AUTOMATIC POSTPROCESSING AND REPORTING TOOLS FOR FUNCTIONAL NEUROIMAGING

by

Engin Demirel

B.S, Electrical and Electronics Engineering, Boğaziçi University, 2003

Submitted to the Institute of Biomedical Engineering

in partial fulfillment of the requirements

for the degree of

Master of Science

in

Biomedical Engineering

Boğaziçi University

2010

ACKNOWLEDGMENTS

Firstly I am greatly thankful to my thesis advisor Cengizhan Öztürk, for his continuous support, motivation and guidance throughout the course of this study. His efforts were very valuable in reaching the goals of this study and planning the work always one step ahead.

I want to send my special thanks to Onur Özyurt, who has led me to the Python programming world, which was a milestone in the development of the software. His continuous support, technical assistance, creative ideas helped this study become a clinically useful program. I also like to thank to Ertugrul Akbaş, for his efforts to combine our studies in a wider platform. I also like to thank to Alp Dinçer for his guidance in clinical routines of fMRI and planning of the modules that would be helpful in real life use.

I would like to express my special thanks to my father Emin, my mother Gülfiye and my sister Esin who have always wished the best for me and encouraged me. I am so lucky to have such a great family.

Lastly I dedicate this study to my lovely wife Nuran, and my daughter Defne. You are my everything.

ABSTRACT

AUTOMATIC POSTPROCESSING AND REPORTING TOOLS FOR FUNCTIONAL NEUROIMAGING

For advanced magnetic resonance imaging (MRI) applications, reporting and postprocessing tools have been designed. The software tools developed consist of DICOM structured report (DICOM SR) preparation, functional image overlay onto anatomical standard atlases, lateralization calculations, maximum Z value and location analysis and mean Z value analysis for cortical and subcortical brain regions.

Radiologists can view the functional image overlayed onto the anatomical ones, visualize the activated regions and edit DICOM SR accordingly with the image references. Lateralization calculation is automatically done and displayed on the user interface. Maximum Z value is found and shown on standard anatomical atlas in axial, sagital and coronal planes with the waveform of the maxima through the timepoints of the data analyzed. The software finally provides the user with DICOM SR and rich text format (RTF) output including edited DICOM SR text fields, referenced images, lateralization or maximum Z value calculations.

Keywords: DICOM SR, Lateralization, Z score, Reporting, Neuroimaging.

ÖZET

FONKSİYONEL NÖROGÖRÜNTÜLEME İÇİN OTOMATİZE POSTPROCESSING VE RAPORLAMA ARAÇLARI

İleri düzey Manyetik Rezonans görüntüleme (MRG) uygulamaları ve araştırmaları için raporlama ve postprocessing araçları geliştirilmiştir. Geliştirilmiş yazılım araçları DICOM yapısal rapor hazırlama, fonksiyonel imajların standard anatomik atlaslar üzerine çakıştırılması, lateralizasyon hesaplamaları, maksimum Z değeri ve konumu analizi ve kortikal ve subkortikal beyin bölgeleri için ortalama Z değeri hesaplamadan oluşmaktadır.

Radyologlar anatomik atlas üzerine yerleştirilmiş fonksiyonel görüntüleri görüntüleyebilmekte, aktivasyon bulunan alanları incelemekte ve DICOM SR formatındaki raporu ilgili imaj referanslarıyla birlikte yazabilmektedir.Maksimum Z değeri bulunmakta, standard anatomik atlas üzerinde aksiyel, sagital ve koronal düzlemlerde gösterilmekte ve maksimum değerin bulunduğu lokasyonun tüm zaman noktaları boyunca dalga formu çizdirilmektedir. Geliştiren yazılım sonuç olarak kullanıcıya DICOM SR ve zengin metin dosyası formatında, yazılan SR text alanlarını, refere edilen imajları ve maksimum Z değeri hesaplamalarını içeren çıktı vermektedir.

Anahtar Sözcükler: DICOM SR, Lateralizasyon, Z skoru, Raporlama, Nörogörüntüleme.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS

y(t)	1D vector of intensity values
x(t)	1D vector for each time points
β	Estimation Parameters
$\overrightarrow{X}$	Time course matrix representation
$\overrightarrow{\beta}$	parameter matrix representation

LIST OF ABBREVIATIONS

MR	Magnetic Resonance
MRI	Magnetic Resonance Imaging
DICOM	Digital Imaging and Communications in Medicine
SR	Structured Report
BUMIL	Bogazici University Medical Imaging Laboratory
GUI	Graphical User Interface
fMRI	Functional Magnetic Resonance Imaging
ROI	Region of Interest
T	Tesla
EPI	Echo Planar Imaging
CBO	Cerebral Blood Oxygenation
BOLD	Blood Oxygenation Level Dependent
HR	Hemodynamic Response
GLM	General Linear Model
1D	One Dimensional
EV	Explanatory Variable
HRF	Hemodynamic Response Function
PE	Parameter Estimate
GRF	Gaussian Random Field
MNI	Montreal Neurological Institute
LI	Lateralization Index
CT	Computed Tomography
ACR	American College of Radiology
NEMA	National Electrical Manufacturers Association
PACS	Picture Archiving and Communication Systems
FAT16	PC File System
IOD	Information Object Definition
ISO	International Organization for Standardization
TCP/IP	Transmission Control Protocol/Internet Protocol

ECR	European Congress of Radiology
RSNA	Radiological Society of North America
FMRIB	fMRI of the Brain
FSL	FMRIB Software Library
AFNI	Analysis of Functional NeuroImages
GPL	General Public License
NifTi	Neuroimaging Informatics Technology Initiative

1. INTRODUCTION

1.1   Motivation

Due to the lack of automated tools and long postprocessing times, supporting radiologists who are employing advanced neuroimaging techniques in magnetic resonance imaging (MRI) is still facing a challenge. Custom third party software is used but their reports are kept seperate from the images. As an alternate, Digital Imaging and Communications in Medicine (DICOM) Structured Report (SR) format has the potential to be used as a reporting tool for neuro MRI since reports can be edited from predefined templates with references to images as best illustration of findings. The images with functional mappings are as important, if not more, to the radiologists as the reports.

At Bogazici University Medical Imaging Laboratory (BUMIL) we are designing a communication platform and technical support software with several analysis modules for radiologists working on Neuro MRI. New modules are designed to be fully integrated with each other to seamless transition from the beginning of the analysis to the final report. This study consists of developing the reporting and postprocessing software, graphical user interface (GUI) and development of visualization tools for a completely automated functional magnetic resonance imaging (fMRI) analysis to be used within this platform. Labeling of regions in functional images by the Harvard-Oxford cortical and Subcortical anatomical atlasses, locating active region and quantitative analysis such as lateralization calculation and maximum z point location on functional data have been integrated within this module.

