum illumination to maximize the detected light at the detector and be able to increase the emitter-detector separation. In a portable solution, it is important to choose a light source with minimum weight, size and power consumption. The two most common sources of light used in NIRS applications are laser diodes and LEDs (Scholkmann et al., 2014).

LEDs emit incoherent light whereas laser diodes allow for coherent light emission. The operating range of laser diodes (< 1 nm) is narrower with low divergence compared to LEDs (~35 nm). However adjusting the LED light intensity is easier. While laser diodes can be easily used with fiber optics, their size is bigger compared to LEDs. There are also limiting factors such as tissue heating, patient safety and noise at the detector with laser diodes.

If the illumination power is too high it may cause tissue heating which can be hazardous for the subject and cause discomfort, as well as affecting the measurements (Bozkurt et al., 2005; Ito et al., 2000). In addition, the eye safety of the subject and the person performing the test may also be at risk (Sliney et al., 2005). Lasers can be more dangerous since they offer a higher power illumination compared to LEDs and it is important to follow the safety guidelines to limit the exposure. Another factor is light source fluctuations, which ultimately affects the signal quality at the detector (Zhang et al., 2005).Thus an optimal light source will emit light invariantly over the whole period, avoiding drift effects. This should be considered in the circuit design to minimize the effect.

Since LEDs are available in a wide range of colors, the selection of wavelengths is much more flexible compared to laser diodes. In addition, the cost of LEDs is lower compared to laser diodes. In our system we have used LEDs to provide a cost-effective and portable design.

1.5 Software overview

We can conclude from the previous section that both NIRS and EEG are feasible solutions for real-time systems, due to their high temporal resolution. Thus it is possible to design a software system that handles the acquired signals from the prototype and after signal processing, displays the relevant information to the user. It is important to note that the choice of hardware has been made before the software system and various configurations and settings are already fixed and cannot be changed. We have taken account of the constraints and hardware capabilities. In the next section, we will discuss the various stages involved in software design, from discovering and documenting the requirements to system modeling techniques and implementation.

1.5.1 Software requirements

Software requirements elicitation is an important step in any software engineering project. It is the process of defining the operations the system must perform, as well as identifying the constraints and bottle necks. An incomplete set of requirements may lead to later problems in the design process and they must satisfy all stakeholders. In our project, the end user is the clinical neurologist who will be performing tests and monitoring sessions. Requirements engineering is the set of activities performed to identify the needs, as well as analysis, documentation and validation. The output of this stage is a requirements document, which clearly specifies all the details (Ian, 2011; Lee, 2013).

Requirements can be organized into two main categories: functional and non-functional. Functional requirements are the main functions and tasks the system must fully perform. These are usually clear and depend on the end-users and the type of software developed. User requirements are more abstract, whereas system requirements are more detailed (inputs/outputs of each system is indicated).

Non-functional requirements are the constraints placed on the results of the functional requirements (for example the speed of performing a particular task). In other words, it represents the quality of a system, in terms of accuracy, speed and performance. This set of requirements is not as straight forward and depends on several factors, including the operating environment, the end user, budget constraints and interaction with other hardware/software systems. Regardless of the requirement, it must be clear and understandable to both teams. Goals that are defined too general, usually by the end user, can leave room for personal interpretation of the developer.

Requirements engineering process

The four main activates in the requirements engineering process are:

1. Feasibility study
2. Requirements elicitation
3. Requirements specification
4. Requirements validation

Figure 1.7: Requirements engineering cycle

As seen in figure 1.7, the activities are performed in an iterative process. Each step will be briefly discussed.

Feasibility Study:

The feasibility study for the overall development of the proposed software system has already been done in the initial development phase. However the specific details and specifications, such as choice of development platform, have been decided in the scope of this project.

Requirements Elicitation:

In the requirements elicitation phase, the required tasks/features of the system are derived, based on the shortcomings of existing systems and user feedback. It is common to develop a basic prototype of the specified system in order to better clarify the primary requirements and design phases needed to complete the project. This will ensure that the end-user and developer needs are consistent, complete and unambiguous. This can also include questionnaires, observations and direct feedback from the users. While one requirement may seem intuitive and essential to the user, it is not always easy or efficient to implement.

Common problems that occur during the requirements elicitation phase can be categorized into three main groups: problems of scope, mutual understanding and the changing nature of requirements. Problems of scope arise when the requirements are too broad to establish boundary conditions, or are over-specified and contain unnecessary details. This may lead to ambiguous

and unnecessary conditions that do not reflect the main user needs and are not implementable given the project time constraints. One of the factors that affect our project scope is the existing hardware architecture already in place. While this does not impose any significant limitations but the software must be designed with this factor in mind. Problems from misunderstandings can be due to the various backgrounds of the two teams; common knowledge to one group is not necessarily obvious to the other group. In our project, the end users come from a medical domain, whereas the development team has limited knowledge of the application domain. The user may not understand the limitations involved in adding a new feature or not have full appreciation for the effort required to make what may seem a minor modification. In our project, some of the basic requirements had already been defined through the development of the initial user interface (more details in section 3.4). Many more have been defined and implemented during this project. Another common problem is the volatile nature of requirements. While new changes can be integrated into the system, they must also remain consistent with the program architecture and existing modules. In our software cycle, the program development was completely iterative, meaning that after each set of small changes the prototype was examined by the end user to ensure there are no mistaken assumptions or misinterpreted information.

Requirements Specification:

The next step is the requirements specification. Once all the requirements have been extracted, they need to be documented and structured to clarify all the details. In this stage we also demonstrate that the system requirements have been defined to accurately meet the customer needs.

Requirements Validation:

The final activity is requirements validation. At this point the specified requirements are examined to ensure they are realistic and achievable within the desired time frame. Prioritization can be useful in determining what the most important features are and what details can be implemented in the next product iteration. The end-users can also experiment with the system prototype to see if it satisfies their needs. After defining all the requirements, a software requirements document is written to clearly identify the agreed terms and conditions. Once the requirements have been defined, we can visualize the system in various graphical models. Each

model focuses on a particular aspect of the system, such as the context, interaction structure or timing.

UML (unified modeling language) is a standardized way of modeling software and non-software systems. By using the models we can properly visualize and describe the system in a uniform manner that can be easily understood and is comparable. The current version, UML 2.0, consists of 13 diagram types, which are divided into structure, behavior and interaction diagrams (Ian, 2011). We will only discuss the relevant cases to our design.

Use case diagram:

Use case diagrams are used to develop a graphical and higher-level representation of the system. Each project can be organized into a system, and one or more actors that interact with it. An actor makes a decision, but is not necessarily a human; it can be a computer program or hardware system.

Each use case identifies the interactions between an actor and the system in various scenarios. The system details are not defined, and it is treated as a black box. These scenarios help both teams to better understand how the system functions and validate it by relating it to real application examples. While use cases provide a static and simplified view of the system, they do not capture all the details, such as the performance and timing of a system.

State diagram:

A state diagram describes the state of each object at any given time, as well as the event that triggers a state change and the resulting action. Each state transition is unique. Overall, it shows the end result of the systems behavior.

Activity diagram:

Activity diagrams are similar to state diagrams, in addition to providing information on the flow of events. It shows the dependency of different states, meaning which process needs to be completed before transitioning to the next stage, as well as parallel activities.

1.6 Conclusion

We have discussed the major advantages of NIRS and EEG as non-invasive imaging technologies that allow us to monitor cortical activates in non-clinical settings while performing

daily routines. Due to the high temporal and spatial resolution that both techniques provide, the development of a software system for real time viewing and long term monitoring of brain activity is quite promising.

The theoretical aspects of NIRS and EEG were presented, as well as the benefits and limitations of a dual mode device. In order to define the system requirements, we must have clear and effective communication with the end user to capture all the details required. By providing an early system prototype and various system models, we can identify the requirements early on in the process. In the next chapter a literature review of existing NIRS-EEG imaging systems with similar functionality will be presented. Chapter 3 will discuss the details of the software system, design methods and the software architecture. In chapter 4 we will present and discuss the results obtained using our system, as well as system metrics and measures. The last chapter will conclude with a general discussion of the work accomplished in this project, the limitations and future directions of development.

CHAPTER 2 LITERATURE REVIEW

In recent years the real-time monitoring of NIRS and EEG data has gained attention as a practical tool for brain-computer interfaces (BCIs), various brain disorders including epileptic patients as well as stroke and rehabilitation units (Gallagher et al., 2008; Gallegos-Ayala et al., 2014; Haginoya et al., 2002; Irani et al., 2007; Li et al., 2012; Matthews et al., 2008; Nguyen et al., 2012, 2013; Osharina et al., 2010; Sokol et al., 2000; Soraghan et al., 2008; Strangman et al., 2006; Terborg et al., 2009; Wang et al., 2008; Watanabe et al., 2002). In order to better understand the advantages and disadvantages of the existing strategies, as well as to define the need for a new system, this chapter aims to provide an overview of the most recently developed NIRS/EEG systems with real time data monitoring. There are two aspects to consider: the hardware component and the software platform. As the hardware system becomes more complex, the software platform must also be updated to reflect these changes. However it must still be robust, easy to use and have a modular design to facilitate the integration of new components and modifications. We will consider the system from two perspectives: one to compare our system as a whole with similar systems in terms of hardware specifications, and one that will focus on the software features and limitations that were considered in our design.

In the first section we will consider five NIRS/EEG devices. The following systems are developed by:

1. “Safaie et al.” (Safaie et al., 2013)
2. “Leamy et al.” (Leamy et al., 2011)
3. “Hebden et al.” (Hebden et al., 2012)

and two commercial devices:

4. LABNIRS developed by Neurospec (Shimadzu Corporation, 2014)
5. Asalab developed by Ant-neuro (Antneuro, 2014)

In the second section we will look at the user interface and software aspects of two systems. The first one is an earlier version of the brain imaging device developed by our group presented in Lareau’s work (Lareau et al., 2011). The second solution is the software system described in (Safaie et al., 2013). In section 2.2 we will consider a number of commercial and non-

commercially available NIRS acquisition systems with similar features and do a comparison. Finally in section 2.3 we will discuss the software design concepts and proposed architectures.

2.1 NIRS-EEG acquisition systems

2.1.1 Hardware comparison

The characteristic of five NIRS-EEG devices is summarized in Table 2.1. All solutions support dual mode, both NIRS and EEG, but they are not necessarily an integrated device.

Table 2.1: Characteristics of NIRS-EEG devices

Characteristics	[1]	[2]	[3]	[4]	[5]
Integrated NIRS and EEG	yes	no	no	no	no
Portable	yes	no	no	no (cables extend up to 13 meters)	EEG is briefcase-sized, NIRS depends
#Emitters/Detectors	32/4	7/3	32/16	Maximum of 142 channels	8/8
#EEG	16	3	3	128	32-128
NIRS Sampling rate	8	25	10	166	Connects to NIRx device[1]
EEG Sampling rate	1024	2048	0.3-70	Connects to BioSemi EEG device[2] (up to 16 kHz)	max 10kHz
Data transmission	Bluetooth	USB	USB	USB	USB
Head coverage	patch	patch	helmet	patch/helmet	helmet for EEG, patch for NIRS

The system developed by (Safaie et al., 2013) is a portable and wireless bimodal NIRS-EEG acquisition system with an online interactive environment for real time monitoring. The hardware design of the system has similar characteristics to ours, with a few differences that are discussed.

[1] http://nirx.net/uploads/Brochures/NIRSport%20Brochure.pdf.

[2] http://www.biosemi.com/faq/adjust_samplerate.htm

Their system has 32 emitters, 4 detectors and 16 EEG channels. The limited number of detectors greatly reduces the number of available channels and limits the brain area that can be imaged. In comparison with our system, we are able to couple 32 emitters with a selection of the 32 detectors (which can be coupled with 4 or 8 detectors), providing a total of 128 NIRS channels. Our NIRS channels are sampled at a rate of 20 Hz, which is higher than the reported system at 8 Hz. The primary benefit of more channels and a faster sampling rate is improved spatial resolution. The proposed system supports only 16 EEG (32 EEG channels for our system) at a sampling rate of 1024 Hz, while in our system it is at 320 Hz.

Another important difference that affects the number of emitter/detector pairs is the helmet design. In the mentioned design the optodes are mounted onto a flexible material that is fixed to the patients head by a Velcro strap. This patch can only cover one area of the cortex, and as a result cannot report the cerebral activity from all regions of the brain. In our system, we have designed a full helmet that covers the entire adult head, with the optodes placed at equidistant sockets. The helmet has been designed with both stability and flexibility in mind, and to allow easy mounting of the optodes.

Our system supports up to 32 EEG electrodes. Depending on the helmet being used, these locations are either fixed based on the 10-20 montage standard, or the electrodes can be inserted at desired locations through the helmet using electrolyte gel. We have also included an auxiliary analog channel that has been proven to be very practical and convenient during experiments. While there is no designated purpose for this channel, it can be easily connected to other physiological and respiration instruments devices, such as an electrocardiogram or pulse oximeter. This provides an easy mode of displaying all data from the various sources on a single display.

The second system by (Leamy et al., 2011) is a combination of separate NIRS and EEG devices which yields a non-portable solution. In addition it offers a low number of NIRS (3 detectors, 3 emitters) and EEG (7) channels. The acquired data is transferred to the host computer via cable connection. The imaging is also limited to a specific part of the cortex, since the optodes are fixed onto a patch.

The third system developed by (Hebden et al., 2012) is a combination of two separate devices, which offers a high number of NIRS channels (32 emitters and 16 detectors). However

there is no wireless data transmission and due to its size it is not portable. The optodes are fixed onto a helmet which covers the entire cortex. The NIRS sampling rate is 10 Hz and uses the frequency domain technology. The EEG system records at a frequency range of 0.3 – 70 Hz, using 3 EEG channels.

The commercial system developed by NeuroSpec, which is called LABNIRS, is a combination of two separate NIRS and EEG devices (Shimadzu Corporation, 2014). While the system is described to be useful in various tasks involving patient movement, the monitoring device is stationary (due to its size), and only connected by fibres to the optodes (which can be extended up to 13 meters). Thus the portability of the system is arguable.

The system developed by Ant-neuro offers a similar solution for NIRS-EEG monitoring using separate devices. The system is designed to interface with NIRx system for acquiring NIRS signals (Antneuro, 2014). It is not mentioned which version of NIRx devices is used, however only one version (NIRSport) is a portable solution, which is limited to the frontal cortex and limited number of emitters and detectors.

2.1.2 Software comparison

In this section we will look at the software aspects and features of two similar systems. The first system is the initial GUI for the portable NIRS-EEG system developed by Lareau (Lareau et al., 2011) which is the base platform for the developments made in this thesis. The initial prototype was created in a reduced channel count version (8 emitters, 8 detectors and 8 EEG channels) with a software module for displaying the results. We will focus on the details of the user interface design and the enhancements/additions that have been made in our system. The GUI in the mentioned system is developed entirely in Matlab, and allows limited user control of the acquisition parameters while transmitting the data in real time via USB to a PC. Our system is developed mainly in LabVIEW with some processing in Matlab. The advantages of using LabVIEW as a centered workstation to control all aspects of the instrument is outlined in (Soraghan et al., 2008). We still have the option of including user defined signal processing functions (written in C or C++) which can be easily integrated via DLL (Dynamic Link Library). LabVIEW supports direct Matlab code, thus we were able to minimize development and test time by taking advantage of existing modules. The developer can monitor the required time for each particular task and identify the bottlenecks of the system in the real time processing chain. In

particular, we have reused previously developed C++ libraries for the serial data communication. In Lareau's system the raw data is received and recorded by the PC, while no additional processing is done online. HomER, which is a specific Matlab toolbox dedicated to the processing of optical functional brain data, was used to perform further processing on the raw signals, and to calculate oxy and deoxy-hemoglobin values and display a 2D image reconstruction of the concentration changes (Huppert et al., 2009). The following is the shortcomings of the user interface which we aim to overcome in our current system:

- Very basic user interface
- Limited parameter adjustments
- Offline signal processing
- Configuration parameters are not adjustable during the test

In this work, we have made the following improvements and included these additional features:

1. Real time parameter adjustment without interruption during a recording session
2. Real time display of HbO and HbR values
3. 2D mapping of the oxygen concentration changes
4. NIRS signal quality monitoring during sessions
5. Ability to restart an acquisition using previously set configurations
6. Adjustable window frame (scale) for both NIRS and EEG graphs
7. Bipolar and referential EEG montage
8. Individual show/hide setting for each graph
9. Functional wireless communication between control module and PC
10. Wireless communication for data transfer between two or more computers
11. Auto calibration feature that quickly adjusts the individual emitter light intensity to achieve best results based on the specific test settings
12. Battery level indicator for the control module
13. Convenient raw data storage (.tdms) and conversion to HomER and Matlab compatible format (.nirs and .mat)
14. Much more robust and flexible graphical user interface
15. Modular design and organization of the code that facilitates maintenance and future development

16. Grouping of various functions into libraries which can be reused and configured as needed

With these additional features, we are fully capable of monitoring, configuring, recording and processing data in real-time.

In the system designed by (Safaie et al., 2013) they have developed a dedicated user interface. The auto-calibration function is present in both systems. This allows the user to achieve the best illumination intensity for each individual emitter, ensuring a good SNR. One common problem in NIRS systems is poor connection between the NIRS optodes and the scalp due to patient movement or hair in the light path. To reduce undesired effects and to ensure the quality of connection, we have also included a software option to monitor the status of all channels. The "analyze" function of our system provides a quick update on the status of all NIRS channels, and states whether they are saturated (from ambient light) or if there is cross-talk between channels. The status can be regularly checked at any time during a test to validate signal quality. This feature does not exist in the mentioned system. They have described a similar feature for EEG signals. This includes a dedicated circuit for determining the electrode-skin impedance to identify any bad connections. Due to hardware limitations, our design does not include this feature; however an impedance meter can be quickly used to verify the skin-electrode connection quality prior to an acquisition. Since the EEG electrodes are securely fixed onto the patients head using electrolyte gel, the contact quality remains relatively stable throughout testing.

Another important detail to consider in all monitoring systems is motion artifact and noise reduction. Our present system uses two external 3D accelerometers, one that is placed on the helmet and the other can be freely placed on a desired location depending on the activity performed. The system reported by (Safaie et al., 2013) only uses one 3D accelerometer. Our current design displays and records the information along the three axes of the accelerometers, but it is not yet applied in motion artifact removal of the NIRS or EEG signals. It appears that the mentioned system does not make any corrections for movement artifacts either. Similar to our system, they simply display the results to help distinguish between normal and abnormal motion artifacts. This source of information is still practical for analyzing the results offline. The data from the accelerometer will be soon integrated as a function in the future version of our system.

Both systems have been designed with long term monitoring in mind. Thus it is important that the system operates for a relatively long period of time without requiring battery replacement. The system developed in (Safaie et al., 2013) can perform acquisitions for up to 5 hours. While this may be sufficient in some applications, it is not suited for all tests. Our system can monitor and record for up to 24 hours, hence providing a much longer testing period.

The mentioned system can be used to simultaneously connect to up to 7 prototypes placed on different locations on the body. This can be used to monitor cerebral as well other muscle tissue activity, creating a network of devices. Our current system supports one prototype connected to a PC for monitoring, but this can be easily extended to include more acquisition models connected to the same PC to create a network of devices if needed.

Another design feature is the choice of development platform. We have chosen to use LabVIEW and Matlab, while the system by (Safaie et al., 2013) has been developed entirely in Matlab. It is difficult to determine which has a better performance; LabVIEW provides easy integration of software and hardware components through a graphical programming technique, while Matlab is solely based on coding. In addition, Matlab code can be directly inserted into LabVIEW programs, which has allowed us to take advantage of previously developed Matlab functions and include them in our design. Thus it appears that both platforms are well suited for this application, and it is more a design decision of the developer.

In terms of hemodynamic value calculations, we have used MBLL for calculating oxygen concentration values (Delpy et al., 1988). The mentioned system has used an additional method, spatial-resolved spectroscopy (SRS), to calculate the tissue oxygenation index (TOI) as well.

2.2 Portable NIRS devices

While there are a limited number of portable NIRS-EEG devices, there are many portable devices dedicated solely to NIRS, commercial and non-commercial ones. We will discuss the details of a similar system to ours, developed by (Hemmati et al., 2012). Based on a recent review by (Scholkmann et al., 2014) a number of existing commercial NIRS acquisition devices have been analyzed and compared in terms of features and specifications.

The system developed by (Hemmati et al., 2012) is dedicated solely to CW-NIRS monitoring and provides 4 emitters and 10 detectors, for a total of 16 NIRS channels. They

process the retrieved data in real time and display oxygen concentration values. However there is no indication of a real-time topographical image reconstruction, or real-time configuration of the parameters. In addition, the data is transferred via cable connection to the processing computer. This will limit the patient's mobility to the close proximity of the monitoring computer, which can be inconvenient in long term testing. Also in the mentioned system the stability and quality of the optode connection is only maintained by ensuring it holds an orthogonal orientation to the skin surface. Thus there is no means of monitoring the connection through the software during long term acquisitions. In addition, in the mentioned system they only monitor the prefrontal cortex activity, as a result the attachment of the probes to the skin surface is much easier to maintain. Similar to our system, they have used the MBLL to calculate oxygen concentration values.

To the best of our knowledge, the mentioned system does not have a dedicated software system for their control unit. The software platform has not been clearly detailed, and only the final results are presented. The limitation of such a system is that it may be difficult to achieve the best possible results for all tests, since they require some level of customization based on the particular task and patient. While our system provides a higher channel count and additional hardware add-ons, such as an accelerometer, it is flexible and allows the user to easily customize the test settings based on individuals and the application. This feature makes our system usable across various applications. Table 2.2 shows a list of commercial NIRS monitoring devices characteristics.