Past approaches for reporting MRI has been in plain text format where links to the images essential for the diagnosis and the display of region of interests (ROIs) in the images are missing. This approach is not practical in fMRI where images are the most important component in the decision making. The report has to contain references to the images and the significant part of this thesis work involves design of a GUI for

reporting of structured text that is DICOM SR compliant.

1.2 The Scope of the Study

The aim of this study is to develop reporting and postprocessing tools for the neuroimaging studies that will support radiologists, postprocessing the fMRI images with developed tools and to display the results as an output in the standard DICOM SR format that can be also loaded to DICOM archive, so they are viewable across platforms, and printable.

The software was developed step by step, module by module according to the feedbacks and guidance taken from Alp Dincer, MD. from Acibadem Kozyatagi Hospital, who is collaborating with BUMIL on many neuroimaging studies.

The problems and obstacles in real clinical life that is making radiologist's life difficult during a neuro MRI study report preperation was analyzed and a graphical user interface with background functions of image overlay, anatomical brain region detection, laterization calculations to be used for linguistic fMRI studies and maximum z-score calculations to be utilized in motor activity fMRI studies are developed.

The tools used for the analysis has a final report output with the selection of the radiologist's images selected that are meaningful in diagnosis and a standard z-score reporting or laterization report automatically taken from the analysis. This software is planned to be used in Acibadem Hospitals initially as beta testing and to analyze further development options beyond this thesis study.

As development platform, Python based programming Eric 4 (Integrated Development Environment for the Python programming language) was chosen and GUI design is made with Glade 3.4.5 (A user interface designer for GTK+) both running on Ubuntu 804 Linux operating system [5, 6, 7].

1.3 Outline of the Thesis

The remaining chapters are organized as follows: Chapter 2, gives information about the background and theory of the study; consisting of MRI, fMRI, DICOM and communication standards, DICOM SR format, SR editing based on tag element modification and containers, development platform and current state of implementations available for clinicians.

Chapter 3 briefly explains the GUI design phase of this study including the methodology used to edit and modify the SR content, the aim for using SR format, tag analysis and report creation method and a brief description of SR preparation for neuro MRI studies with emphasis on report content.

Chapter 4 explains the development of visualization tools, labeling voxels/regions in fMRI images using anatomical atlas and associates them with the SR creation routines explained in Chapter 3.

In Chapter 5, the lateralization index and maximum z-score calculations essential for the final report are introduced, regional z value analysis and the report preperation in DICOM SR and rich text file format is explained as the output of the software developed.

Finally Chapter 6 is a summary of the conclusions and recommendations for future work.

2. BACKGROUND AND THEORY

2.1   Magnetic Resonance Imaging of the Brain

The brain looks like a very simple organ when looked from outside, at its surface a grey and uniform appearance. With the help of medical imaging, specifically MRI, a rich internal structure can be visualized. A 1.5 Tesla (T) MRI can easily provide a detailed picture of the brain with an isotropic resolution of less than 0.5 mm. Even structures located deep in the centre of the brain could be seen. A rich network of arteries, some of which are located deep into the white matter of the brain and are very small in diameter, could also be visualized with MRI.

More than seventy percent of the brain is composed of water similar to the other tissues in the body. Hence, searching for the water distribution in the head is just similar to localise the brain tissue. Different regions of the brain have slightly different percentage of water. For example neurons are quite rich in water, but the fatty coating around the long nerve fibres, called myelin, has less. This creates contrast between the surface cortex and the white matter of the brain that can be exploited to provide full details of structure by the technique of MRI as apperant in Figure 2.1.

2.2   Examples on Mapping Brain Functions

The basic apperance of the surface of the brain and its hidden anatomical location delayed scientists to investigate its functions until quite late in the medicine history. After its importance have begun to be understood from 17th century on, the underlying functions of the brain started to be discussed among scientists. In accordance with the uniform appearance of the brain, most of the scientists were supporting the theory of mass action, which stated that regions of the brain act together for every task, without much regional functional specialization [8].

On the contrary, the phrenology school of the Viennese physician Thomas Gall
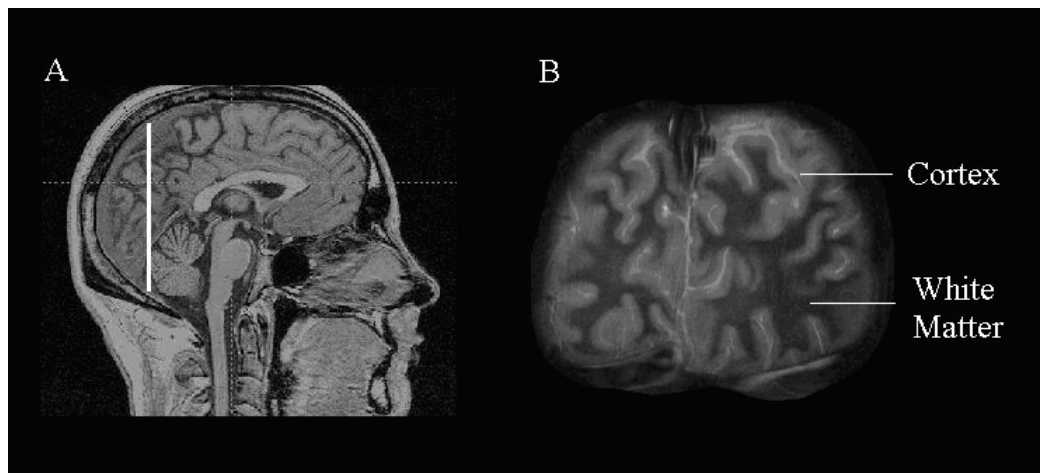
Figure 2.1 A. A sagittal MRI image from a healthy brain. Different structures in the brain can be defined with exquisite detail. B. MRI imaging contrast is based primarily on the distribution of water and differences in its physical environment in different tissues [1].

proposed that particular types of brain function were related with specific regions of the brain that could be localised by hitting the overlying bumps on the surface of the skull. The idea that specific regions of the brain acts as functional modules became the basis for the concept of discreet cortical localization. Gall and his collaborators however further assumed that the structure of the brain was linked directly to character, with individual brain regions determining individual traits. This theory was proved to be incorrect. However, the idea of cortical functional localisation is still supported by contemporary science, although currently it is known that certain types of functions are being performed by specific brain regions.