Table 2.2: Commercial NIRS devices characteristics. Adapted from (Scholkmann et al., 2014)

Device	**Sampling frequency**	**Emitters**	**Detectors**	**Portability**	**Wireless**
D1 – Oxymon MkIII	250	32	16	no	yes
D2 – PortaLite	50	3	1	yes	yes
D3 – fNIR1100	2	Up to 4	Up to 10	no	yes
D4 – fNIR1100w	2	1	Up to 4	yes	yes
D5 – ETG 4000	10	18	8	no	yes
D6 – ETG 7100	10	40	40	no	yes
D7 – WOT	5	8	8	yes	yes
D8 – Genie	5.02	4-16	8-32	yes	yes
D9 – NIRScout	6.25-62.5	8 or 16	4-24	no	yes
D10 – NIRScoutX	6.25-62.5	48	32	no	yes
D11 – NIRSport	6.25-62.5	8	8	yes	yes
D12 – Brainsight NIRS	100	4-16	8-32	no	yes
D13 – FOIRE-3000	7.5-40	4-16	4-16	no	yes
D14 – OEG-SpO2	1.52 or 12.2	6	6	yes	no
D15 – CW6	10-50	4-48	8-32	no	yes
D16 – UCL Optical Topography System	10-160	16	16	no	yes
D17 – Imagent	16-60	16 or 32	4 or 8	no	yes

In order to provide a meaningful comparison in regards to our work, we will categorize the devices based on the following priority list:

1. Portability
2. High-channel count
3. Wireless monitoring

Of the devices listed above, many are small enough to be considered a portable solution, meaning that the patient can wear the device and it is battery operated. Devices D2, D4, D7, D8, D11 and D14 meet this criterion. In terms of channel count, most devices have a relatively low number of emitter-detector pairs, which limits its applications. D8 which is developed by Genie offers a greater number of emitter-detector pairs (4-16 emitters and up to 32 detectors). However the temporal resolution is significantly lower than our device, and is at 5.02 Hz.

The third criterion is wireless monitoring. This is important since we do not want to reduce the patient's mobility by using a wired connection in long term monitoring. In addition, it allows us to use the prototype in outdoor settings where the patient undergoes gait or similar protocols. Of the six devices that were considered portable, all had wireless data transmission to a host PC except for D14 (the data is recorded in the device). Although the mentioned systems provide information from both NIRS and EEG sources, none of them provide an integrated portable solution with a high channel count.

2.3 Software architecture models

The larger a software project is, the more important it is to have well-organized code. Otherwise it will become difficult to continue development and make any minor changes without creating more problems. If the development is already complete, it is even more time consuming to ensure the maintenance and evolution of such software, and can be very challenging to change. This section will focus on the architectural design of the Imaginc software and we will introduce the supporting software tools that are available and can be used in our design. It is important to mention that we are focused at the architectural level of the application. We need to consider the software and components as a whole, while satisfying the requirements. Some design decisions have been made during the initial phases of the project development (by previous team members who carried out the implementation), and we have made the future design choices based on those.

The choice of software architecture is partially dependent on the perspective of the designer. It also depends on the specific constraints and objectives of the project. In our case, there are certain hardware specifications, as well as end-user requirements. The latter is acquired by interaction with the end user, as well as user feedback from trial versions of the software. In our work regular feedback has been an integral part from the beginning with the goal of

developing a customized and flexible program. More than one architectural model may satisfy the given set of requirements, but the best must be chosen. Different quality attributes are important to the design team and the end-user. The designer is more focused on maintainability and reusability whereas the end-user expects the application to be intuitive, easy to use and reliable. In the next section, we will present the architectural choices that were considered in the design of the Imaginc software.

2.3.1 Client-Server architecture

The client-server architecture is a generic architecture which is widely used. It is composed of three main elements: the server, one or many clients, and the network on which they operate. The client passes requests to the server, and the server responds by providing the requested information (after validating the request) through the network. This request can be for data, a file, a calculation or data processing. The advantage of this architecture is to better employ available computing resources and to share data processing loads. In our application, there will be two computers communicating over a local network. Thus one will be the host and the other the client. The acquired data is sent from the first computer to the client in order to be processed and displayed.

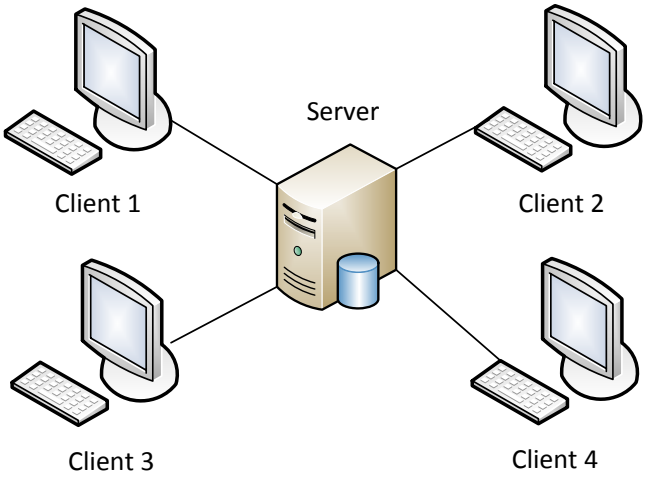


Figure 2.1: Client-server architecture

As seen in figure 2.1, we can add more clients to the network and each client can also serve as the server to another machine. By expanding this network, we can create different types of

architectures. The advantage of this approach is that the system is broken up into manageable components, and the control and data persistence mechanisms are clear.

2.3.2 Layered Architecture

By extending the client-server architecture by more layers, we can achieve a multi-tier architecture. The most widely used is the 3-layered architecture. This model creates flexible and reusable applications by separating the application into distinct layers. By layering we can simplify the complexity of the overall software where each layer N relies on services from layer N-1. Each layer only has access to the layer immediately below it, and not any of the previous ones. Figure 2.2 shows a 3- layer architecture.

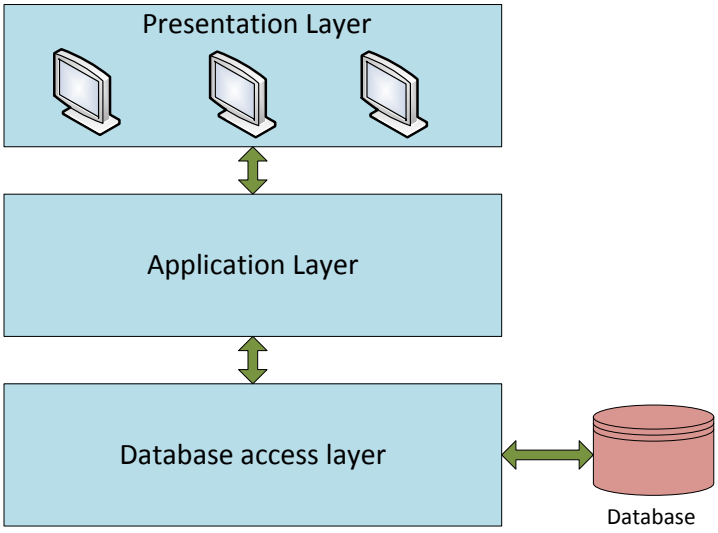


Figure 2.2: Three tier architecture model

The 3–layer architecture is composed of three layers:

- Presentation layer: The top level layer, which is responsible for displaying information to the user.
- Logical/application layer: This layer processes incoming commands from the user and evaluates and performs calculations. It also retrieves data from the data layer as needed.
- Data layer: Contains the data base. It is responsible for querying a database and fetching information. This layer is not directly accessible to the presentation layer and must be accessed through the logical layer.

The advantages of this architecture are that the complexity of each layer is hidden and any change in layer N does not affect any of the other layers, similar to the object oriented methodology. Thus changes and modifications can be easily made and supports an increasing level of abstraction in the design as well as reusability. Our system consists of two main layers, the hardware communication and the software layer. Each layer itself is composed of many sub-layers, and information is passed between them. However some of the data should not be modified by any other layers and should be protected at all times. This is mainly seen in the communication protocol layer for sending and receiving data to and from the prototype. Another advantage of separate layers is that the software development can be done independent from the other system configurations and thus simplifying the task. In a real life situation, more than one person will be working on the project and this allows for separation of tasks.

2.3.3 Model-View-Controller architecture

The Model-View-Controller (MVC) architecture is made up of three main components: the model, the view and the controller. The goal is to have separate components that interact with each other. Figure 2.3 shows this architecture and the data flow between them.

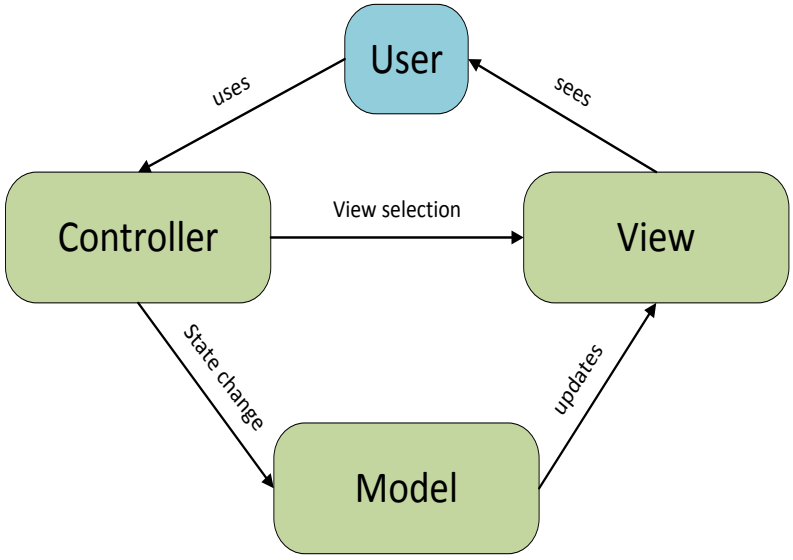


Figure 2.3: MVC model components and interactions

- View: is the interface that interacts with the user (the GUI). The view can call the model to request information.

- Controller: acts as an interface between the user and the model. It sends commands to the model to update the information. It also interacts with the view to update the GUI.
- Model: notifies both the view and controller of an event (change in status). This allows the view to update the information displayed to the user and for the control to update the list of commands.

In this architecture there is one central model, with many viewers. Each view has an associated controller that handles updates made through the user of the view and any changes made to the model are propagated to all the views.

2.4 Conclusion

In this chapter we examined various integrated and non-integrated NIRS-EEG devices, some of which were portable. In addition we also compared several portable and non-portable NIRS devices in terms of various characteristics and features. There are different existing software platforms for real-time monitoring and processing of NIRS-EEG data. The advantages and disadvantages of each system were discussed. Overall we can conclude that it is important to have a modular and customizable software system in order to use the system across a wide range of applications. The software design is also an important aspect linking the gap between the end-user and the prototype. While there may be several potential features implemented in the hardware component, they are only practical and beneficial if they can be properly displayed and configured in a way that is straight forward and clear to the user. For a system with a large complexity such as ours, there is more than one architectural design pattern required in order to satisfy the requirements. Each individual module requires a specific architecture to carry out the task in the most efficient manner possible. While there may be no right or wrong solution, it is up to the developer to apply the most feasible one. We have used the basic design principles mentioned and expanded it to create a more complex yet manageable structure. The next chapter will look at the design details and implementation of the software system.

CHAPTER 3 SOFTWARE DESIGN AND IMPLEMENTATION

This chapter presents the software design and implementation of the Imaginc software. We will first discuss the system requirements and present different models of the system. This is followed by the implementation details and design decisions of the software architecture and user interface.

3.1 Feasibility

The initial version of the software system designed for the Imaginc prototype has been developed in Matlab. The limited and basic functionality of the user interface was not sufficient to meet the demand for more complex and real-time operations. As the hardware system has evolved to include more channels and features, the software system also needed to be updated to reflect these changes. We will specify the required changes and details to support the new system.

3.2 System requirements

Based on the limitations of the initial software system and to overcome these, we design a new system with the following constraints in mind:

- The need to display results in real-time
- The ability to change various recording parameters throughout the acquisition without interruption
- Reliability of the acquisition system for long term monitoring (24 hours)
- Simple and user-friendly interface that is intuitive to use
- Modular design of the software that allows for easy modification/reuse

As mentioned previously, the hardware platform has already been configured. In order to facilitate the integration of the software and hardware components, we chose LabVIEW and Matlab. LabVIEW is a development platform by National Instruments. This graphical programming and control software make it an ideal candidate for the robust and modular design of our system. Matlab provides an environment for programming and visualization. We have taken advantage of both in our design.

This project aims to provide the software system to support the brain imaging device developed by our group. While the hardware component is functional and capable of recording data, this information needs to be displayed to the user to allow for continuous monitoring of the results. Besides performance, it needs to serve as a tool for quick and easy analysis of the incoming data for clinical settings. We also need to be able to adjust certain acquisition parameters to obtain optimum results based on individual test settings. While the user requirements of our system are mostly determined, user feedback is still valuable. In the case of our project it is important to provide the end user with a prototype of the system to be able to gather the user feedback and include it in the final version of the product.

The software architecture for our system was designed with the following constraints and objectives in mind:

1. Manage complex and real time interactions with the hardware device
2. Provide a timely response to operations and user input, as well displaying results
3. Reuse existing libraries as much as possible
4. Adapt easily to hardware changes (in future versions, when more channels or auxiliary data channels are added)
5. Use modular design to allow future development of the code despite changes in the development team, and maintain reusable code.
6. Develop a graphical user interface

With the mentioned constraints and objectives in mind, we have proposed the following solutions:

1. Isolate the interaction with the hardware (as a separate module that can be easily managed with limited knowledge of the rest of the system). This corresponds to requirement #1.
2. Software design must be able to evolve and be easily adaptable to new requirements as well as integrating previous modules. This will satisfy objectives #3, #4 and #5.
3. In order to have the best and most efficient performance, the critical and time consuming processes need to be in parallel and optimized for maximum performance. This will ensure real-time response and display of results, based on objective #2.

4. The end-users of the system will be clinicians and technicians, thus it is important to have an intuitive and easy to follow user interface that requires little staff training, as outlined in objective #6.

3.3 System models

State diagram

The state diagram of the system is presented in figure 3.1. The rounded rectangles are the system states, and the arrows represent the trigger event from one state to the next. As seen in the figure, the system operations are distributed among two computers: one which contains the main LabVIEW user interface and is the primary program for starting an acquisition. The data is sent to the second laptop using a TCP/IP connection for additional processing. The second system has

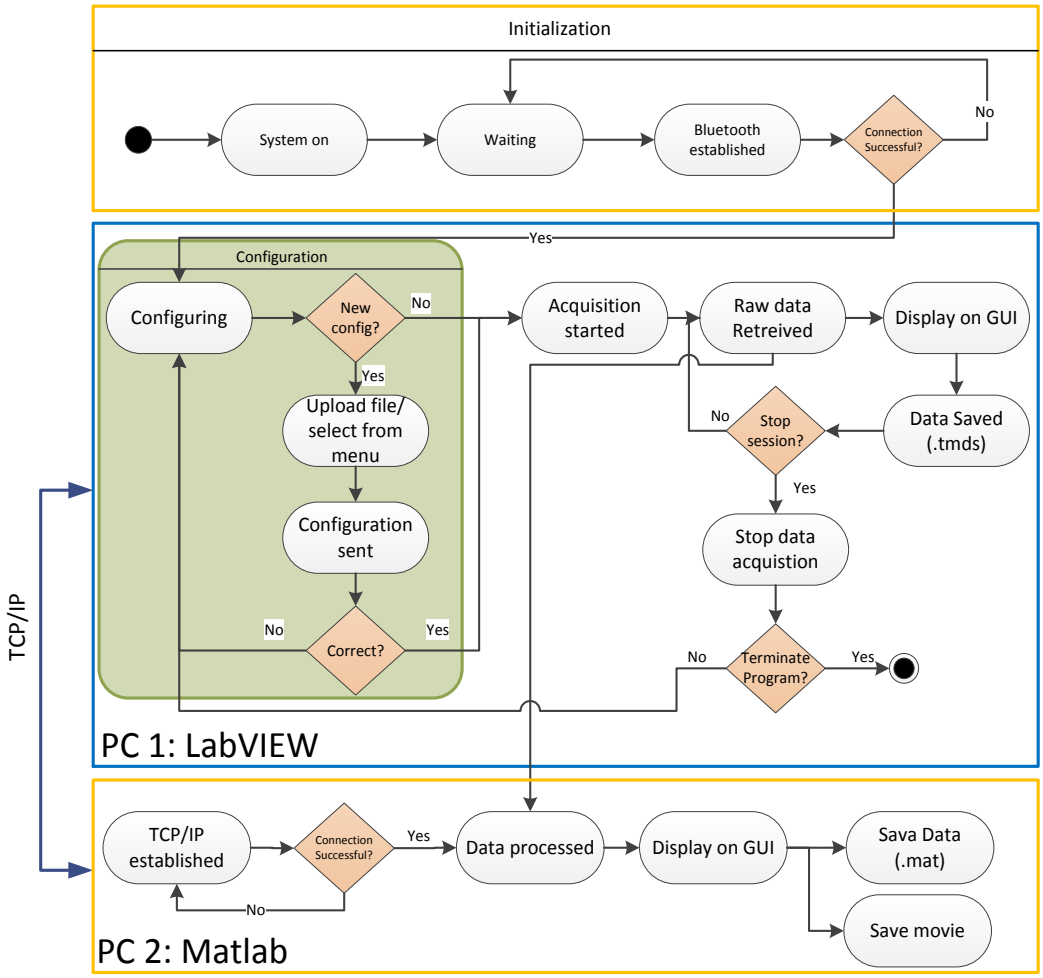


Figure 3.1: State diagram of IMAGINC system

a Matlab interface that displays additional features and image reconstruction. The purpose of this set up was to allow the clinician to remotely monitor the test from their office. The details of each interface will be discussed in section 3.5. The sequence of activities described in figure 3.1 is as follows:

1. The initial step requires turning on the hardware and ensuring sufficient battery level. The red LED on the side of the device will be turned on to notify the user. If the battery level is low, the LED will be flashing, indicating that the batteries need to be recharged. The battery level is also displayed as a percentage on the UI.
2. The computer "PC1" running the LabVIEW program will be in an idle "waiting" mode, until there has been a successful connection established with the prototype. This is in the form of either USB or Bluetooth.
3. Once the connection is verified, the user may start the program
4. The computer "PC2" running the Matlab program (on the same or different network, the default value is that both computers will be operating on the same network) will establish a connection with PC1. Upon successful connection the data will be transferred.
5. The user will set the various configuration parameters through the user interface (or by uploading a previous file)
6. If it is a new configuration file it must be sent to the prototype first. If not, the user can directly start the acquisition (details of the structure of the data and packets are discussed in section 3.4.3).
7. The acquisition is started and the NIRS, EEG, accelerometer, trigger and analog curves are displayed in the appropriate windows.
8. On "PC2", the received NIRS curves will be processed to calculate the oxy and deoxy hemoglobin values, as well as displaying a 2D image of the cerebral activity on the Matlab user interface. The sequence of changes is recorded as a movie.
9. All data is recorded throughout the session and saved to a file for later viewing and analysis.
10. The acquisition will continue to run as long as the user has not stopped the session. If the session is stopped, the program will return to the configuration setting window. If the user has selected to terminate the program, the program will exit.

The system will continue to run until the user has terminated the program. In order to guarantee the order of execution (for example the user cannot start the program without establishing a valid connection first) and prevent users from performing invalid commands there is a built-in dependency between the various tasks mentioned. This dependency is implemented in the program using controlled sequence of execution in the code, as well as disabling buttons that cannot be clicked. At any instance throughout the acquisition, the buttons and windows that may be selected will be clearly displayed as active, while the remaining ones will be grayed out and temporarily disabled. Another method of ensuring the correct order of execution is by dividing the main user interface into various windows, with only one window being displayed at a time. The program will automatically switch to the next window and guide the user to the next step. Further details will be discussed in section 3.5.

Use case diagram

The use case diagram of the system is shown in figure 3.2. Use case diagrams are a static and non-technical view of the system that models the system from the point of view of the actor. The details of each case are not shown; only the necessary information is included. Each use case describes a possible scenario and the interaction with the system. The actor can be the user of the system, or the prototype which communicates with the software module. Figure 3.2 includes 20 use cases and 2 actors. We can clearly see from the diagram what the overall system does, rather than how it functions

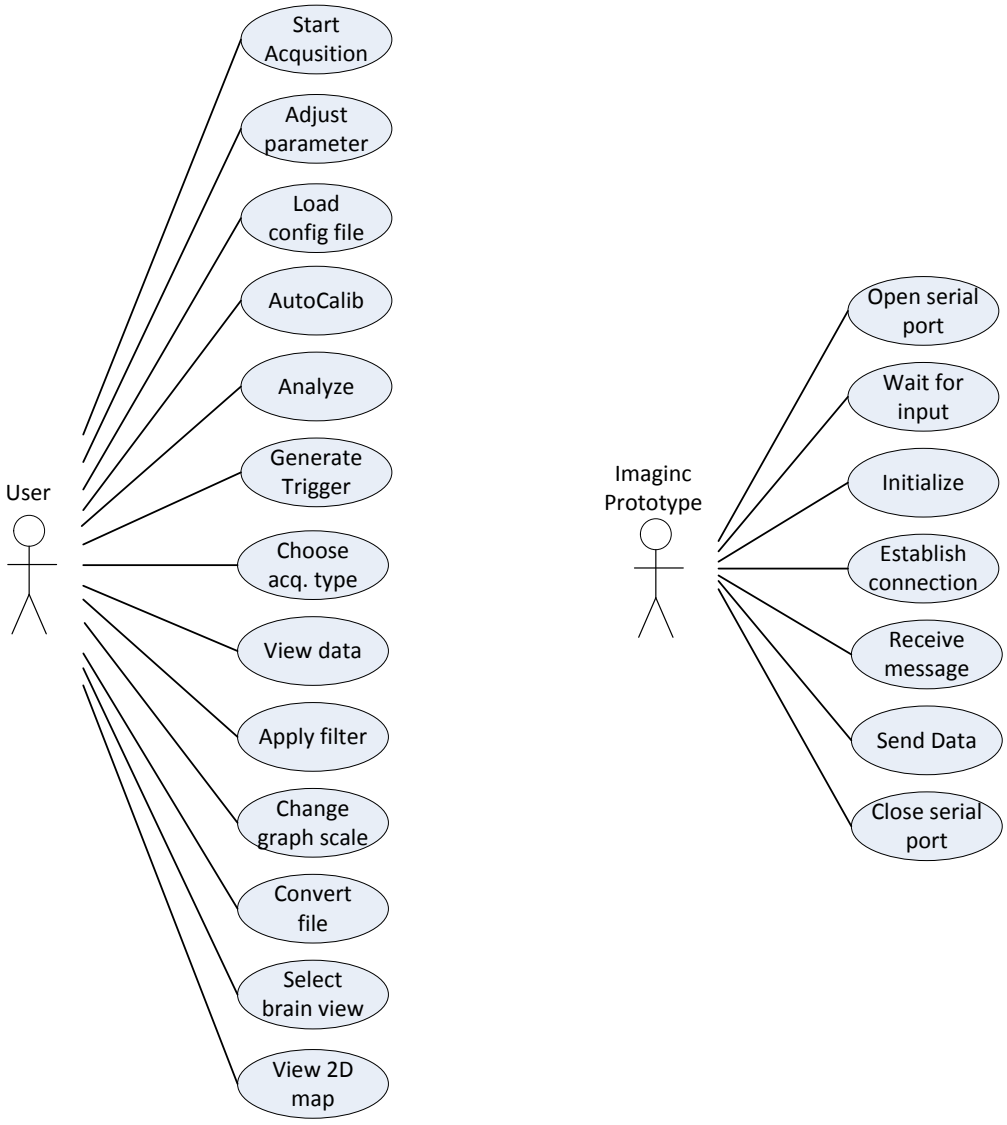


Figure 3.2: UML use case diagram for the IMAGINC system

.

3.4 Software architecture

The choice of software architecture highly depends on the requirements and the constraints that these requirements create. There are various architectural models that we can use as the basis for our program. The use of general architectural models can greatly help in the maintainability, consistency and reusability of a project while being customized to meet our specific needs. By using a component-based development approach, the system will be easily

adaptable to modifications through reusable components. In our case we use the general structure of software architecture, while adapting it to the specific requirements of our system.

The selected software development tool for the main program is LabVIEW. It is an effective graphical system design approach that makes integrating hardware and software systems relatively easy. It is powerful and flexible and allows for easy maintenance and upgrade. Programming is based on the concept of user defined and customized virtual instrumentation software, referred to as "VI"s. The complexity of low-level programming structures and integration with the hardware component becomes invisible to the user, and provides a level of abstraction. This helps the developer to focus more on the design of the system rather than low-level software and hardware issues. It provides a variety of data acquisition, analysis and visualization tools to meet the specific needs of the project. Another advantage is that we can easily integrate Matlab code into our program using Mathscript nodes. Matlab code can be written directly in the LabVIEW program or called as needed.

In order to simplify the maintenance of the code, we have used a modular approach in the software architecture. This will also allow future development of the code to be easy. The software is organized into control and functional modules. This division of functions allows the software developers with limited knowledge of the communication protocol between the hardware and software to be able to modify and adjust the software component. Also any changes made to one module will not affect the overall system and can be seamlessly integrated into the project. There a number of design patterns in LabVIEW used to develop maintainable and readable code. While various patterns are used in the design of our program, the main structure is based on a state machine architecture. At any instant, the system is in a particular state and based on user input switches to another. When the program initially starts, it waits for a valid connection with the prototype. Once this has been satisfied, it will continuously monitor user input and function accordingly. By using this pattern, the user can perform an operation (change states) at any time and there is no particular sequence of steps that must be followed; all functions are running in parallel and the program will respond immediately to any input.

The control functions primarily responsible for sending and receiving data and commands to and from the hardware component have been written in C++, using the set of open source Boost libraries. The collection of these C++ functions is compiled into a DLL which is called

within LabVIEW. Since LabVIEW only supports functions in C, the C++ code is first converted to C and then used in our program. This approach allows for the developers to focus on one aspect of the system, without needing to worry about the interactions with the hardware system. The general architecture of the program can be divided into 4 layers, as shown in figure 3.3:

1. Presentation layer (mainly the GUI)
2. Workflow manager (to control the state of the program)
3. Data processing
4. Control module to communicate with the hardware

Access to each of the four layers is not necessarily in order. The GUI will frequently access the data as they become available to display on the interface. A layered approach makes it easier to deal with the growing complexity of the system.

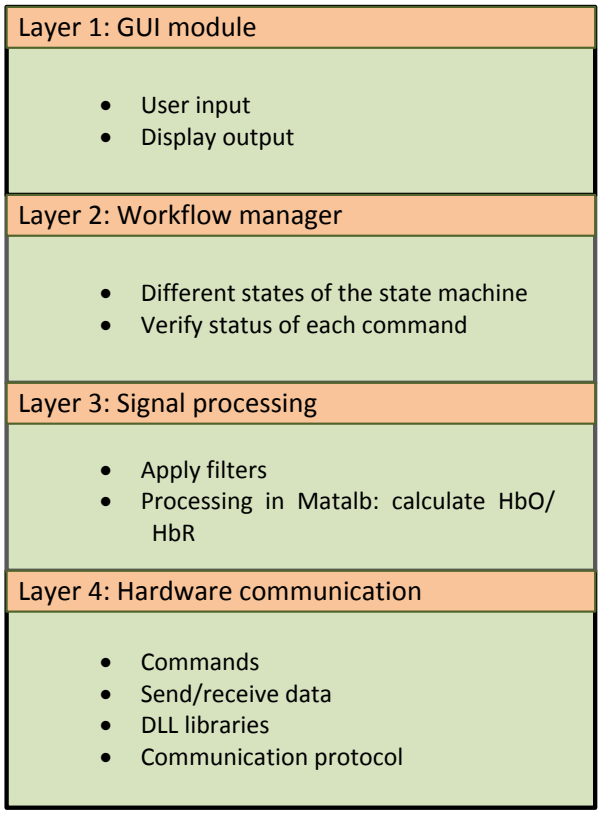


Figure 3.3: Layered architecture of the IMAGINC software

3.4.1 LabVIEW program

Each independent operation that can be potentially reused later in the program is compiled as an individual VI in LabVIEW. This makes it simple to provide an elegant representation of the complex code in the form of various blocks, each with a series of clear input and output connections while speeding up the processing. The general structure of each VI can be easily modified or customized if required. This also ensures consistency throughout the program. There are various design patterns used in our system. A few are discussed below.

Event loop:

The event loop performs an action in response to a user interaction. This is much better than polling since it does not block other processes and there are no lost events. As soon as there is a change in value, the program will respond immediately. The timeout duration is set according to the particular event. If no action occurs within the allocated time frame, a message is displayed.

In some cases, where we need to monitor for an external event, a polling loop may be the only solution. If this is the case, then a waiting time should be assigned. This will force the loop to sleep for some time (depending on the sleep value, usually a few milliseconds) and greatly reduces the CPU utilization. This approach will allocate some CPU time to the other threads while the loop is executing. In other cases, we may be waiting for an internal event to occur within the program. Polling loops should be avoided and the use of occurrences is suggested. The code following the occurrence will not be executed and no CPU time is allocated, until the occurrence condition has been met. In our program, the data acquisition will not start until the correct configuration file has been sent to the prototype. This is implemented as an occurrence and prevents the start of data acquisition without correct configuration settings and features.

State machine:

The state machine implements any complex decision making algorithm in the form of defined states. It performs a specific action for each state, with clear conditions for transition to the next one.

Master/Slave:

This pattern is used when parallel processes need to run continuously at different rates. This will reduce the need to use local/global variables to pass information between the two processes and prevents race conditions. The master loop writes to the shared data, and the slave reads the data.

Multithreading:

One of the advantages of using LabVIEW is the implicit parallelism and threading. The program automatically divides the application into multiple execution threads which can be run independently and in parallel. This may potentially result in a faster execution speed and faster display of information, depending on the complexity of the tasks. In our program, this concept can be applied to the data acquisition loops. NIRS, EEG, trigger channels, accelerometers and analog channels can all run independently, in multiple loops, with different paths of execution while performing their task, regardless of the rest of the program. If the internal operations are time-consuming, we may require an additional thread for supporting the display and GUI update. More threads are not necessarily an advantage, since each thread will be scheduled and requires CPU time. The less number of threads, the more execution time is available per thread. The best solution is to modularly group the code into subVIs, and then apply multithreading. This will reduce the thread count as well as avoids assigning a single thread to basic/simple operations that can be performed together. The order of execution may sometimes be forced, such as in sequential tasks, which limits the multithreading feature. In conclusion, to take advantage of the threading engine, we must limit forced order of execution and leave as many execution paths as possible. However there is some level of order required, and this is up to the developer to determine which operations require can be potentially performed in parallel.

In addition to the discussed methods, data types can also impact the application performance, particularly when large amounts of data are involved. We will briefly discuss the most common types used in our program.

Integers and boolean are processed relatively fast. The execution speed of floating point numbers depends on the precision format and the platform being used, as well as the type of operation. In general floating point operations is slower than integers. String processing is slower, since LabVIEW uses a specific format to store them (4 bytes for string length, followed by the

character data). Longer strings also take longer to process and copy internally (for each operation, a string duplication is performed).

Arrays are widely used structures, and should be used with caution in performance-critical processes. Good practice involves array preallocation and then inserting the values. In our program, most of the data manipulation is performed on arrays. However most are relatively small in size and the values are replaced (rather than inserting into a new array or array copying). In general, the copying and memory allocations need to be considered when operating on arrays. Clusters are another data type that can be relatively slow to process, depending on its contents. They can contain different data types, grouped together for easier variable passing. We have used this structure in the configuration tab setting, to group the emitter/detector coupling, light intensities and bias voltage together.

In addition to the data type, the scope of the variable (global vs local) is also important. Global variables that are altered by more than one VI are hard to manage and troubleshoot by causing race conditions. They are a common source of error and should be used only if required. Local variables can be used for passing information between 2 or more parallel loops without forcing a dependency on the execution order.

It is clear that in addition to the data management of each component, we need to consider its impact on the program as a whole, without affecting the speed and performance. In order to control the flow of execution between various states of the program, and to ensure that the program is in one state at any time we have used a workflow manager block. This module is organized in the form of a state machine. The transition from one state to another is only allowed if all the conditions are met. The default state of the system upon start is a waiting state, called "init" state, where it monitors the user interface for activity. For each command, it verifies the validity of the command, and switches to the appropriate state. For example, the user cannot start an acquisition without defining the emitter/detector coupling first. This approach also helps prevent user errors to some degree. The various states of the state machine are presented in figure 3.4. A detailed explanation of each of the states is explained below.

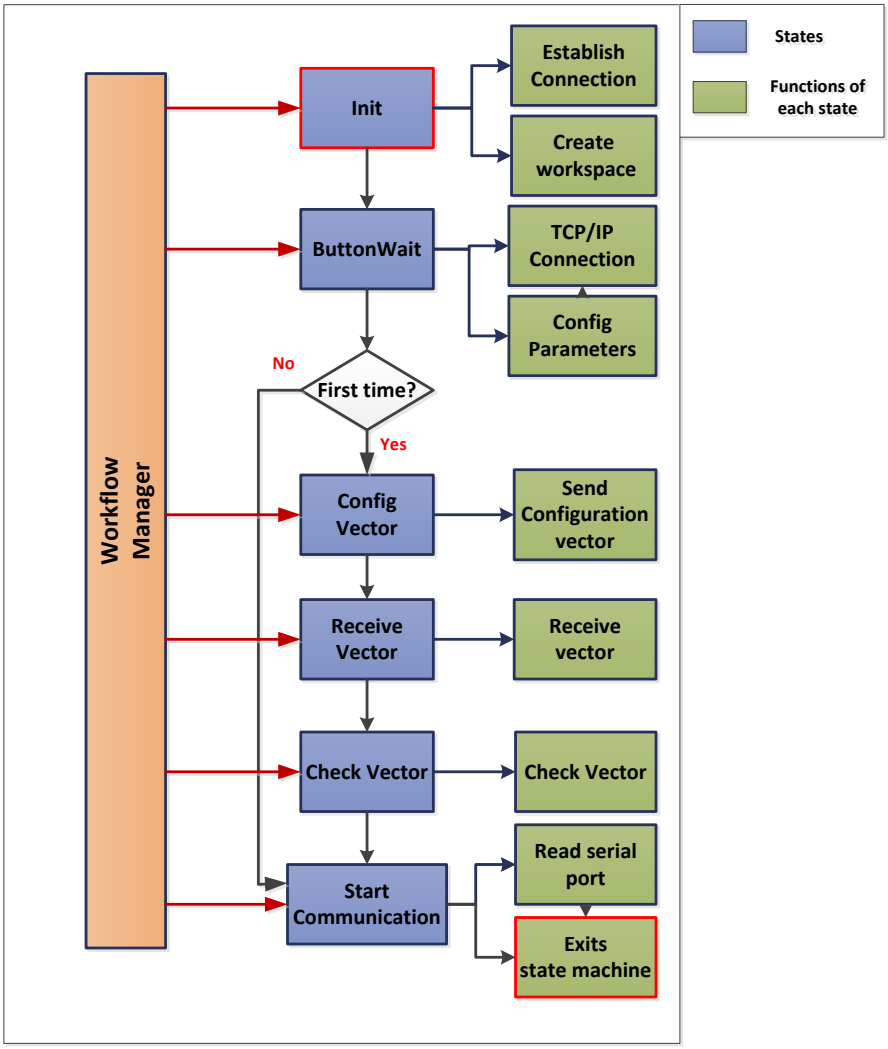


Figure 3.4: Different states of the workflow manager

1. Init

This is the default idle state of the state machine. It consists of a while loop that continuously monitors for a serial or Bluetooth connection. Once it has detected a connection, the serial port is opened and the InitConnection VI is called to start the communication. This VI sends the serial communication settings, such as the port number and baud rate to the Serial Port Write VI of the LabVIEW library. In case of a connection error, it notifies the user. If the connection is successful, the second window (output location window) will be displayed. The folder for saving the data files will be created on the user desktop. The transition code is the "ButtonWait" state.

2. Button Wait

This state consists of a series of synchronized loop structures. Each one listens for user event for each of the buttons. The user can change any of the parameters before choosing to send the data to the prototype. Using an event structure the program will be notified if the user has clicked any buttons.

The TCP/IP connection button can be selected in this state (the LabVIEW program will be the server, and the Matlab UI will be the corresponding client side running simultaneously). More details are discussed in section 3.5. The configuration file settings, which include the emitter/detector coupling, the illumination intensity for each emitter and the photodiode bias voltage, will be saved in a file. If the user chooses to upload a previously saved configuration file, this is also possible and the values on the table will be updated accordingly. When the user clicks the "send" button, the state diagram will switch to the "configVector" state. However if the user has previously sent a configuration setting (e.g. the previous run of the program), they may click the "start" button. In this case the next sate will be "startCommunication". The purpose of this dual mode is that the user may wish to perform several recordings with the same configuration settings. In this case, there is no need to resend the configuration file every time. This allows to quickly perform various recording without needing to re-establish the serial connection and repeating the previous steps.

3. Config Vector

This state reads the bytes available using the SerialPortRead VI. It creates the vector containing the defined parameter settings to send to the prototype. The 32 emitters and corresponding detector pairs are organized in groups of 8. It will send all the couplings, along with the emitter illumination and photodiode bias voltage. For each emitter, it sends the corresponding information in the following order:

Ex Dx_1 Dx_2 Dx_3 Dx_4 Wl_1
Ex Dx_1 Dx_2 Dx_3 Dx_4 Wl_2

where Ex is the emitter, Dx_1 – Dx_4 are the corresponding detectors. Wl_1 is the first wavelength, at 735 nm and Wl_2 is at 850 nm.

Once the user clicks the “send” button, a progress bar will display the status of the data transfer. Once the transfer is complete, the “send” button will be enabled. The next transition state is “receiveVector”.

4. Receive Vector

This state will read the available bytes at the serial port and pass it to the next state, which is the “checkVector”.

5. Check Vector

This state will make sure that the correct number of bytes and commands has been sent to the prototype based on the type of acquisition and number of channels.

6. Start Communication

The display window will switch to the NIRS tab to display the NIRS curves. The program will exit the while loop and proceed with retrieving the data and displaying them. The next step involves creating the NIRS, EEG, trigger, accelerometer and analog channel files. LabVIEW uses the TDMS (TDM Streaming) file format. TDMS is a LabVIEW specific file format which stores data in a three level hierarchy of objects.

The top level contains the name of the file, in the format of “AcqName_DDMMYY/TIME”. The second level is the source of the recording, NIRS or EEG. The third level consists of the channels, 256 channels for NIRS, 32 channels for EEG. The remaining three parameters (digital trigger, accelerometer and analog channel) are all stored in one file as three subsections. The configuration file is saved as ProjetName00x.cfg (x increases with each new acquisition in the same folder). The NIRS data is saved as ProjectNameNIRS 00x.tdms which contains the 256 NIRS channels and the EEG data is saved as ProjectNameEEG 00x.tdms which contains the 32 EEG curves. The TDMS format is also convertible to a Matlab structure, which can be used in HomER (NIRS analysis toolbox) as well. We have included a feature which quickly converts .tdms to .nirs and .mat files.

Certain functions are performed using shared libraries written in C. The DLL is called from LabVIEW using the Call Library function. The rest of the program is organized in a sequence structure, which executes each block in order. Using LabVIEW libraries, the appropriate TDMS files for each data type is created and opened. Data will be written and saved

to the file continuously. The next frame of the sequence structure is made of a series of synchronous while loops that run in parallel. The stopping condition for all is when the user clicks the stop button and the recording is stopped. Each while loop performs one of the following measurements:

1. EEG
2. NIRS
3. ADC/ACC/Trigger
4. Auto calibration
5. Modify photodiode voltage
6. Modify LED intensities

The tasks performed in each loop are explained.

1. EEG: The appropriate DLL file reads the values for each channel and displays it on the graph on the user interface. The EEG graphs are organized in groups of 8. There is also a low pass filter that can be applied to the data, which can be changed as necessary. All data values are saved to the TDMS file. The user can also view the bipolar or referential montage. The term bipolar refers to the linkage of the adjacent electrodes along longitudinal or transverse lines. The bipolar montage consists of a series of channels, where each channel represents the voltage difference between two adjacent electrodes. In a referential montage, each channel is the difference between an electrode and a specified reference electrode (at location Cz). In our system there are three electrodes that can be freely placed at any location, 2 referential electrodes installed over the mastoids and 1 ground near the forehead. Figure 3.5 shows the standard 10-20 system electrode placement.

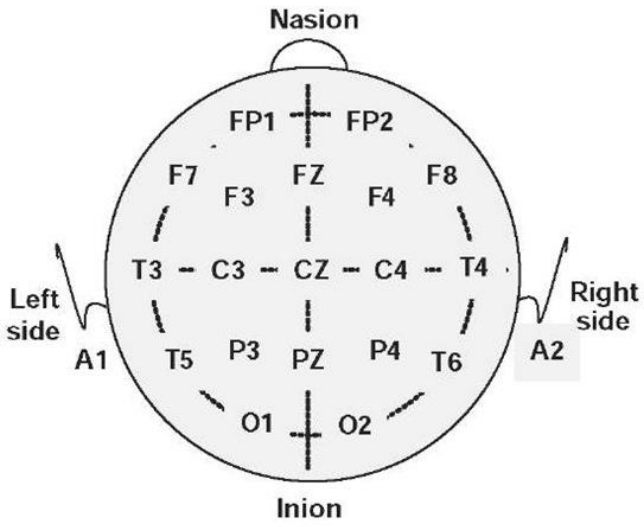


Figure 3.5: 10-20 standard EEG probe placement

For example, the channel Fz – Cz is the voltage difference between the Fz electrode and Cz electrode. The name of each electrode can also be set by the user to provide more meaningful results. The montage file is a text file consisting of EEG channel number and channel name pairs on each line, in the following format:

EEG channel number	Electrode name	EEG channel number	Electrode name

Referring to the mentioned example, the format would be "1 Fz 2 Cz" as the first line in the file. The user can create the montage file before the test, and simply import it to view the results. There is also an option of creating a montage file during a recording session. Each EEG graph can also be set to a visible/invisible state to simplify viewing and interpreting the results.

2. Modify photodiode voltage: At any time if the value of the photodiode voltage changes, the new value will be sent using the respective DLL.

3. Modify LED intensities: At any time if the values of the emitter illuminations are changed, the new values will be sent to the prototype through the DLL. The table in the Configuration window will also be updated to reflect these changes. There is also error checking to make sure the selected value is in the appropriate range.

4. ADC/ACC/Trigger: Using a case structure, reads the values from the ADC, accelerometer and trigger channels and writes them to the TDMS file. The graphs are also displayed. The values are from the accelerometer along 3 axes at 50 Hz. This parameter will be used to distinguish motion artifacts from hemodynamic variations

5. Auto calibration: Although the illumination intensity of the emitters are set to default values, in order to achieve the best results we need to make sure we have the maximum illumination without any saturation. An automated process will choose the best value for each emitter by adjusting the intensity. The intensities are increased while checking to make sure they are still in range. This is done continuously for all the channels and sent using the DLL to the prototype until the desired value is reached. This operation is performed in only a few seconds.