Phrenology led to a new localised methodology to map the brain based on lesion studies. It is based on the observation that damage to different areas of the brain caused specific disabilities in patients. One of the most known 19th century case was a patient studied by French neurologist Charcot and known to us as "Tan" because this was the only sound that he could pronounce. Charcot was impressed that Tan's disability to speak was not about a mental deficiency, a problem of motion, or handicap of linguistic ability. When the patient finally died, Charcot found that a single big tumour was damaging the lower left side of the front of the brain. He then theorized that this area of the brain has a specific function for language [9].

From a practical perspective it is highly unlikely that a lesion or a tumour is

confined only to one discreet brain region. Most lesions or tumours spill out into adjacent regions, making it harder to evaluate the functional deficit. Additionally abnormal brain tissue sometimes causes changes in the functioning of adjacent normal tissue indirectly.

A new methodology where the brain is evaluated as structurally specialised for sensation, cognition and direction of action is being developed with utilization of MRI to detect small differences in the healthy living brain that can be related to behaviour. There are recently exciting examples of studies in which small, local variations in brain shape are correlated directly with changes in brain function. As an example, it has been reported that there is a relationship between the size of the brain region which controls hand function and the number of years of practice of a musical skill, suggesting that the structural variability in this brain region occurs as a consequence of experience or use rather than reflecting individual differences within the population [10]. Also a recent study reported that London taxi drivers have an increased size of the part of the brain involved in the type of memory used for map reading [11].

2.3 Functional Magnetic Resonance Imaging

fMRI is a relatively recent imaging technique with the primary goal to designate the neurobiological correlation of behaviour, by identifying the functional brain regions in vivo that become active during the performance of specific tasks. The non-invasive and safe character of the technique let the studies to be repeated within a given subject so that important neurobiological questions, such as the relationship between experience-dependent changes in the brain function could be addressed. In order to better understand how fMRI provides spatial and temporal information on brain function, it would be beneficial to investigate further the brain [12]. The basic functional unit of the brain is an individualized cell, called a neuron. A neuron has a cell body, which has short extensions called dendrites that receive message inputs from other neurons and then give the information back to the cell body. Neuron also has a long extension called an axon that conducts electrical impulses from the cell body towards

the dendrites of other neurons. The region where the axon from one neuron and the dendrites form a contact gap is called a synapse. When electrical impulses travelling down the first axon reach a certain threshold they cause the release of local chemical neurotransmitters from the end of the axon into the synapse and onto receptors on the nearby dendrites of the neighbouring neuron. The contact of neurotransmitter with these receptors may then trigger the transmission of a nerve impulse down the second neuron and then onto further neurons until that a large number are activated coherently in a functional network of neurons [13]. Essential for fMRI, after the neurotransmitters have been released into the synapse, they are recycled and reenter the neuron in a process that requires energy. fMRI makes indirect use this energy necessity to image the regions of the brain that are part of the activated neural network [12, 14].

The local increase in energy requirements arising as a consequence of neuronal firing is largely met through an increase in oxygen-based metabolism with the increased demand for oxygen being delivered seconds later by an increase in the local blood flow which is called as the hemodynamic response [15]. Changes in the oxygenation level of the blood therefore occur as a consequence of neuronal activity and so the magnitude of change in signal intensity can be used as an indirect measure of excitatory input to neurons which is generally related closely to the cell firing rate [16, 17, 18]. This effect can be used to precisely map areas of the brain involved in brain function. Water molecules in the brain change slightly between regions that are near blood with its oxygen exhausted relative to those near freshly oxygenated blood [13].

In Echo Planar Imaging (EPI), the level of cerebral blood oxygenation (CBO) affects the signal intensity of T2*-weighted gradient-echo MR images. Deoxygenated hemoglobin can function as an endogeneous paramagnetic contrast agent. Physical effect is loss of signal when MRI detection combines appropriate magnetizations over an image pixel. The degree of signal loss is dependent on the absolute concentration of deoxyhemoglobin per voxel and magnitude changes with voxel size and gradient-echo time. CBO or blood oxygenation level dependent (BOLD) contrast can be improved by acquiring the images at prolonged echo times of, for example, 30 to 50 ms [19].
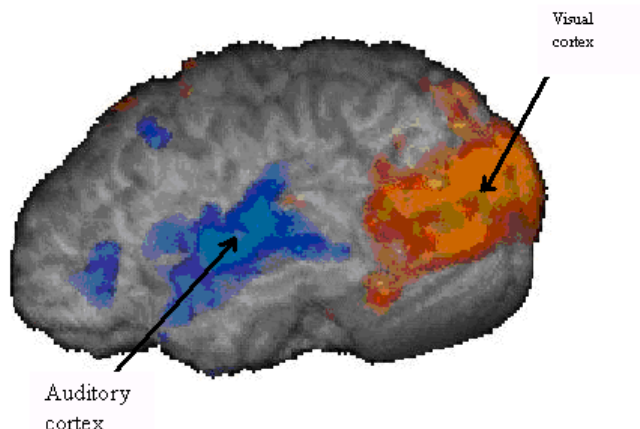
Figure 2.2 Brain auditory and visual cortex regions [1].

If the MRI experiment is done during a mental task given to a subject, a functional magnetic resonance image is obtained. On fMRI images it can be seen how different tasks activate different regions of the brain. For example, it is detected that when listening to music, a specialised area in the auditory cortex along the sides of the brain shows increased signal [20]. Vision activates a region in the back of the brain, namely the occipital cortex, localised exactly to regions of the visual field [21]. Touch brings increased signal along the side of the brain, particularly in the side of the brain opposite to the part of the body that is touched and movements activate regions in the front and the top of the brain in cortex specialised for motor control [22, 23].

As an example, healthy subject was asked to listen to sentences being spoken while watching a screen with a flashing checkerboard presented. The sentences started and stopped at slightly different times than the flashing picture was turned on and off. On the basis of the differences in timing activation in the brain, the areas responsible for hearing (in the middle of the brain in grey) and vision (in the back of the brain in white) could be localised by fMRI as seen in Figure 2.2 [1].