6. NIRS: Using the appropriate DLL, the data that is read from the prototype and displayed on the NIRS graph. The data is also enqueued using the queue structure in LabVIEW for calculating the oxy and deoxy-hemoglobin values. This calculation is done in a separate while loop explained later. Similar to the EEG curves, the NIRS curves are organized in groups of 8. The emitter and detector number as well as the wavelength is displayed beside each graph. The photodiode voltage and emitter illumination can also be changed from this window.

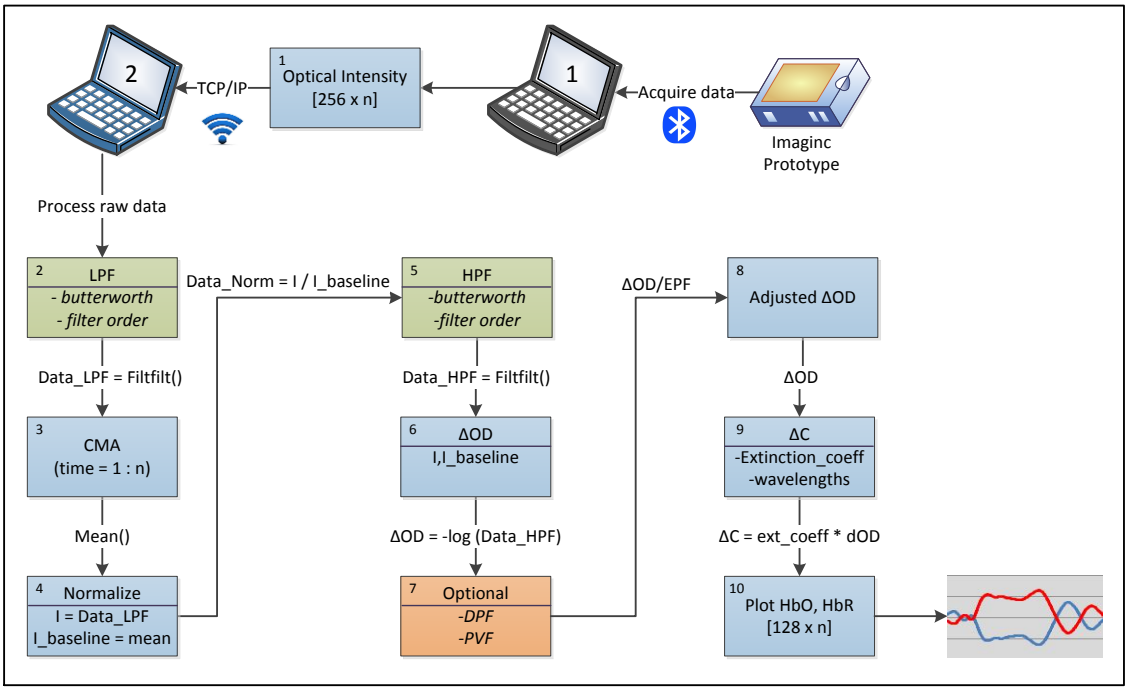
Another tool for optode installation verification is the “Analyze” button. This will show the status of each emitter, and if there is any ambient light pollution or cross-talk between channels. The status can be checked at any time during a recording. For each data point, it will determine if the value is at the maximum threshold or below a certain range. Based on the value, it will mark the signals from a particular channel as “saturated” or “non-detecting”. This will allow for constant verification during a recording especially during movements. Throughout the test it is useful to be able to mark an occurrence or event during an acquisition for later analysis. The software uses a trigger graph to mark the time of an event.

If the user wishes to view the topographic reconstruction of the HbO and HbR concentrations, the data will be sent over a TCP/IP connection to a second computer for processing in Matlab. We use LabVIEW’s built-in libraries to send the channel and data bytes to the Matlab program. A topographic image reconstruction will provide a visual indication of the active regions of the brain. The IP address of the computer running the LabVIEW program is

required in the Matlab code to establish a connection between the two. The Matlab client program needs to be running simultaneously to allow proper transfer of data.

3.4.2 Matlab program

The oxygen calculation processing is done in Matlab on a second computer. This allows for remote monitoring of the process through the network. The acquired NIRS data is sent to the second computer via TCP/IP protocol, where it is then processed and displayed as it is received. Since all channels do not necessarily have the exact same number of data points, the received data values are first placed in a queue structure. Once all channels have an equal amount of data points they are dequeued and processed. A separate function receives all the incoming data and sorts them into 2 matrices, one containing the channel numbers and the other a 256xn matrix containing the data (where n is the number of points read for each channel). Figure 3.6 shows the processing steps involved.



LPF = Low pass filter
FiltFilt = zero-phase digital filtering function
CMA = Cumulative moving average
HPF = High pass filter
ΔOD = Changes in optical density
EPF = Effective pathlength factor
DPF = Differential pathlength factor
PVF = Partial volume factor
ΔC = Changes in concentration

Figure 3.6: Processing algorithm of raw data performed in Matlab GUI

As seen in the figure, the data is first transmitted via Bluetooth to the first laptop, which then transmits it the second computer. The first step is applying a low-pass filter. The user can specify the filter type (Butterworth, Chebyshev, and Elliptic), order and cut-off frequency. The default value is a second-order low pass Butterworth filter, with a default cut-off frequency at 0.1 Hz. This is implemented using the Matlab *filtfilt* function. The next step is taking the mean of all the current data points. We have used a cumulative moving average method (CMA), as shown in equation 3.1. The process is recursive and efficient, and the mean value is updated as new data becomes available.

$$CMA_{n+1} = CMA_n + \frac{x_{n+1} - CMA_n}{n+1} \tag{3.1}$$

where x_n is the data available at time n and $CMA_0 = 1$.

From the MBLL discussed in section 1.1 we know that we need both the intensity and a baseline value to calculate the changes in optical density. We use the mean data as the baseline value, as shown in block 4 in figure 3.6. In the next step a high pass filter is applied. Similar to the low pass filter, we can adjust the parameters. The default value is a third order Butterworth filter, with a cut off frequency of 0.005 Hz. The change in optical density value is calculated by taking the negative logarithm of the high pass filtered values, as indicated in block 6 of figure 3.6. The delta optical density values, which is referred to as "dOD" contains the values for each emitter at the two wavelengths. Based on the order of recording from our prototype, we know that the values for each channel at wavelength 1 and 2 are separated by an index of 4. This is shown in table 3.1.

Table 3.1: Format of emitter and detector coupling at two wavelengths

Curve number	Emitter	Detector	Wavelength
0	Ex_1	Dx_1	1
1	Ex_1	Dx_2	1
2	Ex_1	Dx_3	1
3	Ex_1	Dx_4	1
4	Ex_1	Dx_1	2
5	Ex_1	Dx_2	2
6	Ex_1	Dx_3	2
7	Ex_1	Dx_4	2

Thus we couple the appropriate data and create an output array for each emitter at its two wavelengths.

Another optional parameter is setting the differential path length factor (DPF) and partial volume factor (PVF). The user can select the extinction coefficients to be used in the calculations by selecting the values from one of the references (detailed list of references is explained in section 3.5.2). The DPF and PVF values will be calculated accordingly and the dOD value will be updated. The default values are [50 50] for PVF and [6.11 5.87] for the DPF values. In the final step the changes in concentration for oxy and deoxy-hemoglobin is calculated by multiplying the extinction coefficients by the dOD. This is shown in block 9 in figure 3.6. The final output is [128 x n] array for each of HbO and HbR. The oxy and deoxy-hemoglobin graphs for each of the channels are displayed on the HbO and HbR windows on the user interface. They are organized in groups of 4, shown as an emitter-detector pair in the form of *Ex-Dx.* To provide a better feedback of brain activity, a topographic image reconstruction of the hemoglobin concentration changes is also displayed in a separate figure. This requires uploading a template file of the emitter-detector positions and selecting the desired brain view from the interface. The concentration variations are recorded as a movie file which can be viewed later. All the data is saved as .mat files, which can be easily be viewed in other programs.

3.4.3 Hardware communication Layer

We will highlight the important details of the hardware design and parameters that is relative to the software design. To facilitate the maintenance of the software, we have separated the hardware system configuration settings from the rest of the software.

The prototype consists of 32 near infrared light sources, 32 light detectors and 32 EEG electrodes (sampling rate of 320 Hz). Each light source can be coupled with 4 detectors at 2 wavelengths, providing a total of 256 NIRS channels (with sampling rate of 20 Hz). The EEG probes are inserted through the cap and placed on the scalp using conductive gel. In addition to these, there are also two auxiliary analog channels, one digital trigger channels and two accelerometer channels. The accelerometers will provide data on motion artifacts and can help in removing the noise from the signal. One can be placed on the helmet, and the other on a desired location depending on the test protocol.

The signals acquired from all channels are prepared as data packets by the hardware and transmitted to the software. The control module packages the data in the form of 8 EEG and 4 NIRS samples at a time, and sends them to the computer through the USB or Bluetooth connection. Each received packet is checked to make sure it contains all the required information using an error detection code. In the case of data loss or error, an error code will be sent to the software and the received data will be interpolated to compensate for the missed points. The information is sent as packets, where each packet is 32 bytes. The structure of the packet is organized as:

- 2 Bytes for the header
    - Contains information on the wavelength, the detector number, whether there is auxiliary data or not, and the packet number.
- 2 Bytes for control
    - Type of data channels (NIRS or EEG)
- 28 Bytes for data
    - 2 Bytes for digital trigger channel
    - 2 Bytes for digital auxiliary data channel
    - 24 Bytes remaining for NIRS (2 Bytes) and EEG (2 Bytes) data
        - There are 256 NIRS channels, at 20 Hz vs. the 32 EEG channels, at 320 Hz. Thus we need twice the number of EEG data compared to the NIRS. Based on the available number of Bytes (which is 24), we can have 4 NIRS (8 Bytes) and 8 EEG (16 Bytes) data in each packet

In order to detect information loss, all packets are numbered (from 0-63) and sent. This allows for temporary packet loss for duration of 3.2 seconds, which is the result of the number of packet numbers sent at the sampling rate of 20 Hz (64 / 20 Hz). If this is not the case, an error detecting code is used to interpolate the data. The control module and the computer are linked using a USB cable emulated by a UART RS-232 serial port using an FTDI chip on the device, or through a Bluetooth serial port module (OBS241).

The accelerometers measure the acceleration changes along the three axes at 50 Hz. This feature is used in later version of the system to remove motion artifact and improve the SNR. The

analog data channels are sampled at a frequency of 340 Hz each at 12-bit resolution using an ADC.

Figure 3.7 shows the components of the entire system. It consists of a control module that acquires the data, which is connected to the illuminator and detector optodes via cables. There are also EEG ports that attach directly to the electrodes through the sockets on the helmet. The control module and the computer are linked over a Bluetooth connection. This wireless connection allows for subjects undergoing tests to have some degree of movement within an approximate range of 6 – 10 meters in a typical recording setting.

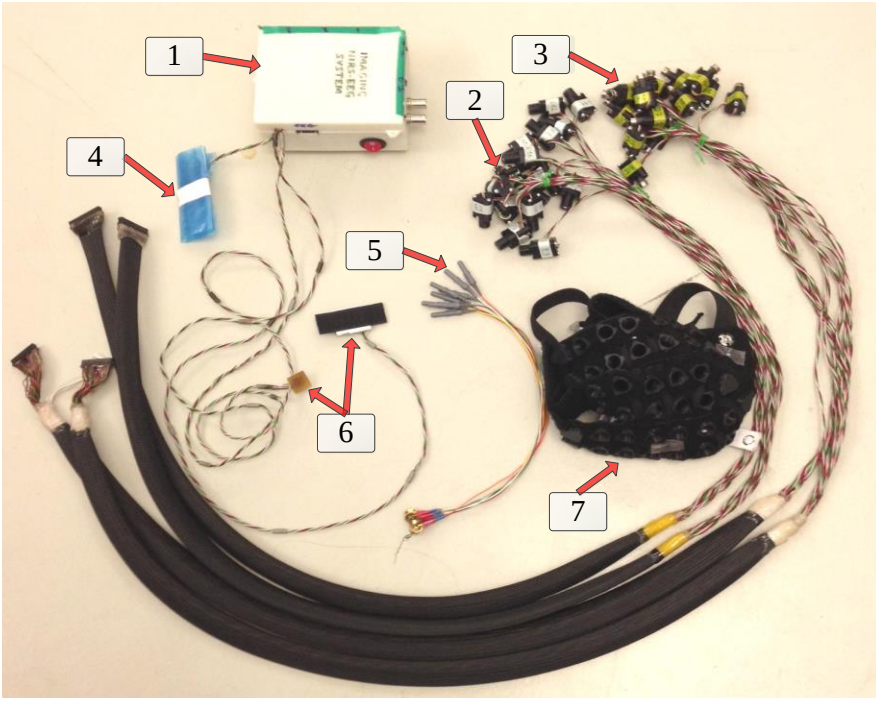


Figure 3.7: Various components of the Imaginc system. 1: Control Module, 2: Illuminators, 3: Detectors, 4: Bluetooth dongle, 5: EEG ports and electrodes, 6: Accelerometers, 7: Helmet

3.5 User interface

The user interface is an important element of the overall software design. In our project the end user of the system will be clinicians and technicians, thus it is important to develop an easy and intuitive model. Some of the improvements of the current version of the GUI compared to the original one is that has a user-friendly interface, various parameter selection options and an

overall better performance. Figure 3.8 shows the interaction diagram of the GUI with the other parts of the system.

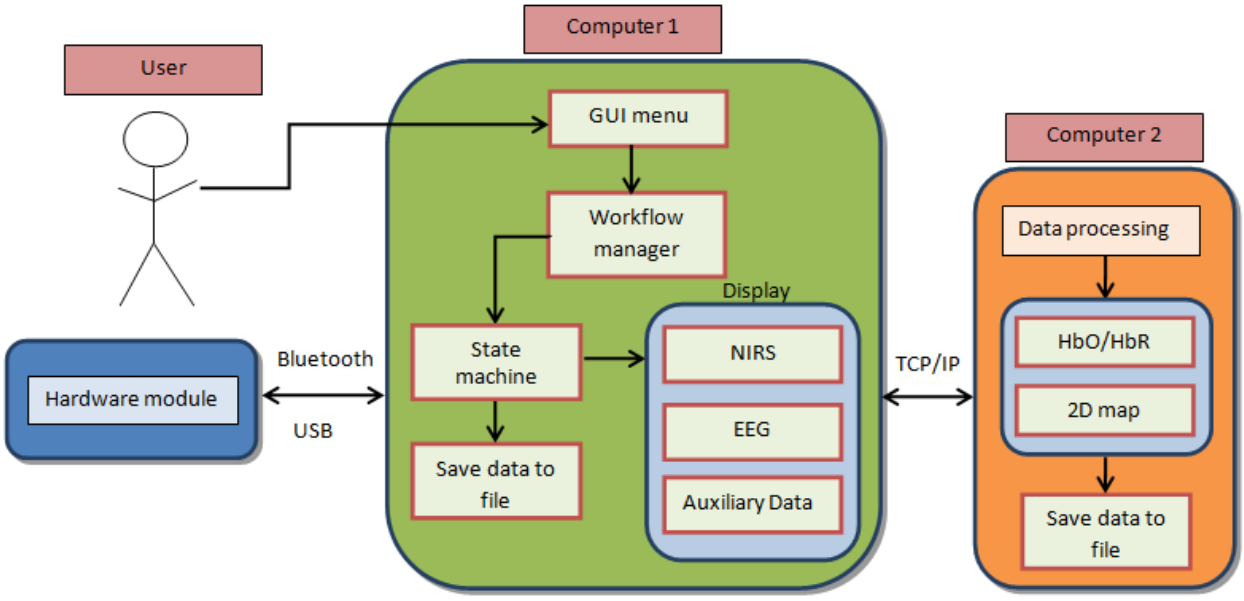


Figure 3.8: Interactions of the GUI with other components

In the next sections we will present the different windows of the user interface and the user options available from the GUI menus. The first section covers the LabVIEW interface and section 3.5.2 shows the Matlab user interface.

3.5.1 LabVIEW GUI

The entire GUI Is organized as a multi-structure window. Figure 3.9 shows the initial start-up screen that appears when the program is launched. There is an automatic recognition of the connectivity mode (USB or Bluetooth). In the case of an error, a message will be displayed and the system will not proceed to any further steps.

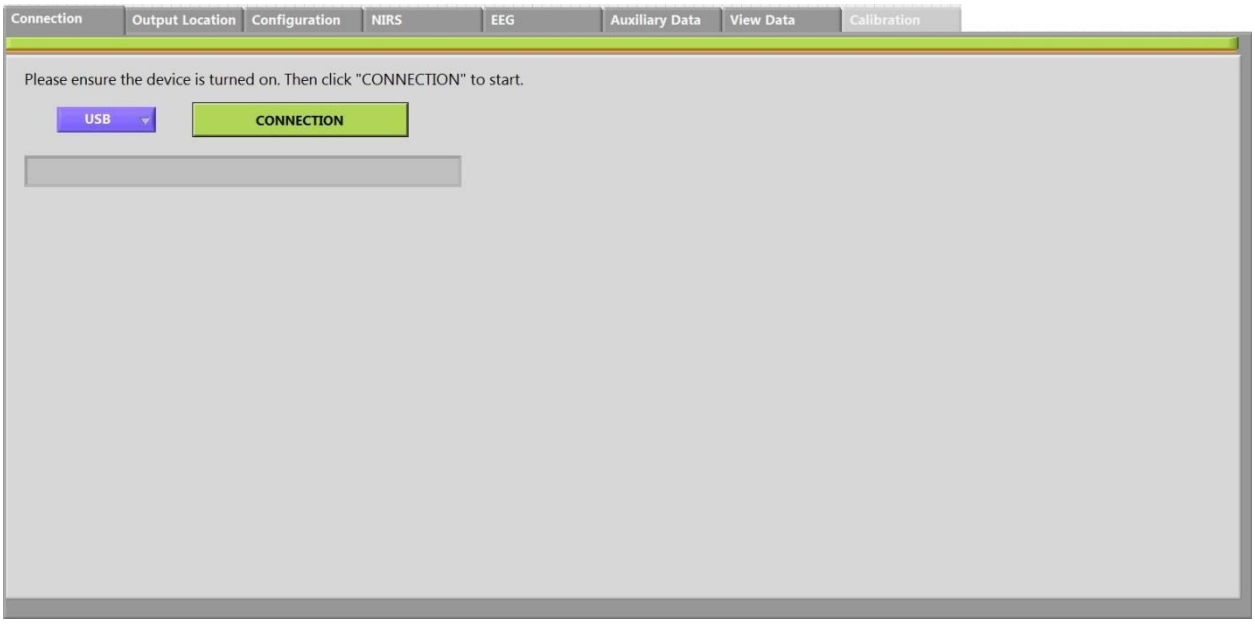


Figure 3.9: Connection window

When a successful communication is established with the hardware, the program will switch to the Output Location tab, and create a project folder on the user desktop. The next screen is the Configuration window. As shown in figure 3.10, box #1 is where the user can select the emitter/detector coupling. They are displayed in groups of 8, and they can be selected from the drop down menu on top. There is also an option to select a previous configuration file, which can be selected using the file browser shown in box #2. The user can also select the type of the acquisition (only EEG, only NIRS or both) from box #3. The 4 buttons on the upper right corner select the number of accelerometers to include. Once the settings have been made, the TCP/IP connection must be established with the client computer. This is done using the TCP/IP button highlighted in box #4. Using the orange “send” button the configuration settings are sent to the hardware and the green “start” button will start the acquisition. The program will switch to the next tab, which is the NIRS display.

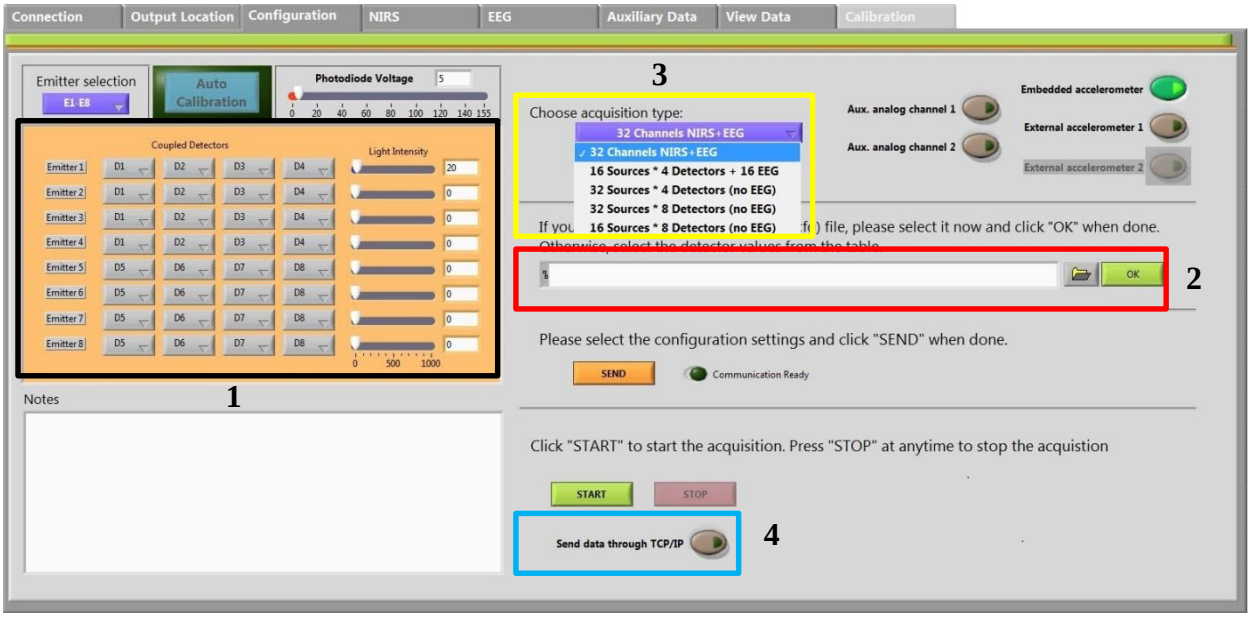


Figure 3.10: Configuration window

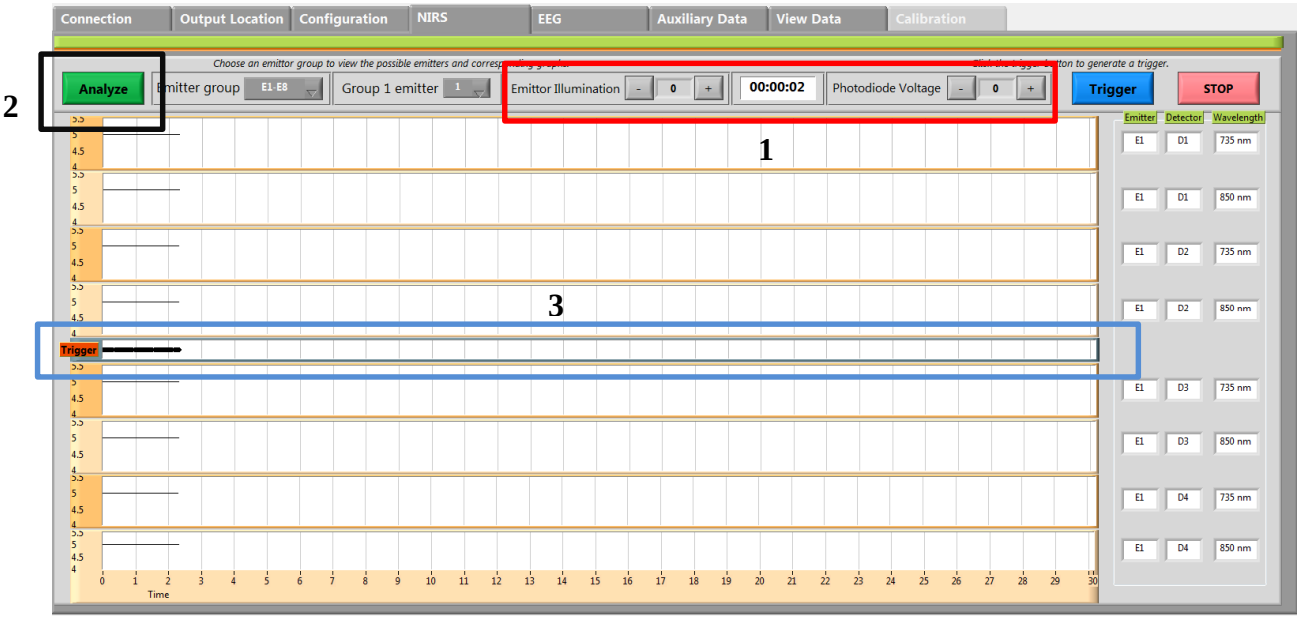


Figure 3.11: NIRS window

Figure 3.11 highlights the features of the NIRS window. The graphs are displayed in groups of 8, and the labels on the right side of each one show the emitter/detector pair for each graph. The emitter illumination and photodiode voltage can be adjusted at any time during an acquisition. These buttons are on the top, shown in box #1 of figure 3.11. The green "Analyze" button, on the left side provides the saturation information for each channel, as explained in section 3.4.1(box

#2). The trigger button will record the time of a specific event, as indicated by the user. The corresponding graph is shown in box #3.

The next tab is the EEG display. Similar to the NIRS window, the graphs are displayed in groups of 8 for each of the 32 EEG channels as shown in figure 3.12. The type of filter can be selected from the menu shown in box #1, and the results will be updated immediately. The user can switch back to the NIRS tab at any time. The scales of the graphs are also adjustable (box #2). For example selecting an x-scale of 10 will show the data 10 points at a time, while the y-scale will zoom in/out on the data. The user can also choose to show/hide individual graphs to only display the relevant ones on the screen. Each of the 32 EEG graphs can have custom labels (e.g. Fz, Cz) to provide more meaningful results when the various channels are being compared in the montage section (box #3). There is also an ON/OFF button beside each graph to change the visibility. The last two channels can be used to connect to an ECG (or any other) device and the scale can be changed from microvolts to millivolts for easier reading as shown in box #4.

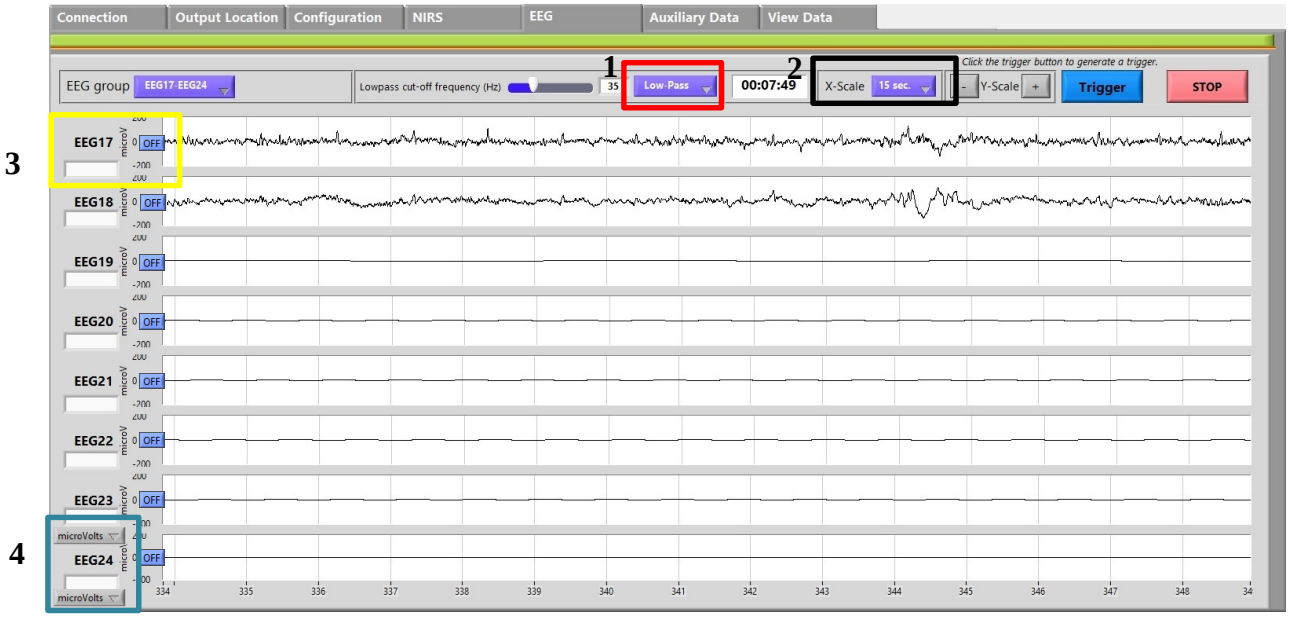


Figure 3.12: EEG window

Figure 3.13 shows the EEG montage tab. In the montage selection, there is an option of loading a previously configured EEG montage file (box #1), or saving the current configuration. The results will be displayed for each pair in the adjacent graph. Even if a graph is not visible the data will still be recorded. The rest of the features are similar to the ones in figure 3.12.

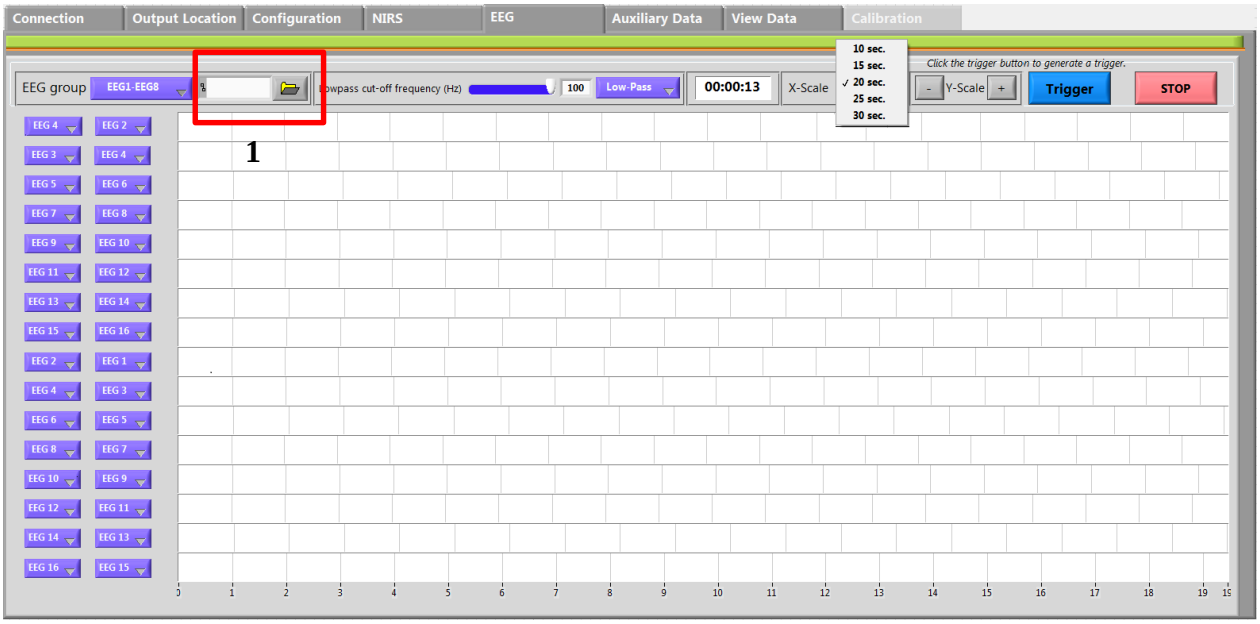


Figure 3.13: EEG montage window

The next tab is the auxiliary data window (figure 3.14), which displays the information from the accelerometer, analog and the digital trigger channels. The battery percentage remaining is also displayed (box #1).

Figure 3.15 shows the "View Data" window which is where all the recorded data files can be opened and viewed in LabVIEW. These files are also available after the recording in the project directory. There is also an option to automatically convert the data files (which are in tdms format) to a HomER-compatible format to view and analyze in other analysis software. There is a conversion button for each of the NIRS, EEG and accelerometer data files.

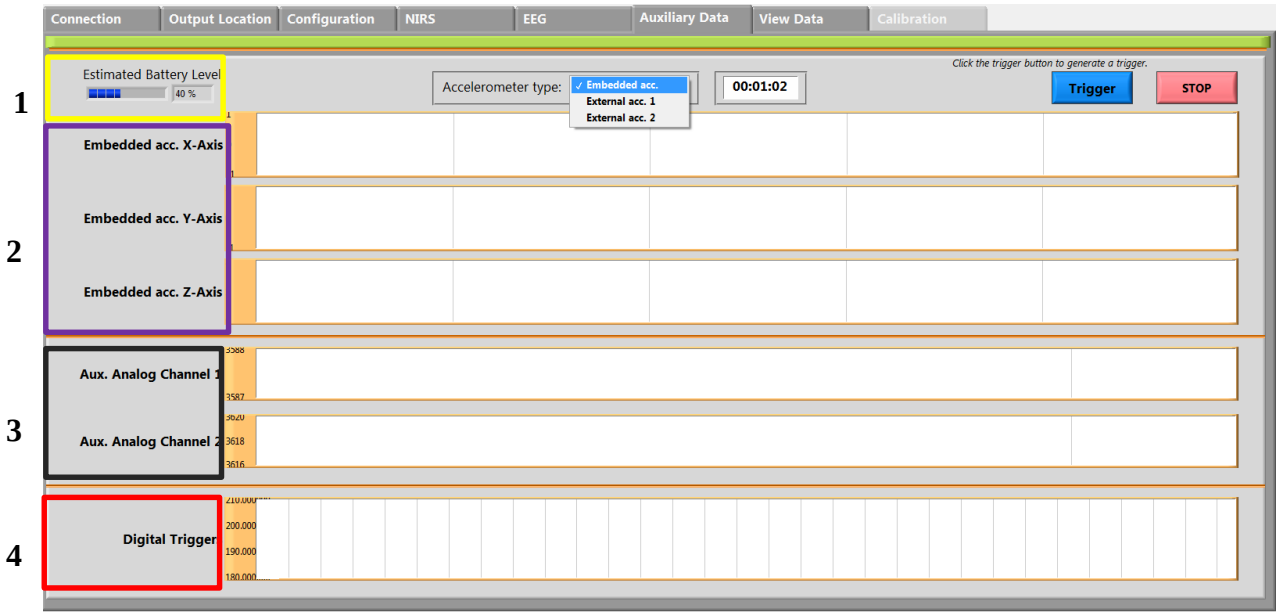


Figure 3.14: Auxiliary data window

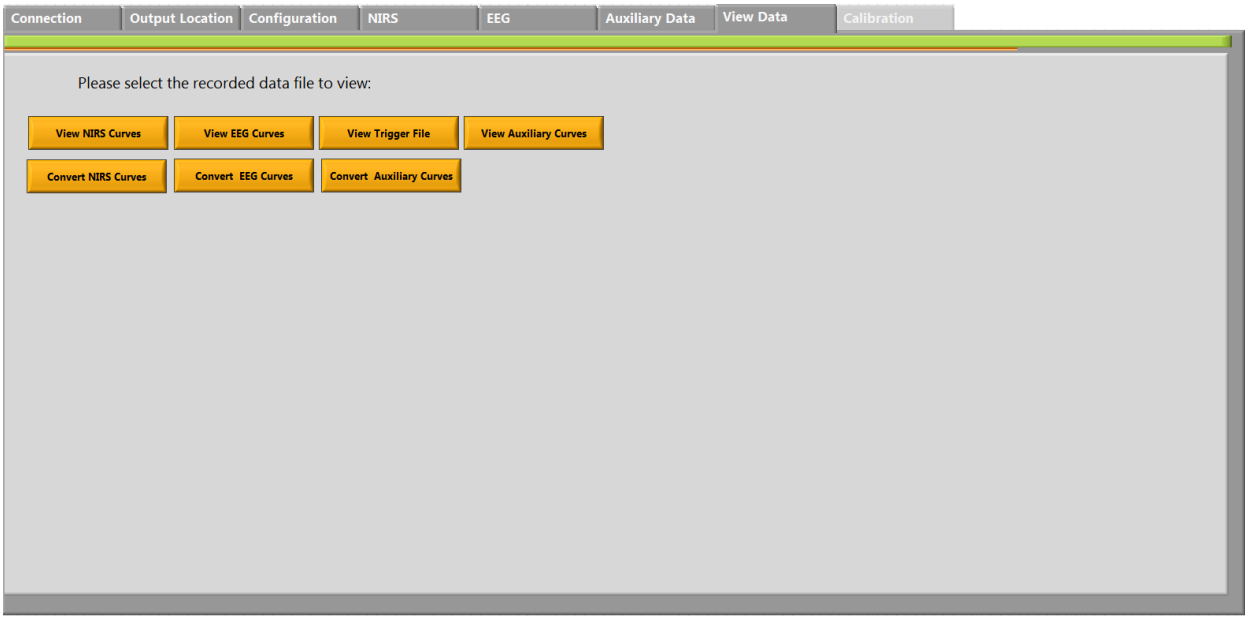


Figure 3.15: ViewData window

The acquisition can be stopped at any time by clicking the red “STOP” button available on all windows. The user will be brought back to the configuration screen, where they have the option of changing the parameters or using the previous ones (figure 3.10).

The presented GUI displays a total of 294 curves, organized as follows:

- 256 NIRS curves

- 32 emitters coupled with 4 detectors at 2 wavelengths (32 * 4 *2)

- 32 EEG curves
  - 32 EEG electrodes
- 3 accelerometer curves
  - Data acquired from an external accelerometer, and displays information along each of the x, y, and z axis.
- 2 auxiliary analog curves
  - Data from 2 analog channels
- 1 digital trigger curve
  - Records the time for event-related protocols

3.5.2 Matlab GUI

This Matlab GUI is created using the GUIDE tool. It allows quick and easy creation of user interface while being able to use Matlab's features and toolboxes. It will run in parallel with the main LabVIEW GUI, in order to visualize the oxy and deoxy-hemoglobin values. Figure 3.16 shows the start-up screen. The details of each of the sections are as follows:

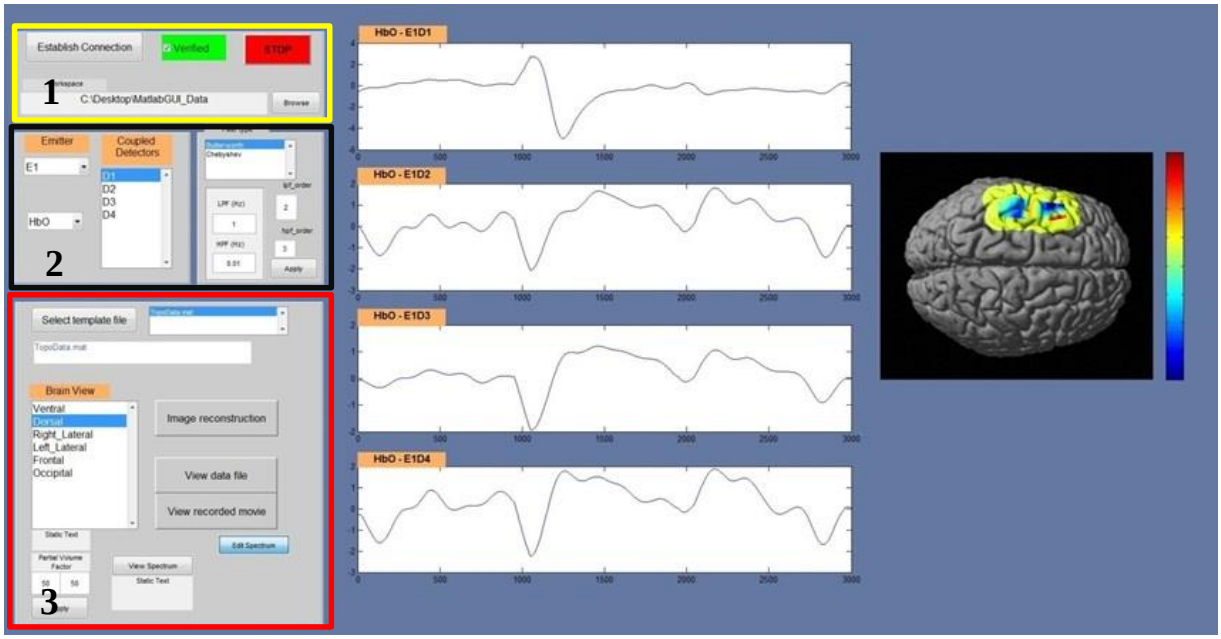


Figure 3.16: Matlab GUI

Box 1: The green “Verified” box will be checked if there has been a successful TCP/IP connection established with the host computer. Once this is complete, the other sections are enabled.

Box 2: The user can select any emitter (from E1 to E32) and the corresponding detectors (chosen in the beginning of the acquisition from the LabVIEW interface) will be displayed. There are three options to choose from: HbO, HbR and HbT. The values for each emitter-detector pair is shown in the four corresponding graphs in the center of the window. The type of filter can also be changed.

Box 3: The template file, which is a Matlab ‘.mat’ file based on the location of the emitter and detectors on the helmet can be selected from a file browser. This template has been previously created (using the Brainsight NIRS software system,) and the real coordinates of the optodes have been mapped onto the available brain template. The brain view selection allows the user to choose from one of the 6 brain views: ventral, dorsal, right lateral, left lateral, frontal and occipital. Once these parameters have been set, the “image reconstruction” button is selected and the image on the right displays a topographic image reconstruction of the chosen concentration parameter. The series of changes is used to create an animation, for each of the HbO, HbR and HbT parameters.

The parameters for the modified Beer-Lambert law can be adjusted. The default values for the partial volume factor (PVF) and the differential path length factor (DPF) are set to [50 50] and [6.11 5.87]. The absorption spectrum for the oxygenated and deoxygenated hemoglobin can be selected from one of the three references and the corresponding extinction coefficients will be used for the calculations:

1) “W. B. Gratzer, Med. Res. Council Labs, Holly Hill, London, N. Kollias, Wellman Laboratories, Harvard Medical School, Boston”
2) “J.M. Schmitt, "Optical Measurement of Blood Oxygenation by Implantable Telemetry," Technical Report G558-15, Stanford.”
3) “S. Takatani and M. D. Graham, "Theoretical analysis of diffuse reflectance from a two-layer tissue model," IEEE Trans. Biomed. Eng., BME-26, 656--664, (1987)”

Each selected reference will display the absorption spectrum on a separate pop up window. Figure 3.17 shows the spectrum for reference 1. Similar windows are displayed for the other references.