Functional MRI could identify the brain regions that become more activated during specific task carried. However, there are several obstacles to interpretation of the increase in fMRI signal. It is possible that the magnitude and scope of the hemodynamic response is a measure of more than a single neural process that requires
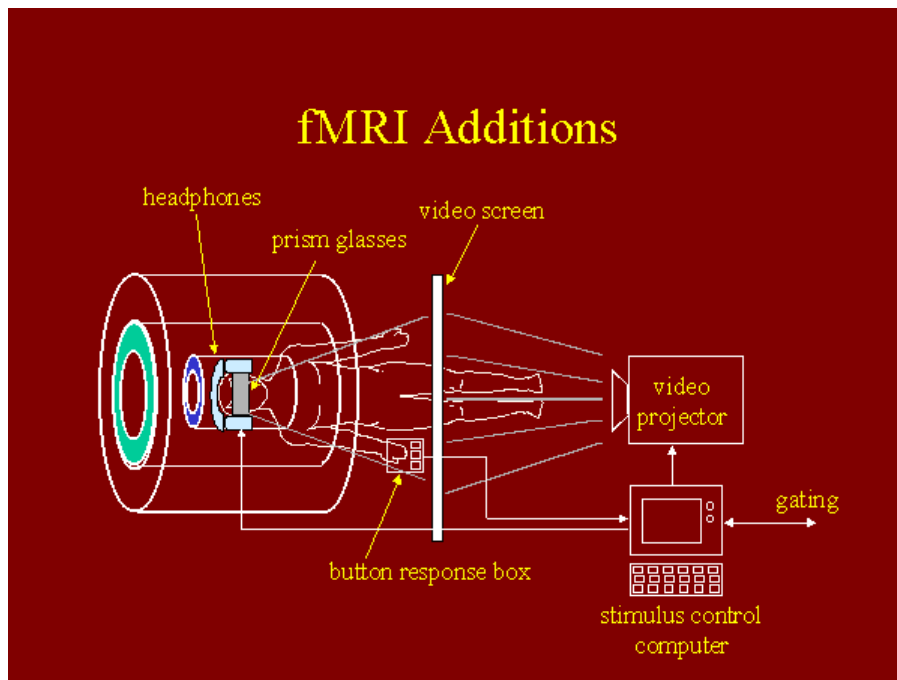
Figure 2.3 fMRI additions for procedure setup [2].

energy. It may reflect not only the frequency of local excitatory synaptic input, but also the extent of the post-synaptic depolarization (neuronal firing) [17]. Although it can be less energy requiring, the firing of inhibitory neurons also could add on to an increase in local energy requirement and related hemodynamic response [24, 25].

For fMRI experiments, the time-course of the response is not the most critical factor. A standard method of task presentation as also presented in Figure 2.3 and 2.4, Paradigm or "block" design where blocks of the stimulus or task are presented typically for 20-30 seconds, alternating with periods of rest or a control condition is applied. This control task is especially selected such that it activates all of the neural processes common to the stimuli task except the cognitive process of experiment. By subtracting the brain regions obtained during doing the control task from the brain regions obtained during the experiment condition, the regions of the brain whose activity is relevant to the cognitive process of interest can be detected.

There are customized hardware solutions for stimulus delivery and software system components for experimental control in fMRI studies as seen in Figure 2.3. These systems are designed for behavioral and physiological experiments that collect fMRI,
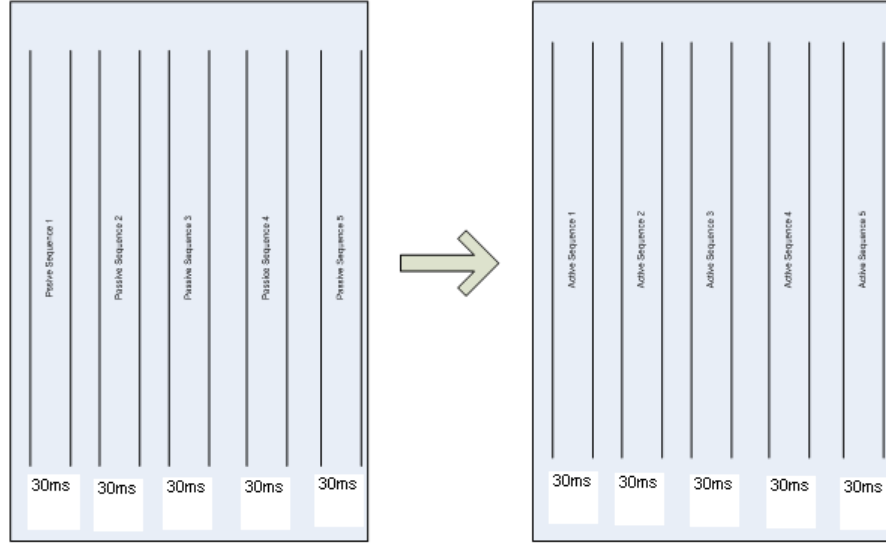
Figure 2.4 Paradigm block design.

reaction time, and electrophysiological data. These automated systems are designed to provide the best possible timing accuracy and timing verification on standard hardware. The setup also provides visual stimuli, auditory stimuly and has reaction devices like buttons. Hence fMRI synchronization and control of stimulus presentation in real-time is provided.

A block design based fMRI data acquisition is displayed in Figure 2.4, consequently taking T2* weighted images with EPI sequence as a block design. Sequences are identical, active and passive ones are only used for post-processing phase. fMRI process consists of four main components of stimuli presentation, Data Acquisition, Data Transfer and Analysis, and Postprocessing and Reporting as seen in Figure 2.5.