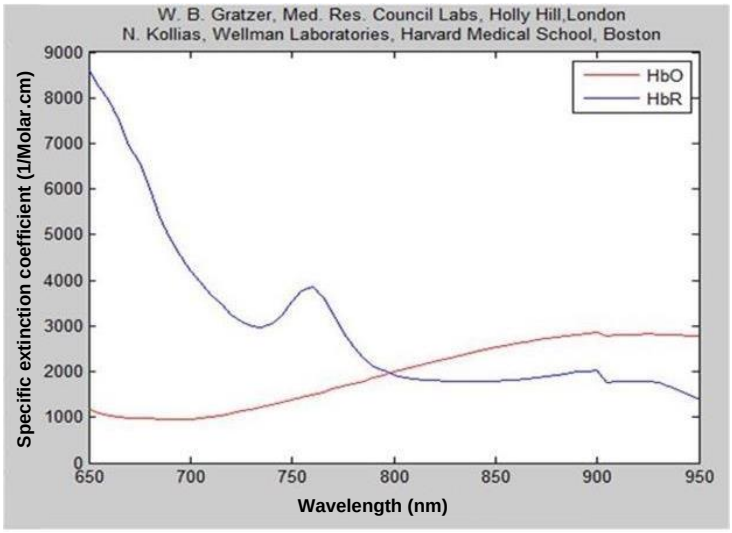


Figure 3.17: Absorption spectra of HbO and HbR based on values specified in reference 1

The Matlab GUI displays a total of 384 curves:

- 128 oxy-hemoglobin curves
    - 32 emitters coupled with 4 detectors at wavelength 1
- 128 deoxy-hemoglobin curves
    - 32 emitters coupled with 4 detectors at wavelength 2
- 128 total hemoglobin curves, which is the sum of HbO and HbR

In addition, it displays the topographic image reconstruction of the oxy and deoxy-hemoglobin values based on the positioning of the probes on the helmet.

CHAPTER 4 EXPERIMENTS AND VALIDATION

In this chapter we will present the processing algorithm results obtained using our acquisition system and their validation based on processing using the HomER toolbox in Matlab. In the second section we will present the software performance metrics, system complexity and UI metrics.

4.1 Processing algorithm and test protocol

The software system has been developed to retrieve data in real time, perform processing and display results in a continuous manner. In the first set of tests, we used recorded raw NIRS data files (acquired using our system) and simulated the real-time data transmission, meaning that the raw NIRS data was acquired at a rate of 20 Hz. The data was obtained from a previous study performed by (LeLan, 2013) using 18 subjects. The data was sent in small blocks of 20 points, which is the real size of data that the hardware module transmits.

The data acquired from the prototype is sent from the first computer (retrieved via Bluetooth) over a TCP/IP connection to the second computer for processing. The incoming data is stored in a queue structure, to make sure all channels have the same number of data points before processing. Since we know the test protocol used to acquire the raw data, we can perform an analysis of the HbO and HbR values calculated to verify the performance. The same calculations were also performed in HomER offline, using the same set of data with all data points. It is clear that as the number of data points used in the processing algorithm increases, the results become more accurate. However since it is a real-time display, we need to choose the minimum number of points that will yield valid results in a reasonable amount of time. A delay of 2 – 3 seconds is determined to be acceptable.

The test protocol was a finger tapping task using the right hand and the left motor cortex was imaged. The emitter/detector probes were placed on the desired location of interest. There were 4 emitters and 8 detectors used, providing a total of 14 channels. A detailed description of the optode placement is shown in figure 4.1.

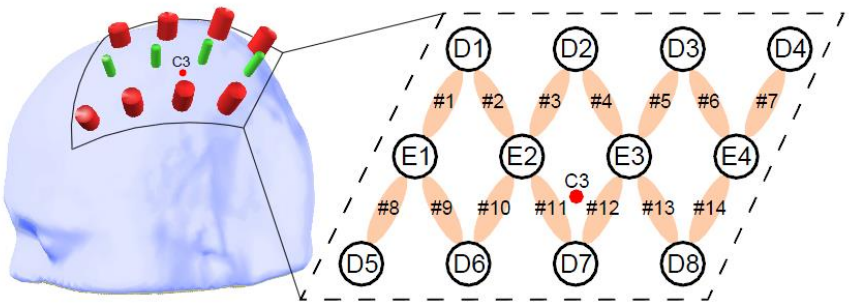


Figure 4.1: Emitter-detector placement in the finger tapping test. Image obtained from (LeLan, 2013)

The test paradigm is shown in figure 4.2. There is a 60 second baseline acquisition, followed by repeated blocks of rest and finger tapping of the right hand. The finger tapping period is 30 seconds followed by a rest period of 15 seconds.

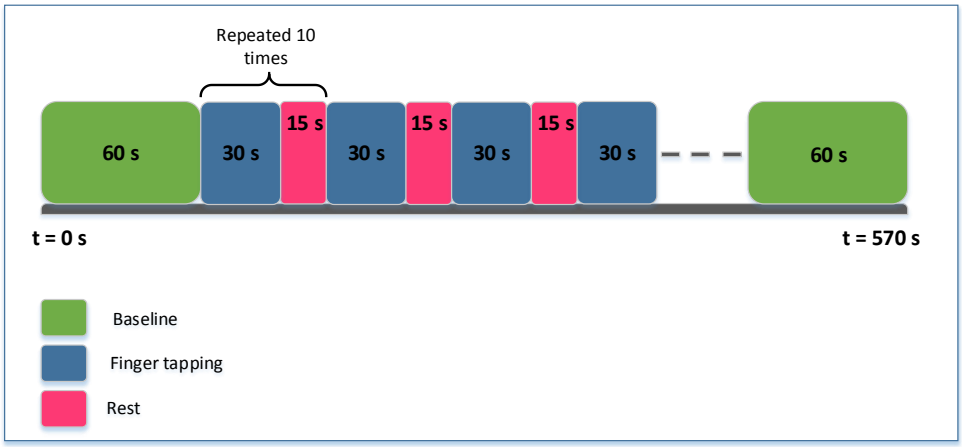


Figure 4.2: Finger tapping protocol details. Each test consists of 10 blocks of 30 seconds of finger tapping interleaved with 15 seconds of rest. Each test starts and ends with a 60 second resting period.

Figure 4.3 shows the calculated HbO values for all channels from one subject, processed using HomER and our program using all data points for the first 150 seconds. As seen in the figure, the two curves are very similar.

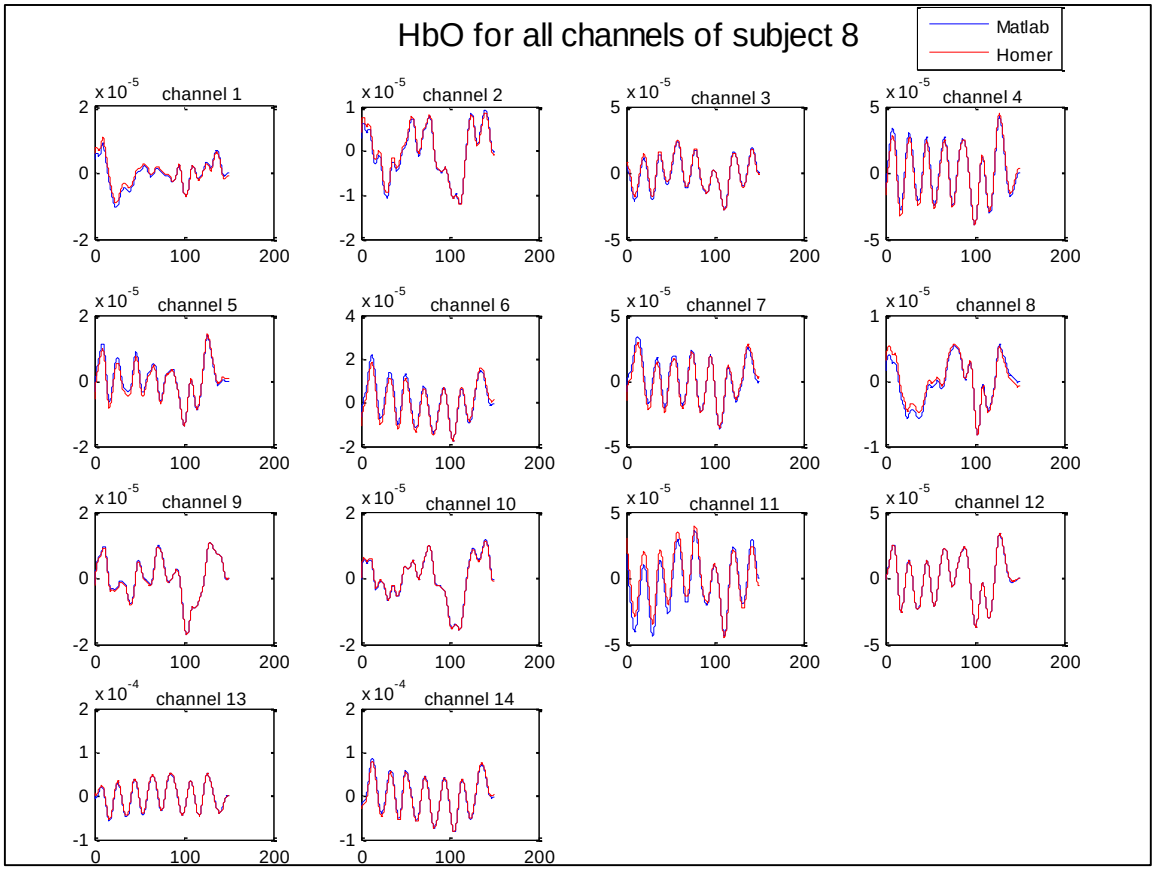


Figure 4.3: HbO values for all channels of subject 8. The results of HomER are shown in red and the results processed by our program are shown in blue. The x-scale has units of time(s) and the y-scale has units of dConcentration (Molar).

Figure 4.4 shows similar values for HbR. The correlation coefficient for each channel is calculated in table 4.1.

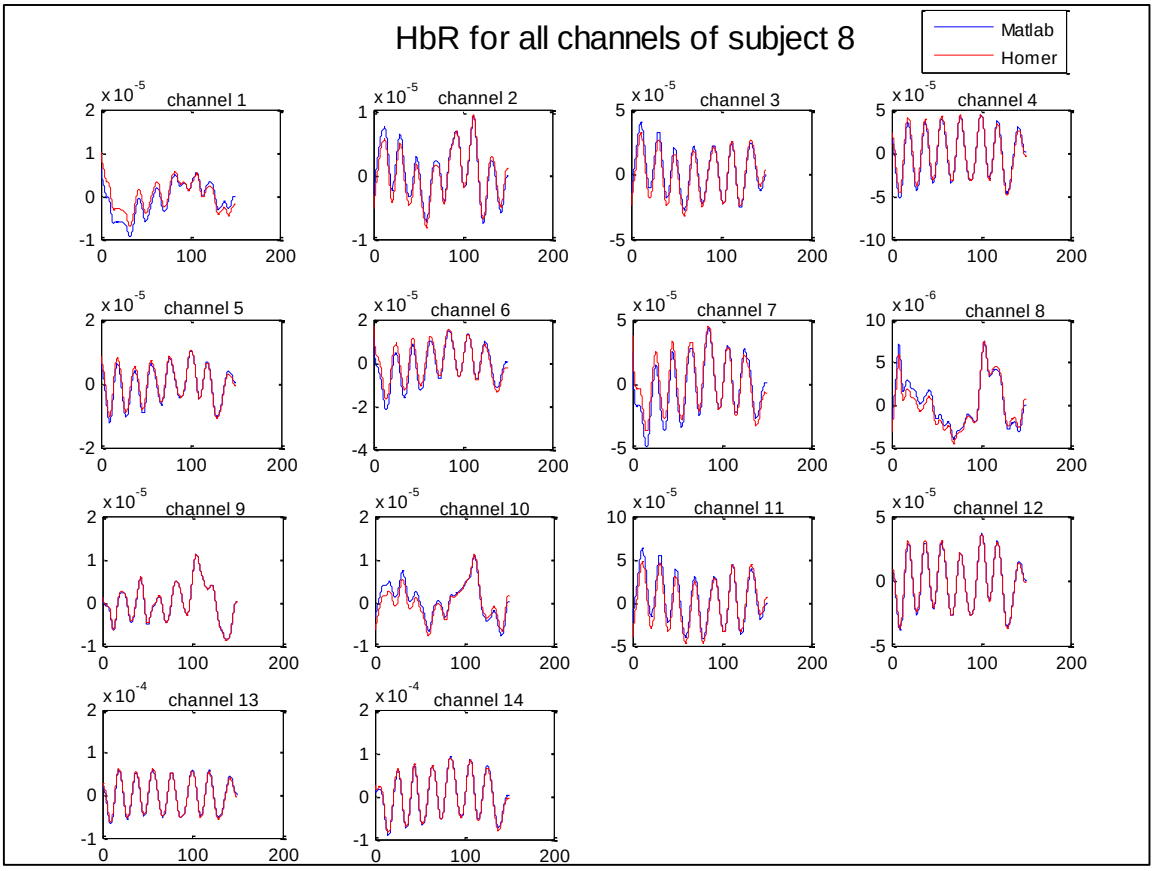


Figure 4.4: HbR values for all channels of subject 8. The results of HomER are shown in red and the results processed by our program are shown in blue. The x-scale has units of time(s) and the y-scale has units of dConcentration (Molar).

Table 4.1 shows the correlation coefficient for HbO and HbR for each of the channels displayed in figures 4.3 and 4.4. The Pearson correlation coefficient is determined using Matlab's *corr(X, Y)* function, where X and Y are the two time series. The correlation coefficient of two time series X and Y is determined using the following equation:

$$corr\ (X,Y) = \frac{cov(X,Y)}{\sigma_X \sigma_Y} \tag{4.1}$$

Where cov is the covariance and σ is the standard deviation.

Table 4.1: Pearson correlation coefficient for each channel of subject 8 processed using all data points.

Channel number	1	2	3	4	5	6	7	8	9	10	11	12	13	14
R^2 for HbO (%)	98.3	99.5	99.4	99.1	98.7	98.2	98.5	98.4	99.9	99.9	96.1	99.9	99.8	99.5
R^2 for HbR (%)	90.2	97.4	97.4	99.2	98.7	96.2	95.8	97.9	99.9	95.6	96.9	99.8	99.6	99.7

Table 4.2 shows the average R^2 values across all channels for each of the 10 subjects (using all data points in the processing.)

Table 4.2: Average correlation coefficient across all channels for each subject processed using all data points

Subject	Mean R^2 HbO	Mean R^2 HbR
S01	0.9613	0.9653
S02	0.9355	0.9270
S03	0.9937	0.977
S04	0.9370	0.966
S05	0.9511	0.905
S06	0.9534	0.888
S07	0.9540	0.9653
S08	0.9899	0.965
S09	0.9501	0.9536
S10	0.9827	0.9734
Average	0.9608	0.9485

With varying number of data points used in the processing, the exact signal quantification slightly varies, however the trend of the hemodynamic variations is correct. In our application, we are mostly interested in whether there is an activation present or not, and the location of the activated region, rather than the absolute values. Using this recursive approach, we are still able to compare the changes between different test paradigms and local mapping of activity based on the relative changes. Since the calculations are performed in real time, it requires processing

smaller blocks of data of 60 points, which based on the sampling frequency is 3 seconds. Figure 4.5 shows the processing of a single channel (channel 9) for one subject (subject 8) for HbO and HbR respectively. The correlation coefficient value for HbO and HbR is 0.8911 and 0.9540, indicating a strong correlation and the two curves follow a similar trend.

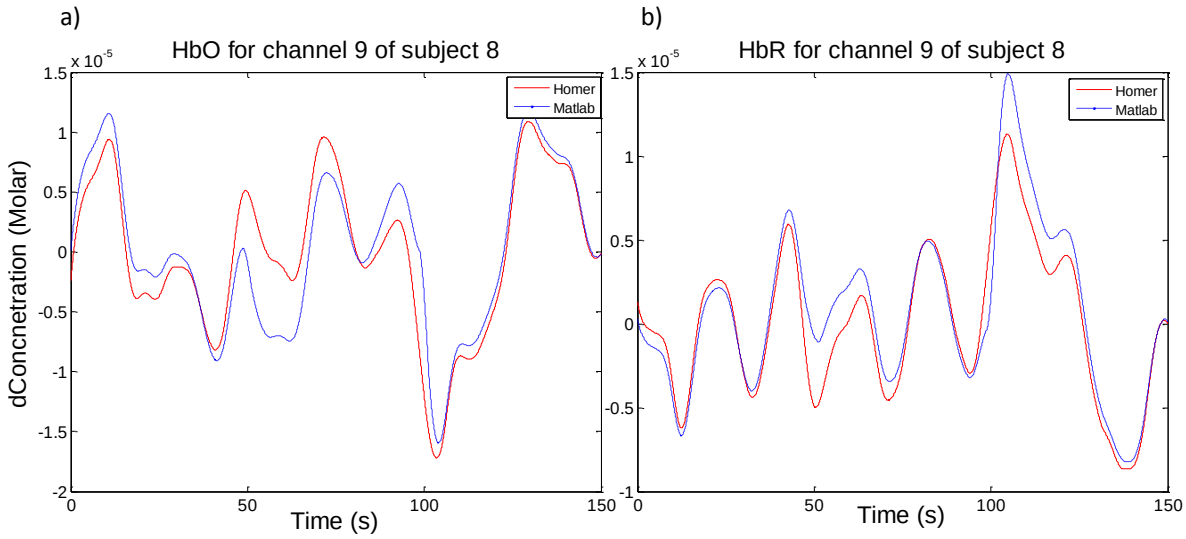


Figure 4.5: Block processing for one channel of subject 8. a) HbO curves for both HomER and our program (R^2 = 0.8911) b) HbR curves for both HomER and our program (R^2 = 0.9540)

Figure 4.6 shows the HbO and HbR values on the same plot. As expected, an increase in HbO is accompanied by a decrease in HbR during the activation period. The shaded areas indicate the rest periods. The top figure shows the entire test period, while the bottom one displays the results for the first 150 seconds. There is an initial 60 second baseline period. From time 60 – 90 seconds, the value of HbO increases as a result of the finger tapping task, while HbR decreases. HbO concentration decreases again during the rest period from time 90 – 105 seconds, while HbR increases. This change is repeated during each of the activation/rest blocks which confirm expected hemodynamic variations during tasks. The final baseline period is from time 510 – 570 seconds.

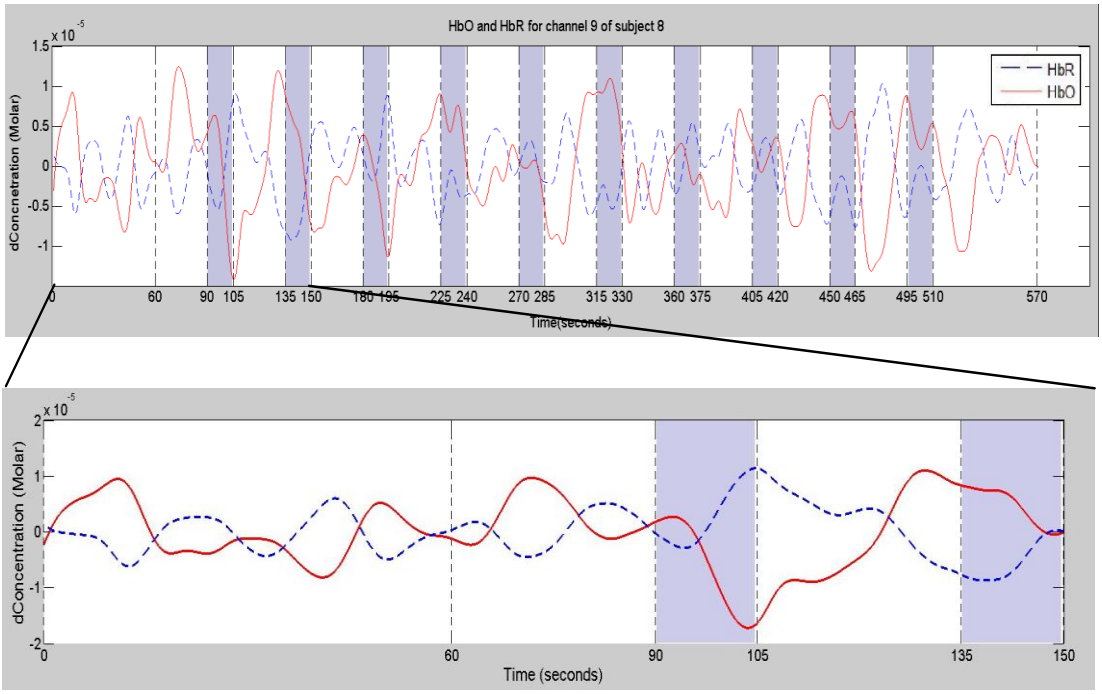


Figure 4.6: HbO and HbR for one channel of subject 8 with rest/activation blocks. Top figure shows the entire test session (570 seconds). Bottom figure shows the results from time 0 – 150 seconds.

To provide an overall comparison among subjects, 10 subjects were chosen, and the correlation coefficient, RMSE (root mean square error) and MAE (mean absolute error) of each subject (across all channels) was calculated, as shown in table 4.3. The RMSE and MAE are both measured in the same units as the data. The MAE is relatively similar in magnitude but slightly smaller than the RMSE. This is due to the fact the MAE is less sensitive to occasional large errors since the errors are not squared, as with RMSE.

Table 4.3: Comparison of results obtained for HbO and HbR for calculations of blocks of data. All reported mean values (R^2, RMSE and MAE) are across all channels for each subject.

Subject	HbO			HbR		
	Mean R^2	Mean RMSE	Mean MAE	Mean R^2	Mean RMSE	Mean MAE
S01	0.8897	5.70E-06	2.89E-07	0.8531	5.22E-06	1.38E-08
S02	0.8360	6.10E-06	6.52E-07	0.7794	5.45E-06	6.11E-07
S03	0.8592	5.16E-06	1.03E-06	0.8699	4.64E-06	2.85E-07
S04	0.9013	8.41E-06	2.09E-06	0.9105	5.80E-06	6.03E-07
S05	0.7650	3.89E-06	2.06E-07	0.8283	1.38E-06	1.38E-07
S06	0.8568	2.55E-06	4.72E-07	0.8046	2.72E-06	1.84E-07
S07	0.7813	4.65E-06	7.53E-07	0.8660	2.36E-06	2.77E-09
S08	0.9205	5.39E-06	3.32E-07	0.9045	7.24E-06	1.86E-07
S09	0.8598	4.80E-06	4.15E-07	0.8817	3.05E-06	3.14E-07
S10	0.8809	6.17E-06	2.66E-06	0.8495	2.53E-06	9.66E-08
Overall average of all subjects	0.855	5.28E-06	8.91E-07	0.854	4.04E-06	2.42E-07

The overall average correlation coefficient for HbO was determined to be around 85.5%, with an average RMSE of around 5.3e–06. Similarly for HbR, the average correlation coefficient was around 85.5% with an average RMSE of around 4e–06. While the correlation coefficients for both parameters are similar, HbR values seem to follow the HomER calculations much more closely, resulting in smaller RMSE and MAE values. In most NIRS studies, the results are analyzed based on the region of interest during the test. Therefore usually the channels with a higher SNR are considered in the analysis. In our case we have considered all channels, even the ones which our not necessarily relevant. Our goal was to compare the overall quality of results obtained using the two methods.

4.2 Software metrics

The software system has been designed to perform continuous monitoring without any data loss or interruptions, to ensure valid results. The data also needs to be displayed to the user in a real-time manner. To quantify these parameters, we have measured the average response time for each of the major time-consuming operations of the system. Assuming there are no hardware malfunctions, the device can operate for up to 24 hours without any interruption. This has been verified using a NIRS phantom and monitoring the signal quality acquired through the software over the whole period (LeLan, 2013). The NIRS signal quality was evaluated based on the changes in standard deviation value of the signals. This value was calculated every 60 seconds and was a constant value of 1.2 mV (the ADC input range was 5 V) (LeLan, 2013).The software functionality was also tested separately by running the software connected to the hardware (without any optodes or phantom) for a period of 24 hours. The device was operated in Bluetooth mode and all data files were saved. The acquired values remained constant throughout the acquisition.

To measure the response time of each operation, we used the "Profile performance and memory" tool in LabVIEW. The acquisition procedure was performed 10 times for a session of around 20 seconds. Each measured function is listed below and the results are displayed in table 4.4.

- T1: Read NIRS data
- T2: Read ADC data
- T3: Read EEG data
- T4: Perform auto calibration: This time is variable, depending on the individual subject and test setting. The default illumination (initial value) is set to 500, and this value is then increased/decreased accordingly.
- T6: Increase/decrease emitter illumination manually
- T5: Increase/decrease photodiode bias voltage
- T7: Change the low-pass cut off frequency of the EEG
- T8: Time to display the status of all NIRS channels using the "Analyze" button

Since the data is transmitted to another computer for hemodynamic variation calculations, we have also measured the following:

- T9: Time from when user clicks the start button on the first computer to the time the HbO/HbR concentrations are displayed on the second computer. Depending on the required input size for processing, it is usually 3 – 4 seconds

- T10: Time from when user clicks the start button on the first computer to the time the topographic image reconstruction of the hemoglobin concentration values is displayed on the second computer.

Table 4.4: Average response time of system to different operations.

Task	Average time (ms)
T1	280.8
T2	31.2
T3	124.8
T4	4.5e04 – 6e04
T5	47.5
T6	46.2
T7	50.1
T8	40.2
T9	3e03 – 3.5e03
T10	3e03 – 4e03

4.2.1 System testing

Testing has been performed in two stages: unit testing and system testing. Unit testing focuses on testing individual system components. This step is done from the beginning of the development cycle so it easier to identify and correct faults from the early stages. It also reduces the complexity of the system testing which is done during the final stages and once the whole system has been developed. In our system, each individual function (that was potentially going to be reused) was organized as a VI. This allowed us to test each component independent from the others.

System testing is defined as testing the integrated solution with all the components to ensure it satisfies all the functional and non-functional requirements. As each major feature was integrated into our program, system testing was performed to ensure compatibility and

correctness. A final system test included testing the software and hardware module connected to optodes for a period of 24 hours to ensure functionality.

4.2.2 System complexity

A common tool for measuring system complexity in text-based programming languages is source lines of code (SLOC). However in LabVIEW, due to the graphical programming approach used, we must analyze the system using other metrics. The LabVIEW "VI metrics window" is a tool that provides an overview of the complexity of the specific VI, in terms of number of nodes, various structures and function calls. It is important to note that both metrics only provide an estimate of the measurement of complexity. In order to provide practical and meaningful results, we have chosen the important parameters and results in order to provide a measure of our system complexity. The number of nodes in a VI is a rough metric comparable to the standard SLOC approach. The number of nodes is the sum of functions, sub VIs, structures, front panel object terminals, constants, global and local variables and property nodes. In our system, the total number of nodes is 5513, with an average of 125.30 based on the total number of independent Vis used.

The system complexity measures the degree to which components are related. There are two terms that define the complexity: cohesion and coupling. Cohesion refers to the extent that the methods and properties within a component are specific to that particular module, and serve to accomplish an individual purpose. A system with high cohesion is considered desirable, indicating that the various modules are maintainable. Another term related to system complexity is coupling. It is defined as the degree to which different components in a system are dependent on each other for execution. While there is some sort of information sharing between modules, the amount of data transferred should be limited. Thus an optimal system is one with low coupling, often referred to as loose coupling. This provides an advantage of having independent components that can be easily reused and modified without affecting the entire system.

Our system uses several user-defined and general-purpose LabVIEW libraries. The number of specific user-defined libraries is 44, and the number of LabVIEW libraries used is 86 for a total of 130 VIs. Thus nearly one third of the libraries used are custom to our specific application, emphasizing the fact that we have used independent components that are loosely coupled, yet each one performs a specific task creating high cohesion among the various

modules. The number of shared library calls (calls to DLLs) is 17, indicating that we have been able to reuse previously developed (although some of these DLLs have been modified) code in our software system. Another measure of the complexity of individual components is the fan-in and fan-out (Ian, 2011). Fan-in for each component is the number of components that call this particular one. Similarly, fan-out is defined as the number of components that are called by the particular component. In order to have modular and reusable components, the number of fan-in should be maximized, while the fan-out should be minimized to less than 7 (Lee, 2013). In our LabVIEW user interface, the average number of fan-in for a component is 7.02, while the fan-out is 2.05. Thus we can conclude that our system design is still manageable and modular, despite the complexity. Another test parameter calculated using LabVIEW tools is the modularity index. It is defined as:

$$\text{Modularity index} = \left(\frac{\text{number of user Vis}}{\text{total number of nodes}}\right) * 100 \quad (4.1)$$

A higher modularity index indicates the use of individual sub VIs rather than performing all operations in the main VI. A low modularity index (approaching 0) indicates that all code is one large diagram and there has not been many VIs used, which can make it hard to manage. The measured modularity index for our system is 3.2. It is recommended in (Blume, 2007) that a modularity index of greater than 3 is an acceptable value. While this is a relative parameter and there is no standard value, but for most programs that have a medium complexity this value is determined to be between 2 and 4.

Another factor that affects the overall performance is the number of structures in the program. This includes for loops, while loops, case structures, event structures and flat sequence structures. The average number of the structures (based on the number of VIs) is 7.3.

4.2.3 General UI metrics

In terms of overall system quality, based on the requirements defined in section 3.2, the program needs to provide a user friendly interface that is easy to use, robust and reliable. There are a number of terms that need to be defined:

- Complexity: Each task should be easy to accomplish using system indications and guidelines.

- Learnability: is defined as the ease with which a new user could learn and use the system.
- Satisfaction: is the willingness of the user to use our system.

In order to specify these metrics, we have considered the following.

1. Easy to use
    - This is determined based on the user survey and training time required.
2. Robustness
    - This is determined based on the probability of data corruption (files are not saved properly) in case the program stops and time to restart after program failure. It is also the acceptability of effects when the product is given invalid input.
3. Reliability
    - This is based on the rate of failure, in other words how often the product fails and how bad the effects are.

User satisfaction survey

In this section, we will examine the software system in terms of user interactions and overall user experience. Since the software program will be used by clinicians and hospital staff, it is important to provide a clear and usable interface that is easy to learn and troubleshoot, and user preferences and opinions are important. Common methods for evaluating the user interface include observing the user during a live interaction with the system and gathering information through questionnaires on the user's experience (Ian, 2011). The first step was performed by attending various recording sessions and taking note of any difficulties/problems that occurred. We have also used the feedback from 4 users, who were familiar with the concepts of NIRS/EEG data recording, and have used the IMAGINC system a number of times to acquire real time data. They also have experience using other similar commercial software. They each filled out a survey on their overall user experience. The results should be analyzed taking into consideration that the sample size is small, however at this point there are a limited number of users. A better survey should ideally include more participants to obtain a better feedback. The survey questions are listed below. Each answer was rated from 1 to 5, with 5 being completely satisfied, and 1 being completely dissatisfied.

1. Overall appearance of the software and attractiveness (colors, font size,…): 100% of participants were completely satisfied.
2. Ease of use in terms of navigability and simplicity (from the first step, which is turning on the prototype and starting the program): 75% of participants thought it was very easy (5/5) while 25% rated it as (4/5).
3. Performance of our system compared to similar commercial software you have used: 50% rated it as 4/5 while 50% rated it as 5/5.
4. Level of training required for a new user: 100% of participants thought it required minimum training.
5. Overall user satisfaction: 75% of participants were completely satisfied (5/5) while 25% were somewhat satisfied (3/5).

Easy to use:

Overall, our system was found to be easy-to-learn, due to the separate windows dedicated to each task. The selection menus were clear and self-explanatory. The ultimate goal is that with minimum training, a user could successfully operate and control the device (learnability). All the tasks were identified to be easy to perform (complexity), through the display of required user actions on the screen, or automatic switching to the appropriate panel. Overall user satisfaction was reported at 75%.This can be explained by the fact that our system is still a prototype and while it performs required functions similar to a commercial software, it has been designed and tested on a smaller scale. Thus it is acceptable that the software system has room for many more improvements in future versions.

Robustness:

One problem that was observed was the user speed in menu/button selection. The system was reported to freeze often, which required restarting the prototype and software program. However this was more due to the user quickly increasing/decreasing values or clicking buttons before one operation had fully completed. Although the system responds quickly to user input, it requires a small delay between various operations to function properly. For example when the system is sending the configuration settings to the prototype or performing auto calibration, no other activity should be performed. To prevent unexpected errors, there is a progress bar

displayed to the user, indicating there is an operation in progress. There is also error messages displayed when an invalid operation is performed to notify the user. For example when the Bluetooth connection is not successful, the program does not crash, it simply displays a message to retry.

Reliability:

The program is considered to be reliable, meaning that the rate of failure is low. In case of a system failure (after the acquisition has started) all the data up to that point is saved. If the acquisition has not been started but the configuration file has been sent, the configuration file will also be saved. However the program needs to be restarted to start a new LabVIEW session. This only takes 3 – 4 seconds. The system has also been tested while running for a period of up to 24 hours, with the prototype connected. All the data was successfully recorded, and the behavior of the application remained the same. This was determined by the recorded file size and consistent number of data points for each channel. We have also performed a test to run the program several times in series. This was performed 15 times and in each session the data was sent to the prototype and the acquisition lasted for around 5 minutes. All the data files were examined to ensure correctness. Only one out of fifteen sessions (6.67%) was unsuccessful and the program did not operate properly due to an internal LabVIEW error.

CHAPTER 5 GENERAL DISCUSSION AND FUTURE WORK

In this section we will discuss the overall system, as well as its current limitations and areas for future improvements. This includes short and long-term goals in terms of hardware and software development.

We have been able to successfully retrieve and process the EEG and NIRS data, as well as displaying the hemodynamic variations. EEG electrodes are secured using a conductive gel, and the connection remains relatively stable throughout the test. However connection quality is more problematic with the NIRS channels. As with all NIRS acquisition systems there is still the concern of systemic physiological interference that contaminates the signals. Emitted and detected light both have to penetrate through several layers of tissue before it reaches the brain, adding superficial noise to the acquired data both upon entry and exit. This can impact the accuracy of the hemodynamic variations. The source of this physiological noise is usually from spontaneous hemodynamic oscillations in different frequency bands. The cardiac signal (~ 1 Hz), respiratory signals (~ 0.3 Hz), Mayer waves (~ 0.1 Hz) and very low frequency oscillations (VLF) which are less than 0.1 Hz, are all considered a contributory factor (Diamond et al., 2009; Obrig et al., 2000; Payne et al., 2009; Toronov et al., 2000). It has been shown that the amplitude of the Mayer waves and VLF further depends on the activity being performed and age (Peng et al., 2008). Thus one method of improving the SNR and obtaining better results would be to include these parameters in the signal processing chain. Our present hardware system includes two pairs of short distance emitter-detector sockets that are separated by a distance of 1 cm. This measured light intensity is mainly the activity along the superficial tissue and it contains the same systemic interference present in the other channels. We can use this short channel measurement to build repressors to eliminate this undesirable component and improve the overall measurements in the longer channels, as demonstrated in the work done by (Gagnon et al., 2012). It has also been shown that using multiple short separation channels (both at the source and detector) rather than just one, is more feasible and can provide more accurate results (Gagnon et al., 2014). This can also be considered in the future design to increase the number of the short distance channels available. Physical space and managing the layout along with all the other optodes and EEG electrodes must also be considered. In our current design, the optodes and electrodes cannot be placed on the exact same location, due to the separate designs. This can lead to a difference in the exact imaging location of combined NIRS-EEG systems. Many solutions

have been proposed that integrates the optodes and electrodes into one sensor to overcome this limitation by maximizing the number of possible sensors placed in a given area (Cooper et al., 2009). This will ensure optimal co-registration of the two data sources. Gel-free probes (dry electrodes) for EEG detection is also feasible since it would eliminate the need to apply and remove gel at the scalp (Lopez-Gordo et al., 2014), resulting in a shorter setup time and less discomfort to the patient. This has also been implemented in commercials solutions such as the one developed by NIRSport, which has a dedicated circuit at each EEG probe that verifies the impedance.

Another method for removing motion artifacts is using the data acquired through the external accelerometers. This is especially useful in protocols that involve patient movement and exercise routines performed in outdoor environments. Ideally, this function will be easily integrated with the current software system as an additional toolbox to analyze the results. The optode stability also greatly impacts the initial signal quality. Overall, improving the optode contact with the scalp will ultimately lead to higher SNR, which improves the hemodynamic signals as well. Our current optodes have some degree of stability however the channels are still contaminated with noise.

The current helmet design in the Imaginc group has fixed locations for each of the optodes, separated by a distance of 3 cm. Adjustable source-detector distances is being used in some systems and may be feasible in various test sessions. This would simply require a helmet design that provides multi-distance holes rather than fixed locations.

Data types and operations can greatly impact the performance of the program. Throughout our design, we have taken this into consideration and have designed code to avoid redundant and CPU-intensive operations. Loop polling was replaced by event structures and occurrences, to allow the CPU to process other data instead of being locked up in a particular operation. Most of the data processing is done on arrays of integers, and special attention was paid to pre-allocating array sizes and copying. Subroutines (subVIs) have been used widely throughout the program to modularize the code. We also discussed the benefits of multithreading, and allowing LabVIEW to perform the operation by minimizing the sequential flow of data as much as possible. This can provide a more efficient use of CPU resources under certain conditions. In a program with such complexity, there are still methods and data flows which can be optimized and improved. Future

versions of the program will require a more detailed analysis of the data structures and operations, to further optimize the program and enhance the performance.

The data acquisition in full mode (32 NIRS and 32 EEG channels) currently works in USB 2.0 mode only. Due to the amount of data that needs to be transferred, it is not possible to perform this using our current Bluetooth device based on the transfer rate limitation. However the Bluetooth functionality has been tested and verified using smaller amounts of data (16 NIRS and 32 EEG) and it functions properly. This causes a disadvantage since the auto calibration is performed in Bluetooth mode in a first step (only the NIRS data are sent), and the obtained configuration file is loaded in the second step while operating in USB. In most cases, multiple USB cables are connected (up to 3, for a total length of 5 meters) without affecting the speed. Using USB 3.0 does not provide any major advantage, since the cable length is still limited to around 3 meters, in addition to requiring the replacement of the USB serial adapter on the prototype. Ultimately, it would be much more convenient to perform all of this in one single step using Bluetooth. This requires using a Bluetooth module that allows a higher data transmission rate to support the operation in full mode. In most test settings, the range of operation of the Bluetooth device was sufficient (around 6 m) however in some tests it may be more useful to have a longer range.

In our program, in order to calculate the hemodynamic variations using the MBLL, the baseline value for each period was calculated as the average of the current time points, in the form of a cumulative average. An alternative method is to use the block average of the signal over a certain time period before and after the stimulus. This would require the user to define the trigger onsets, as well as the pre and post-time to include in the calculations before the acquisition starts, which implies that the stimulations are determined in advance. As new data is received, each block average will be calculated using the previous ones, to provide a progressive average value of all the blocks. In addition, the baseline value will be the value immediately before the stimulation, and not the baseline at the beginning of the test. This can be implemented as an additional Matlab library and integrated into the program. However this is only applicable in tests where the stimulus markers are known and predefined, such as finger tapping protocols. In situations where the goal is more focused on real-time monitoring of the data, block averaging cannot be applied, and our current system will continue to display the hemodynamic variations. In terms of pre-defining the events, it would be practical to include a file (or pop-up window) for

specifying the number and names of triggers, and their onset times and duration for experiments with known event occurrences. This would enable more precise event recording (compared to clicking the button) as the system would automatically generate the triggers. The trigger event files could also be saved and reused for subsequent test sessions, ensuring consistent results across subjects.

An important feature of imaging devices is the ability to map the location of the probes onto an image template. We propose two approaches; the first one is a basic model while the second is more detailed.

The configuration file used in our current system contains the emitters and their coupled detectors in a text file. To provide an overall view of the source-detector layout and a means of quick comparison between different test protocols, it would be practical to include a 2D grid (the size is equal to number of sources by number of detectors), indicating the relative positions of the source and detectors as coloured squares. The user would select the squares (channels) that are active in the experiment, along with the name (optional). This template would be saved along with all the other configurations settings. Another proposed solution is using the standard 10-10 montage layout as a template, where each position is a possible point for emitters and detectors. However this model is limited to the fixed locations and the probes cannot be placed at arbitrary positions. It would still allow the user to quickly review and compare the configuration of experiments.

A more advanced solution involves using a 3D digitizer to obtain the accurate location of the probes and map the data onto the brain template. In our current system, there is one configuration template for topographically projecting the hemodynamic states onto a model brain template. These spatial coordinates were obtained using a 3D digitizer and mapped onto the anatomical MR image using the Brainsight software tool (RogueResearch). This involves using reference points on the head (either based on the 10-20 system or capsule markers), to obtain the relationship between the MR coordinates and the real coordinates in the 3D digitizer in the form of x, y and z coordinates. In future developments, it would be feasible to integrate this feature into our system, to allow the spatial registration and data acquisition to be performed in one program. A short-term goal is to provide a series of templates, focusing on more commonly imaged areas of the brain that can be easily uploaded and used in displaying the topographical

image. In the long-term, this can be implemented either using the subjects own MRI database (if it is available) or to use a reference MRI image and register the channel coordinates onto the template. In our present system the data is mapped onto a 2D brain surface. Future improvements will include different topographic mappings, such as 3D head and 3D cortex surfaces. Depending on the level of detail required for analysis, there would be a "low" and "high" resolution option and the 3D map would also rotate, to view the brain from different angles. Our system currently records the temporal evolution of the topographic maps, as they are displayed onto the brain template. This file is saved in the project folder and can be played back for later analysis. An optional parameter is to define the time period of interest to record the data, instead of the entire test session. This can be useful in long term monitoring, where we are only interested in hemodynamic variations in a specific time interval. The default playback rate for the video is 15 frames per second (fps), however this could also be modified on the user interface.

In the long term, the brain imaging system will be distributed across clients and hospitals along with the developed software which requires both LabVIEW and Matlab licenses to run. One solution is to use the open source GNU Octave to replace Matlab. Matlab code is compatible with Octave, however GUI development in Octave is not as straight forward and simple compared to Matlab. Overall our program should function seamlessly in either platform, however there has been no testing or implantation done with Octave and it is only based on the known compatibility of the two. This will be a future design decision that will mainly depend on the end user's choice and preferences as well as the development team's resources and expertise.

CONCLUSION

In this thesis we presented the design and details of the real-time acquisition and analysis software system to support the existing hardware module for NIRS and EEG brain imaging. EEG measures the electromagnetic effects of the activation of neuron groups, while NIRS measures the subsequent vascular response to the activation. The combination of these two techniques provides complementary information on brain activity. In addition, the high temporal resolution and inexpensive solution make it ideal for real time monitoring and analysis

The software system has been developed using a combination of LabVIEW and Matlab development platforms. The overall system was designed as a layered architecture to separate the hardware communication layer from the rest of the program to have modular and easily manageable components. We took advantage of several general software architecture patterns and customized based on the requirements. Modifications or additions to the libraries can be easily made with minimum effort. This designed system is configured across two computers, one primary station that records the raw data and allows for acquisition parameter adjustment, and a secondary station that can be used for remote monitoring and displays the change in hemodynamic variations during the test session. The data for each of the NIRS and EEG channels are processed and displayed as they are acquired, providing live feedback on brain activity. Due to the natural variations in skin color, head size and hair density it is important to have a system with customizable parameters that can achieve the best results regardless of the individual subject settings. Automatic and manual adjustment of individual emitter illuminations, photodiode bias voltage, filter settings, and configuration files are among the features integrated in our program. This will ensure best possible results regardless of the variations that occur. The auto calibration setting greatly reduced the setup time and allowed for easy adjustment of the various intensities in less than a minute. In addition, automatic NIRS channels signal analysis throughout the session confirms a good optode connection, since the desired orthogonal optode placement may shift slightly during the session. This is a practical feature in long-term studies where the subject is not restrained from movement or daily activities. To reduce the overall setup time and avoid redundant configuration selection, previously defined NIRS and EEG configuration files can be uploaded and used. These files are in a simple text format that can be easily modified as needed. Current configurations can also be saved and reused later.