2.4 fMRI Analysis

fMRI can give good visualization of the regions of activity in the brain resulting from sensory stimulations or cognitive functions. After an fMRI experiment has been performed, the resulting data must be postprocessed with some analysis steps before the answers to questions about experimentally related activations can be obtained. This section gives a brief explanation of the most commonly used analysis covering data preprocessing, general linear model and activation thresholding [26].
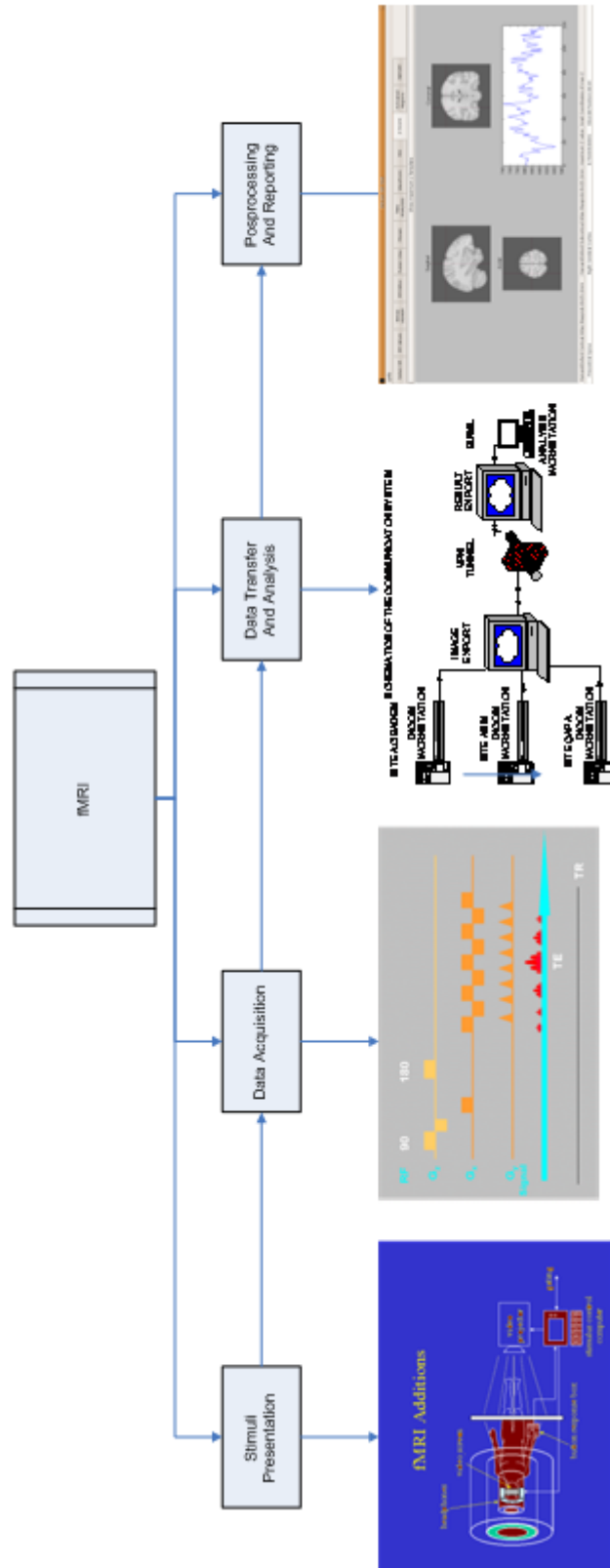
Figure 2.5 fMRI Workflow.
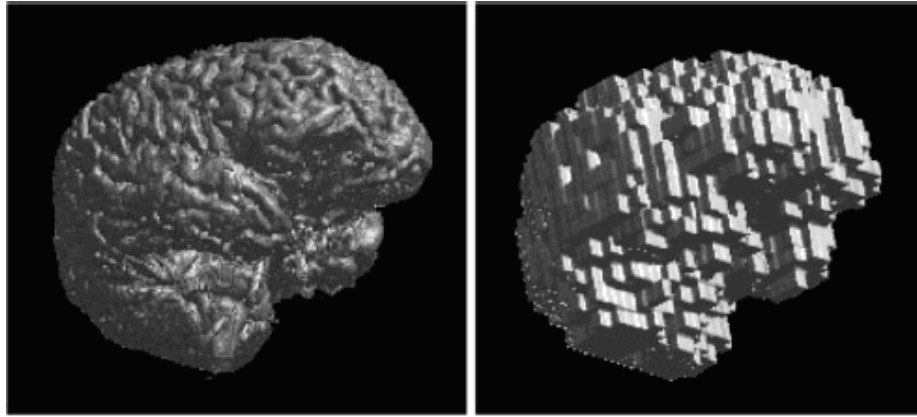
Figure 2.6 Voxels. Surface renderings of 3D brain images. On the left is a high-resolution image, with small (1 mm x 1 mm x 1.5 mm) voxels. On the right is a low-resolution image of the same brain, with large (7 mm x 7 mm x 10 mm) voxels [3].

2.4.1   fMRI Data Acquisition

In a typical fMRI session a low-resolution functional volume is acquired every 2-3 seconds. During the experiment, typically more than hundred volumes are acquired. In simple experiments, some images are taken while stimulation is applied, and some will be taken with the subject at rest state. Since the images are acquied using an MRI sequence which is sensitive to differentiations in BOLD signal, segments of the images acquired during stimulation typically will have increased intensity, compared with the ones taken at rest state [26]. These parts with increased intensity should correspond to the brain regions which are activated by the stimulation. The aim of fMRI analysis is to detect these regions.

A single volume is made up of individual cubic elements called voxels as seen in Figure 2.6. An fMRI data set from a single session can either be represented as t volumes where each volume consists of measurements taken from v voxels.

An example of a time series from an individual voxel is shown in Figure 2.7. Image intensity is depicted on the y axis, and time on the x axis. As usual for fMRI, for varying period of time stimulation was applied, and at some time points the subject was at rest state. The effect of the stimulation is observed and the high frequency noise is also seen. The goal of fMRI analysis is to determine in which voxels' time series the signal is significantly greater than the noise level [3].
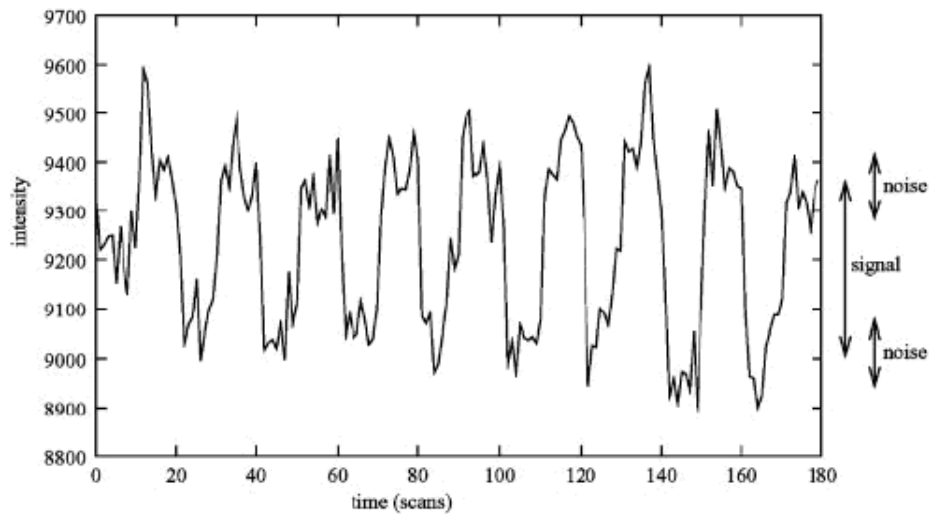
Figure 2.7 An example time series at a strongly activated voxel from a visual stimulation experiment. The signal is significantly larger than the noise level. Periods of stimulation are alternated with periods of rest – a complete stimulation-rest cycle lasts 20 scans [3].

2.4.2 Statistical Analysis of fMRI Data

In this section, a brief overview of different approaches to get activation maps will be given. After the preprocessing steps, statistical analysis is performed to find out which voxels are activated by the stimulation. This may be a basic correlation analysis or advanced modelling of the expected hemodynamic response (HR) to the stimulation. The main output from this procedure is a statistical map, which shows the voxels where there is activation in response to the stimulus applied.

Each voxel's time series are independently analysed usually which is called univariate analysis. As an example, standard general linear model (GLM) analysis is univariate. There are also multivariate methods which process the data as a whole, these methods implement spatial affinity within the data [27].

There is also an alteration between model-based and model-free methods. In a model-based method, a model of the expected output is generated and compared with the data obtained [28]. In a model-free method, effects of interest in the data are acquired by use of some specific criteria [29, 30]. This method leads to an unexpected outcome in the data, and difficulties to set up a good model for the analysis of data. There are also methods which are in between model-based and model-free. As an

example Clare and colleagues proposed a method, where the barely model data given is the starting time of each stimulation period. A statistic map is obtained by associating the variance within periods, with the variance across periods [31].

GLM proposes a model which is expected as an output and fits it to the data. When the model is acquired from the timing of the stimulation, then a good fit between the model and the data implies that the data was caused by the stimulation. Since GLM is a univariate analysis, a single voxel is considered only, and the fitting of models to this voxel's time-course. When the evaluated data is assumed to be a single one dimensional (1D) vector of intensity values, then a basic example of linear model should be like in Eq. 2.1:

$$y(t) = \beta \times x(t) + c + e(t) \qquad\qquad (2.1)$$

Here $y(t)$ is the data, a 1D vector of intensity values, one for each time point. $x(t)$ is the model, and it is as well a 1D vector with one value for each time point. In considering a square-wave block design, $x(t)$ might be a series of 1's and 0's, for example, {1 0 1 0 0 1 1 0 1 0 1 0 0 1 0 0}. β is the parameter estimate for $x(t)$, in other words the value that the square wave of height one (1) has to be multiplied, to fit the square wave component in the data. c is a constant, and for this case, corresponds to the baseline intensity value in the data. e is the error in the fitting of the model. Thus the fitting of the model requires adapting the baseline level and the height of the square wave, to best fit the data; the error term constitutes the remaining error between the model that is fit and the data [3].

$$y = \beta_1 \times x_1 + \beta_2 \times x_2 + c + e \qquad\qquad (2.2)$$

If case of a two different stimulus is applied, the model should be as seen in Eq. 2.2. There are currently two different model waveforms that are in accordance with the two stimulus time-courses. There are also two estimation parameters, β_1 and β_2. Then if a certain voxel is in correspondence with model x_1 the fitting of the model will give a high value for β_1; else if the data is more similar to the second model time-course, x_2, then fitting of the model will give β_2 a high value. In a complex model different
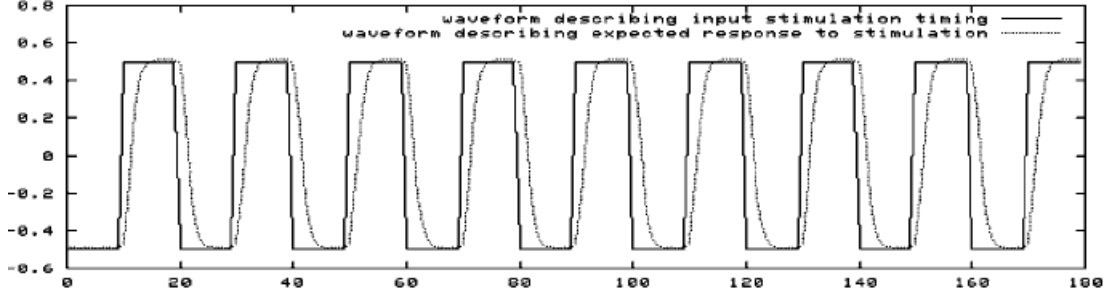
Figure 2.8 Model waveform formation [3].

model waveforms are named as explanatory variables (EVs), as they explain different processes in the data.

For getting the best fit of the model to the data, the stimulus function is convolved with the hemodynamic response function (HRF). This process acts as the effect similar to the brain's neurophysiology had on the the stimulation. The brain's HRF is a delayed and distorted version of the applied time-series, so a mathematical function is applied to the stimulus to take the square wave input and form a delayed and distorted version, which fit the data best. An example of this can be seen in Figure 2.8, depicting the raw stimulation timing waveform and the HRF-convolved model – x(t) – which is implemented in model fitting. The square waveform describes the applied stimulation timing in Figure 2.8, the smoothed waveform results from convolving the stimulation with the HRF, a transformation which presents the model more similar to the actual measured data.

The GLM is often formulated in matrix representation. So all of the parameters are represented together in a vector $\vec{\beta}$, and all of the model time-courses are shown in a matrix $\vec{X}$, namely the design matrix. In Figure 2.9, an example design matrix with two model time-courses is depicted. Each column is a different component of the model. As an example in an experiment, it can be that there is both visual and auditory stimulations as an input, but with varying timings. Then the left column (x_1 or EV1) models the visual stimulation, and the right column (EV2 or x_2) models the auditory one. Time is on the y axis, in down direction and a column has two representations of the model's value at each timepoint. The base intensity encodes the model's value at a particular time point, and also the line graph does [3].
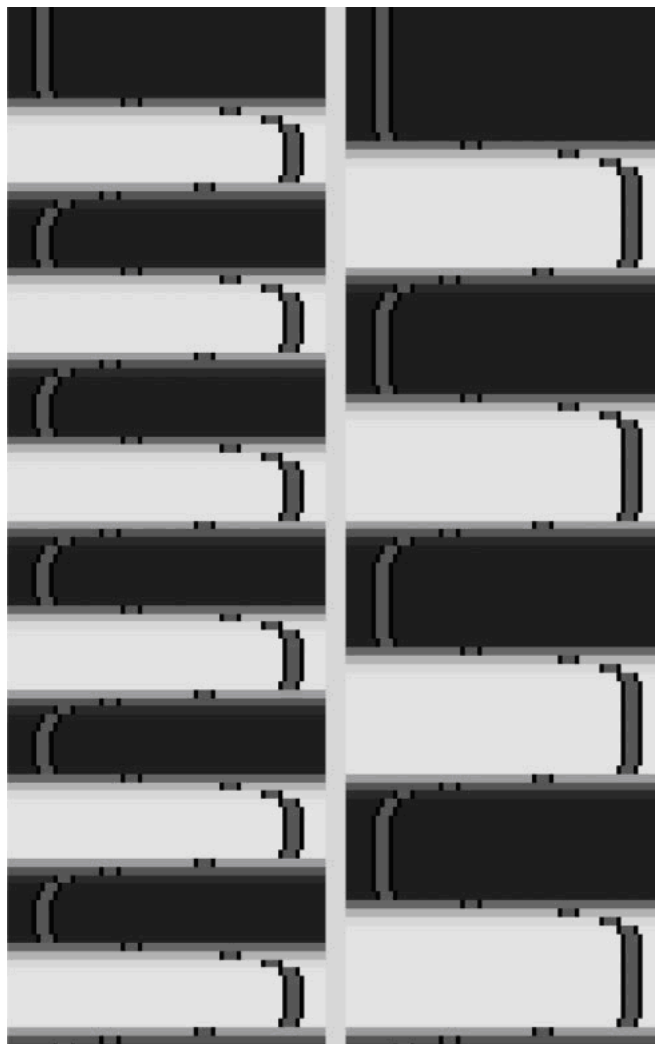
Figure 2.9 Example design matrix with two explanatory variables [3].