After receiving the raw NIRS signals, the hemodynamic variations are displayed on graphs as well as a 2D topographical map to provide better visual feedback of brain activity. The sequence of changes occurring during the session is also recorded as a movie file which can be played back later for detailed analysis of the results. There is always a risk of malfunction, whether it is Bluetooth interference with nearby devices, battery failure, or any other unexpected event. All acquired data is continuously saved on the laptop. This prevents loss of data in case of any connection loss or system malfunction. The file format is specific to LabVIEW however it can be easily converted to other formats that are compatible with Matlab and other analysis software through the program.

Another factor that was taken into consideration is the user interface design. Since our system will be used in a clinical setting, it is important to provide a simple and easy to use interface while offering full control over the experiment settings and parameters. This was achieved through user surveys and regular feedback from the collaborators at the hospital. Overall from a clinical point of view, the system was found to be practical and reliable in terms of signal quality, ease of use and features. In order to verify the system performance, data was acquired in a finger tapping task using 10 subjects. The hemodynamic variations were calculated and validated based on similar processing using the HomER toolbox. The results were found to be similar, with an average correlation of around 85.5% for HbO and HbR. Various other system metrics such as complexity and UI metrics were also discussed and analyzed.

The overall system (hardware and software) has been tested extensively to ensure correct and precise functionality. Most of the hardware features were tested independently, however the combination of the two was also considered. The software system was tested for long recording sessions and it was successfully able to record all data, which confirms the reliability of the system. However there are still certain limitations that need to be considered and improved in future versions. Optode stability and physiological noise directly impact the SNR of the NIRS data, which ultimately leads to less accurate hemodynamic activity mapping and data analysis. This can be improved by using better optode designs, as well as multiple short-distance channels to remove unwanted physiological interference. Data acquired from the accelerometers can also be integrated to reduce the effects of patient movement.

In terms of software and user interface features, we discussed future improvements and various additions, such as the option to perform block averaging, creation of trigger files and automatic event markers and obtaining probe locations and mapping onto 2D and 3D brain templates. A long term goal is to integrate a 3D digitizer module into the current software to obtain accurate probe locations and topographical projection of the hemodynamic states on the brain template. The time intervals for inclusion in the animation obtained from the temporal evolution of the maps, as well as the movie playback rate will also be adjustable parameters through the user interface.

The developed system can serve as a fundamental tool for the study and analysis of various neurophysiological phenomena. One particular long term goal is epileptic spike detection for seizure prediction. The information obtained from the EEG signals, along with the co-registration with hemodynamic activity, can be used to predict seizure onsets and take required measures to prevent or stop the progression. Other fields of application include language studies, sleep apnea and disorders, and rehabilitation. Thanks to the non-invasive approach, it can be used across a wide range of subjects, from premature neonates to the elderly. In addition, there are no restrictions on patient mobility and their environment, making it also an ideal candidate for outdoor studies and freedom in task design. The technique is relatively simple with no known side effects, and can be easily performed at the bed side. Technology loudness is an important factor in language studies, and with a silent system such as ours there are minimum disturbances throughout the test. Online evaluation of hemodynamic response during therapy sessions can serve as a guide to identify the most appropriate and effective rehabilitation program for individual subjects tailored to their needs. Overall it is a promising technology with potential applications in a wide range of research fields, providing significant insights into brain functionality.

BIBLIOGRPAHY

Antneuro. (2014). Asalab. Retrieved from http://www.ant-neuro.com/products/asa-lab

Blume, P. A. (2007). *The LabVIEW style book*: Pearson Education.

Boas, D. A., Dale, A. M., & Franceschini, M. A. (2004). Diffuse optical imaging of brain activation: approaches to optimizing image sensitivity, resolution, and accuracy. *Neuroimage, 23 Suppl 1* S275-288. doi: 10.1016/j.neuroimage.2004.07.011

Boas, D. A., Gaudette, T., Strangman, G., Cheng, X., Marota, J. J., & Mandeville, J. B. (2001). The accuracy of near infrared spectroscopy and imaging during focal changes in cerebral hemodynamics. *Neuroimage, 13*(1), 76-90. doi: 10.1006/nimg.2000.0674

Bozkurt, A., Rosen, A., Rosen, H., & Onaral, B. (2005). A portable near infrared spectroscopy system for bedside monitoring of newborn brain. *Biomedical engineering online, 4*(1), 29.

Cooper, R., Everdell, N., Enfield, L., Gibson, A., Worley, A., & Hebden, J. C. (2009). Design and evaluation of a probe for simultaneous EEG and near-infrared imaging of cortical activation. *Physics in medicine and biology, 54*(7), 2093.

Delpy, D. T., Cope, M., van der Zee, P., Arridge, S., Wray, S., & Wyatt, J. (1988). Estimation of optical pathlength through tissue from direct time of flight measurement. *Phys Med Biol, 33*(12), 1433-1442. Retrieved from http://www.ncbi.nlm.nih.gov/pubmed/3237772

Diamond, S. G., Perdue, K. L., & Boas, D. A. (2009). A cerebrovascular response model for functional neuroimaging including dynamic cerebral autoregulation. *Math Biosci, 220*(2), 102-117. doi: 10.1016/j.mbs.2009.05.002

Duncan, A., Meek, J. H., Clemence, M., Elwell, C. E., Tyszczuk, L., Cope, M., & Delpy, D. (1995). Optical pathlength measurements on adult head, calf and forearm and the head of the newborn infant using phase resolved optical spectroscopy. *Physics in medicine and biology, 40*(2), 295.

Essenpreis, M., Elwell, C. E., Cope, M., van der Zee, P., Arridge, S. R., & Delpy, D. T. (1993). Spectral dependence of temporal point spread functions in human tissues. *Appl Opt, 32*(4), 418-425. doi: 10.1364/AO.32.000418

Ferrari, M., Mottola, L., & Quaresima, V. (2004). Principles, techniques, and limitations of near infrared spectroscopy. *Can J Appl Physiol, 29*(4), 463-487. Retrieved from http://www.ncbi.nlm.nih.gov/pubmed/15328595

Ferrari, M., & Quaresima, V. (2012). A brief review on the history of human functional near-infrared spectroscopy (fNIRS) development and fields of application. *Neuroimage, 63*(2), 921-935. doi: 10.1016/j.neuroimage.2012.03.049

Fukui, Y., Ajichi, Y., & Okada, E. (2003). Monte Carlo prediction of near-infrared light propagation in realistic adult and neonatal head models. *Appl Opt, 42*(16), 2881-2887. Retrieved from http://www.ncbi.nlm.nih.gov/pubmed/12790436

Gagnon, L., Cooper, R. J., Yücel, M. A., Perdue, K. L., Greve, D. N., & Boas, D. A. (2012). Short separation channel location impacts the performance of short channel regression in NIRS. *Neuroimage, 59*(3), 2518-2528.

Gagnon, L., Yücel, M. A., Boas, D. A., & Cooper, R. J. (2014). Further improvement in reducing superficial contamination in NIRS using double short separation measurements. *Neuroimage, 85* 127-135.

Gallagher, A., Lassonde, M., Bastien, D., Vannasing, P., Lesage, F., Grova, C., . . . Nguyen, D. K. (2008). Non-invasive pre-surgical investigation of a 10 year-old epileptic boy using simultaneous EEG-NIRS. *Seizure, 17*(6), 576-582. doi: 10.1016/j.seizure.2008.01.009

Gallegos-Ayala, G., Furdea, A., Takano, K., Ruf, C. A., Flor, H., & Birbaumer, N. (2014). Brain communication in a completely locked-in patient using bedside near-infrared spectroscopy. *Neurology, 82*(21), 1930-1932. doi: 10.1212/WNL.0000000000000449

Giacometti, P., & Diamond, S. G. (2013). Diffuse optical tomography for brain imaging: continuous wave instrumentation and linear analysis methods. In *Optical Methods and Instrumentation in Brain Imaging and Therapy* (pp. 57-85): Springer.

Haas, L. (2003). Hans Berger (1873–1941), Richard Caton (1842–1926), and electroencephalography. *Journal of Neurology, Neurosurgery & Psychiatry, 74*(1), 9-9.

Haginoya, K., Munakata, M., Kato, R., Yokoyama, H., Ishizuka, M., & Iinuma, K. (2002). Ictal cerebral haemodynamics of childhood epilepsy measured with near-infrared spectrophotometry. *Brain, 125*(Pt 9), 1960-1971. Retrieved from http://www.ncbi.nlm.nih.gov/pubmed/12183342

Hebden, J. C., Cooper, R. J., Gibson, A., Everdell, N., & Austin, T. (2012). *Simultaneous EEG and diffuse optical imaging of seizure-related hemodynamic activity in the newborn infant brain.* Paper presented at SPIE Photonics Europe (pp. 84271N-84271N-84276).

Hemmati, N., Setarehdan, S. K., & Ahmadi Noubari, H. (2012). *Multi-channel Near-Infrared Spectroscopy (NIRS) system for noninvasive monitoring of brain activity.* Paper presented at Biomedical and Health Informatics (BHI), 2012 IEEE-EMBS International Conference on (pp. 212-215).

Hiraoka, M., Firbank, M., Essenpreis, M., Cope, M., Arridge, S. R., van der Zee, P., & Delpy, D. T. (1993). A Monte Carlo investigation of optical pathlength in inhomogeneous tissue and its application to near-infrared spectroscopy. *Phys Med Biol, 38*(12), 1859-1876. Retrieved from http://www.ncbi.nlm.nih.gov/pubmed/8108489

Huppert, T. J., Diamond, S. G., Franceschini, M. A., & Boas, D. A. (2009). HomER: a review of time-series analysis methods for near-infrared spectroscopy of the brain. *Appl Opt, 48*(10), D280-298. Retrieved from http://www.ncbi.nlm.nih.gov/pubmed/19340120

Ian, S. (2011). software engineering, 9 th education: Addison–Wesley, Ed.

Irani, F., Platek, S. M., Bunce, S., Ruocco, A. C., & Chute, D. (2007). Functional near infrared spectroscopy (fNIRS): an emerging neuroimaging technology with important applications for the study of brain disorders. *The Clinical Neuropsychologist, 21*(1), 9-37.

Ito, Y., Kennan, R. P., Watanabe, E., & Koizumi, H. (2000). Assessment of heating effects in skin during continuous wave near infrared spectroscopy. *Journal of biomedical optics, 5*(4), 383-390.

Jobsis, F. F. (1977). Noninvasive, infrared monitoring of cerebral and myocardial oxygen sufficiency and circulatory parameters. *Science, 198*(4323), 1264-1267. Retrieved from http://www.ncbi.nlm.nih.gov/pubmed/929199

Kassab, A. (2014). *The design of a NIRS cap for long-term brain activity monitoring.* (ECOLE POLYTECHNIQUE DE MONTREAL, ECOLE POLYTECHNIQUE DE MONTREAL).

Kocsis, L., Herman, P., & Eke, A. (2006). The modified Beer-Lambert law revisited. *Phys Med Biol, 51*(5), N91-98. doi: 10.1088/0031-9155/51/5/N02

Koizumi, H., Yamamoto, T., Maki, A., Yamashita, Y., Sato, H., Kawaguchi, H., & Ichikawa, N. (2003). Optical topography: practical problems and new applications. *Applied optics, 42*(16), 3054-3062.

Lareau, E., Lesage, F., Pouliot, P., Nguyen, D., Le Lan, J., & Sawan, M. (2011). Multichannel wearable system dedicated for simultaneous electroencephalographynear-infrared spectroscopy real-time data acquisitions. *J Biomed Opt, 16*(9), 096014. doi: 10.1117/1.3625575

Leamy, D. J., Collins, R., & Ward, T. E. (2011). Combining fNIRS and EEG to improve motor cortex activity classification during an imagined movement-based task. In *Foundations of Augmented Cognition. Directing the Future of Adaptive Systems* (pp. 177-185): Springer.

Lee, R. Y. (2013). Modeling with UML. In *Software Engineering: A Hands-On Approach* (pp. 39-58): Springer.

LeLan, J. (2013). *Prototype d'imagerie cerebrale multicanale portable par spectroscopie proche-infrarouge et electroencephalographie.* (ECOLE POLYTECHNIQUE DE MONTREAL, ECOLE POLYTECHNIQUE DE MONTREAL).

Li, Z., Zhang, M., Xin, Q., Chen, G., Liu, F., & Li, J. (2012). Spectral analysis of near-infrared spectroscopy signals measured from prefrontal lobe in subjects at risk for stroke. *Medical physics, 39*(4), 2179-2185.

Lopez-Gordo, M., Sanchez-Morillo, D., & Valle, F. P. (2014). Dry EEG Electrodes. *Sensors, 14*(7), 12847-12870.

Matthews, F., Soraghan, C., Ward, T. E., Markham, C., & Pearlmutter, B. A. (2008). Software platform for rapid prototyping of NIRS brain computer interfacing techniques. *Conf Proc IEEE Eng Med Biol Soc, 2008* 4840-4843. doi: 10.1109/IEMBS.2008.4650297

Nguyen, D. K., Tremblay, J., Pouliot, P., Vannasing, P., Florea, O., Carmant, L., . . . Lassonde, M. (2012). Non-invasive continuous EEG-fNIRS recording of temporal lobe seizures. *Epilepsy Res, 99*(1-2), 112-126. doi: 10.1016/j.eplepsyres.2011.10.035

Nguyen, D. K., Tremblay, J., Pouliot, P., Vannasing, P., Florea, O., Carmant, L., . . . Lassonde, M. (2013). Noninvasive continuous functional near-infrared spectroscopy combined with electroencephalography recording of frontal lobe seizures. *Epilepsia, 54*(2), 331-340. doi: 10.1111/epi.12011

Nunez, P. L., & Srinivasan, R. (2006). *Electric fields of the brain: the neurophysics of EEG*: Oxford university press.

Obrig, H. (2014). NIRS in clinical neurology—a 'promising'tool? *Neuroimage, 85* 535-546.

Obrig, H., Neufang, M., Wenzel, R., Kohl, M., Steinbrink, J., Einhaupl, K., & Villringer, A. (2000). Spontaneous low frequency oscillations of cerebral hemodynamics and metabolism in human adults. *Neuroimage, 12*(6), 623-639. doi: 10.1006/nimg.2000.0657

Obrig, H., & Villringer, A. (2003). Beyond the visible--imaging the human brain with light. *J Cereb Blood Flow Metab, 23*(1), 1-18. Retrieved from http://www.ncbi.nlm.nih.gov/pubmed/12500086

Osharina, V., Ponchel, E., Aarabi, A., Grebe, R., & Wallois, F. (2010). Local haemodynamic changes preceding interictal spikes: a simultaneous electrocorticography (ECoG) and near-infrared spectroscopy (NIRS) analysis in rats. *Neuroimage, 50*(2), 600-607. doi: 10.1016/j.neuroimage.2010.01.009

Payne, S. J., Selb, J., & Boas, D. A. (2009). Effects of autoregulation and CO2 reactivity on cerebral oxygen transport. *Ann Biomed Eng, 37*(11), 2288-2298. doi: 10.1007/s10439-009-9763-5

Pellicer, A., & Bravo Mdel, C. (2011). Near-infrared spectroscopy: a methodology-focused review. *Semin Fetal Neonatal Med, 16*(1), 42-49. doi: 10.1016/j.siny.2010.05.003

Peng, T., Ainslie, P. N., Cotter, J. D., Murrell, C., Thomas, K., Williams, M. J., . . . Payne, S. J. (2008). The effects of age on the spontaneous low-frequency oscillations in cerebral and systemic cardiovascular dynamics. *Physiol Meas, 29*(9), 1055-1069. doi: 10.1088/0967-3334/29/9/005

RogueResearch.). Retrieved from https://www.rogue-research.com/

Safaie, J., Grebe, R., Abrishami Moghaddam, H., & Wallois, F. (2013). Toward a fully integrated wireless wearable EEG-NIRS bimodal acquisition system. *J Neural Eng, 10*(5), 056001. doi: 10.1088/1741-2560/10/5/056001

Scholkmann, F., Kleiser, S., Metz, A. J., Zimmermann, R., Mata Pavia, J., Wolf, U., & Wolf, M. (2014). A review on continuous wave functional near-infrared spectroscopy and imaging instrumentation and methodology. *Neuroimage, 85 Pt 1* 6-27. doi: 10.1016/j.neuroimage.2013.05.004

Scholkmann, F., & Wolf, M. (2013). General equation for the differential pathlength factor of the frontal human head depending on wavelength and age. *Journal of biomedical optics, 18*(10), 105004-105004.

Shimadzu Corporation. (2014). The LABNIRS™ fNIRS system. Retrieved from http://www.neurospec.com/?p=prodresearch_nirs_foire

Sliney, D., Aron-Rosa, D., DeLori, F., Fankhauser, F., Landry, R., Mainster, M., . . . Trokel, S. (2005). Adjustment of guidelines for exposure of the eye to optical radiation from ocular instruments: statement from a task group of the International Commission on Non-Ionizing Radiation Protection (ICNIRP). *Applied optics, 44*(11), 2162-2176.

Sokol, D. K., Markand, O. N., Daly, E. C., Luerssen, T. G., & Malkoff, M. D. (2000). Near infrared spectroscopy (NIRS) distinguishes seizure types. *Seizure, 9*(5), 323-327. doi: 10.1053/seiz.2000.0406

Soraghan, C., Matthews, F., Markham, C., Pearlmutter, B. A., O'Neill, R., & Ward, T. E. (2008). A 12-channel, real-time near-infrared spectroscopy instrument for brain-computer

interface applications. *Conf Proc IEEE Eng Med Biol Soc, 2008* 5648-5651. doi: 10.1109/IEMBS.2008.4650495

Strangman, G., Goldstein, R., Rauch, S. L., & Stein, J. (2006). Near-infrared spectroscopy and imaging for investigating stroke rehabilitation: test-retest reliability and review of the literature. *Arch Phys Med Rehabil, 87*(12 Suppl 2), S12-19. doi: 10.1016/j.apmr.2006.07.269

Taga, G., Asakawa, K., Maki, A., Konishi, Y., & Koizumi, H. (2003). Brain imaging in awake infants by near-infrared optical topography. *Proc Natl Acad Sci U S A, 100*(19), 10722-10727. doi: 10.1073/pnas.1932552100

Terborg, C., Groschel, K., Petrovitch, A., Ringer, T., Schnaudigel, S., Witte, O. W., & Kastrup, A. (2009). Noninvasive assessment of cerebral perfusion and oxygenation in acute ischemic stroke by near-infrared spectroscopy. *Eur Neurol, 62*(6), 338-343. doi: 10.1159/000239794

Toronov, V., Franceschini, M. A., Filiaci, M., Fantini, S., Wolf, M., Michalos, A., & Gratton, E. (2000). Near-infrared study of fluctuations in cerebral hemodynamics during rest and motor stimulation: temporal analysis and spatial mapping. *Med Phys, 27*(4), 801-815. Retrieved from http://www.ncbi.nlm.nih.gov/pubmed/10798703

Tsuzuki, D., Cai, D.-s., Dan, H., Kyutoku, Y., Fujita, A., Watanabe, E., & Dan, I. (2012). Stable and convenient spatial registration of stand-alone NIRS data through anchor-based probabilistic registration. *Neuroscience research, 72*(2), 163-171.

Van der Zee, P., Cope, M., Arridge, S., Essenpreis, M., Potter, L., Edwards, A., . . . Reynolds, E. (1992). Experimentally measured optical pathlengths for the adult head, calf and forearm and the head of the newborn infant as a function of inter optode spacing. In *Oxygen Transport to Tissue XIII* (pp. 143-153): Springer.

Wabnitz, H., Moeller, M., Liebert, A., Obrig, H., Steinbrink, J., & Macdonald, R. (2010). Time-resolved near-infrared spectroscopy and imaging of the adult human brain. *Adv Exp Med Biol, 662* 143-148. doi: 10.1007/978-1-4419-1241-1_20

Wallois, F., Patil, A., Héberlé, C., & Grebe, R. (2010). EEG-NIRS in epilepsy in children and neonates. *Neurophysiologie Clinique/Clinical Neurophysiology, 40*(5), 281-292.

Wang, Q., Liang, X., Liu, Z., Zhang, Q., Carney, P., & Jiang, H. (2008). Visualizing localized dynamic changes during epileptic seizure onset in vivo with diffuse optical tomography. *Med Phys, 35*(1), 216-224. Retrieved from http://www.ncbi.nlm.nih.gov/pubmed/18293577

Watanabe, E., Nagahori, Y., & Mayanagi, Y. (2002). Focus diagnosis of epilepsy using near-infrared spectroscopy. *Epilepsia, 43 Suppl 9* 50-55. Retrieved from http://www.ncbi.nlm.nih.gov/pubmed/12383281

Wolf, M., Ferrari, M., & Quaresima, V. (2007). Progress of near-infrared spectroscopy and topography for brain and muscle clinical applications. *J Biomed Opt, 12*(6), 062104. doi: 10.1117/1.2804899

Zhang, Y., Franceschini, M. A., Boas, D. A., & Brooks, D. H. (2005). Eigenvector-based spatial filtering for reduction of physiological interference in diffuse optical imaging. *Journal of biomedical optics, 10*(1), 011014-01101411.

Zijlstra, W. G., Buursma, A., & van Assendelft, O. W. (2000). *Visible and near infrared absorption spectra of human and animal haemoglobin: determination and application*: VSP.