When the model is fit distinctly to the data at each voxel, an estimate of the accuracy of fitting will be obtained. This corresponds each column in the model to the voxel's time-course. For the case of a visual/auditory experiment, the first column will give a high first parameter estimate in the visual cortex where the second will give a low second parameter estimate, since this part of the model will not fit well to the voxel's time-course.

To convert a parameter estimate (PE), estimated β value, into an appropriate statistic, its value is associated with the uncertainty in its estimation. This results in a T value where T is PE divided by standard error. If the PE is low relative to its estimated uncertainty, the fit is not significant. So T is a well measure of if the estimate of the PE value is significantly different from zero. To convert a T value into a probability (p) or Z statistic requires standard statistical transformations. Even if they are named differently, in fact T, p and Z all contain the same information. They all reflect how significantly the data is related to a particular part of the model [3].

In addition of producing images of Z values that show how well each voxel fits to each EV, parameter estimates can be associated to analyze directly whether one EV is more applicable to the data than another. In order to find this, one PE is subtracted from the other, the standard error for this subtraction value is calculated, and a new T image is formed. These procesures are controlled by adjusting contrasts.

In order to compare two EVs via substraction, EV1's contrast value is set to -1 and EV 2's to 1. This can be represented as a contrast of [-1 1], since the contrasted parameter estimate is -1 x β_1 + 1 x β_2. A T statistic image will then be acquired, indicating in which region the activation of stimulus 2 is significantly higher than the activation of stimulus 1. It can be that the response to two different stimuli applied at the same time is greater than expected by adding up the responses to seperate stimulus. In this case, these kind of non-linear interactions need to be taken into account in the model.

It is important that all of the EVs are independent of each other. If any EV is

near to be a sum or weighted sum of other EVs in the design, then the fitting of the model to the data does not work well bacause the design matrix is not of full rank [3].

2.4.3   Tresholding the Statistical Map

The next step in fMRI analysis is to threshold the statistic map (T or Z) obtained. This is required to decide which regions of the brain were activated significantly. There are a different procedures for applying thresholding. The most basic method of thresholding is to select a significance (p) threshold and apply this to every voxel in the statistic map. The shortcoming of this method is that many tests need to be performed, since there is a high number of voxels in the brain. If 40 000 voxels are tested at a significance of p<0.01, then it is assumed that 400 will activate sporadically, even if no stimulation was applied. It is not a good way to accept these as being activated as an assumption. This multiple-comparison problem brings the necessity not to accept all activations reported by this method of thresholding. A correction is needed to decrease the number of false positives. Usually a Bonferroni correction is implemented, where the significance level at each voxel is divided by the number of voxels in order to adjust the number of comparisons done. But this gives an outcome of very strict thresholding. For the case given above, the resulting p threshold will be 0.01/40 000 = 0.00000025.

A better way of voxel by voxel thresholding is to implement Gaussian random field (GRF) theory [32]. The basic distinction is that this method considers the spatial smoothness of the statistic map by estimating the count of statistically independent voxels, which should be lower than the original number. This method is less strict compared to simple voxel by voxel thresholding by Bonferroni correction. Usually the correction to p-values is reduced by a factor intthe range of 2 to 20.

Implementing GRF theory, it is convinient to consider spatial extent of regions of activations, before estimating significance. Hence instead of assigning a p-value to each voxel, clusters of voxels are formed according to an initial thresholding, and then each cluster is assigned a p-value. Usually this method is more sensitive to activation

than the voxel based methods [3].

2.4.4   Statistics with Multi-subjects

An experiment can be performed several times on the same subject or with several different subjects. In this case the sensitivity of the overall experiment is increased. This is necessary to make any conclusions of the application of outcomes to a wider range of subjects. The required procedure is to allocate the brain images from all sessions into a common space in order to put together statistics from different sessions or subjects which is also known as registration. This can be achieved by implementing common registration tools and can be performed either on the raw data or on the statistic maps obtained with the first-level analyses.

After all the data are allocated into a common space, there are different kinds of statistical methods to affiliate the outcomes between sessions or subjects. A single result for a group of subjects can be formed. Also different groups of subjects can be compared. These methods assemble fixed-effects and mixed-effects analyses. Fixed-effects accepts that all subjects activate equally, and takes into account only the within-session errors. Mixed-effects analysis also considers between-session errors, and therefore makes less assumptions about the data; hence its results are valid for the whole population from which the group of subjects is drawn. However, the mixed-effects analysis tends to give more conservative results [33].

It is an issue of debate how many subjects are needed to do multisubject statistics to be sensitive and reliable. This is related with many factors assembling the degree of response to the stimulation, between-subject variance and MRI scanner features. It is widely accepted that groups having less than 10 subjects are suboptimal [34, 35, 36].

2.4.5   Registration on Standard Brain Atlases

As mentioned previously, registration is used widely when combining fMRI data from different subjects or sessions. It can also be implemented to allocate the low-resolution fMRI images to a high-resolution structural image, so that the activations can be viewed on a high-resolution brain image in order to make the interpretation of the activations easier. To achieve this templates and brain atlases are used. These consist of data which are transformed into a "standard brain space", for example the coordinate system specified by Talairach and Tournoux [37]. A template is typically an average of many brains, all registered into any given common coordinate system. An example is the Montreal Neurological Institute (MNI) 305 average template [38]. An atlas is also based in a common coordinate system, but contains more sophisticated information about the brain at each voxel, for example, information about tissue type, local brain structure or functional area. Atlases can inform interpretation of fMRI experiments in a variety of ways, helping the experimenter gain the maximum value from the data. Harvard-Oxford Cortical and Sub-Cortical atlases used in this study are an example for this [39].

Registering data on an atlas usually means modify an respective brain to adapt the shape of the target atlas. After this adjusting experimental data, like a functional MRI (fMRI) activation pattern, passively along with the modification [40]. There are many registration methods developed [41, 42, 43], which may be grouped as volume-based and surface-based approaches [44]. Since surface-based registration considers the topology of the cortex, it is likely to have implicitly greater comformance, particularly when it spherical maps are utilized to outmatch the simulated cuts necessary when preparing flat maps [45, 46].

2.4.6   Lateralization in fMRI Compared with Wada Test

Lateralization index (LI) is calculated in order to determine the relative localization of cortical functions across right and left hemisphere. Localization of cortical functions in presurgical planning for brain surgery is important in in three aspects: 1)

the general level of risk needs to be predicted, 2) to help the surgeon determine the boundaries of the excision, 3) to help determine the location of abnormal brain areas preoperatively [47]. Generally used localization technique is the intracarotid amobarbital, or Wada test, which measures the relative lateralization of language and memory functions across the two hemispheres [48, 49]. Preoperative determination of language lateralization is essential in selecting patients for more invasive and specific localization procedures, like intraoperative stimulation mapping [50]. Determination of language lateralization is particularly useful in the preoperative evaluation of epilepsy patients, because this population tends to have a higher percentage of atypical language dominance than the normal population [51, 52]. Despite there are various methods for determining language dominance, the Wada test remains the only method used widely for this aim [53, 54].

The Wada test has a proven measure of language lateralization, but it has several drawbacks as well. Firstly the necessary procedure is invasive, with reported complication percentage of up to 3% [55]. Secondly the test determines only the relative distribution of language dominance across the two hemispheres. A detailed information about localization within a hemisphere, which is needed for preplanning an excision needs to be get by other methods. Lastly validity of the test relies on demonstration of relatively distinct and symmetric arterial supply paths for the two hemispheres.

fMRI can be a noninvasive alternative to the Wada test. fMRI gives detailed brain images visualizing the location of MR signal changes related with cerebral activity. Previous reports described applications of fMRI in mapping motor [56, 57, 23], visual [58, 59, 60], auditory [61, 62] and somatosensory systems [63]. It is necessary to know if these lateralized regions observed in normal subjects actually correspond to systems of the language dominance, as determined with the Wada test. A good way to investigate this is to compare Wada test results with that of the fMRI lateralization patterns on the same subjects.

Binder et al. found in their study The Wada and fMRI tests were in accordance in all 22 patients studied in determining the hemisphere with greater language dominance.

Eighteen patients (16 right-handed, 1 ambidextrous, and 1 left-handed) had strong left hemisphere dominance for language as found by the Wada test, with LI values ranging from +56 to +100. Also these 18 patients showed precise left hemisphere dominance by fMRI, with LI's ranging from +35 to +81. The four remaining patients ,three right-handed and one left-handed, had atypical language distributions as judged by both Wada and fMRI. Wada LI's in these patients ranged from -60 to +25, and corresponding fMRI LI's ranged from -52 to +12 [47].

2.4.7   Lateralization Calculation in fMRI

The LI is calculated by $(nR - nL)/(nR + nL)$, where nL and nR are the number of activated voxels in the left and in the right hemisphere, respectively. Subjects with a negative LI are considered to be left lateralized for language, while those with a positive LI are considered right lateralized. This choice of criteria for lateralization is made on the basis of lateralization results obtained from both invasive investigations (ICA and electro-cortical stimulation) and a non-invasive method which involves the direct statistical comparison of left and right hemisphere fMRI activation, namely statistical lateralization maps [64].

As an explanation to value normalization in LI calculation, Binder et al. and Nagata et al. used the opposite convention of LI $(nL - nR)/(nL + nR)$ multiplied with 100 compared to this study [47, 65]. Liegeois et al. and Staudt et al. used the same convention implemented in this thesis study [64, 66].

2.5   DICOM and the Communication Standards in Medicine

With widely installations of computed tomography (CT) worldwide, also followed by other digital diagnostic imaging modalities in the 1970's, and the increased use of computers in clinical applications, the American College of Radiology (ACR) and the National Electrical Manufacturers Association (NEMA) realized the urgent need for a standard method for transferring images and related information between devices

produced by different vendors. These devices were initially producing different kinds of digital image formats.

The American College of Radiology (ACR) and the National Electrical Manufacturers Association (NEMA) established a joint committee in 1983 to develop a standard to enable communication of digital image information, independent of device manufacturer. This committee also aimed to improve the development and wide use of Picture Archiving and Communication Systems (PACS) that can also interface with other medical information systems and to lead to the building of diagnostic information databases that can be reached by a large number of devices distributed in different locations.

ACR NEMA Standards Publication No. 300 1985, published in 1985 was released version 1.0. The Standard was followed by two revisions: No. 1, dated October 1986 and No. 2, dated January 1988. ACR NEMA Standards Publication No. 300 1988, published in 1988 was designated version 2.0. It included version 1.0, the published revisions, and additional revisions. It also included new material to provide command support for display devices, to introduce a new hierarchy scheme to identify an image, and to add data elements for increased specificity when describing an image. These Standards Publications specified a hardware interface, a minimum set of software commands, and a consistent set of data formats [67].

The Standard, which is currently known as DICOM, includes a number of major developments to previous versions of the ACR-NEMA Standard. DICOM is applicable to a networked environment. DICOM supports working in a networked environment using the industry standard networking protocol TCP/IP. DICOM is also applicable to an off-line media environment using industry standard media such as CD-R and Magnetooptic Disc (MOD) and logical filesystems such as ISO 9660 and PC File System (FAT16). DICOM describes the way that devices claim for conformance to the Standard react to commands and data being exchanged. DICOM specifies, through the concept of Service Classes, the semantics of commands and relevant data with levels of conformance. DICOM explicitly defines in which way implementor must compose

a Conformance Statement to select specific options. DICOM is structured as a multi part document. This facilitates evolution of the Standard in a rapidly changing environment by simplifying the addition of new features. International Organization for Standardization (ISO) directives to define structured multipart documents have been utilized in the building of the DICOM Standard. DICOM supports explicit Information Object Definitions (IOD) not only for images and graphics but also for waveforms, reports, printing and more. DICOM describes an assembling technique for uniquely identifying any IOD. This makes exact relationships between IODs possible as they are interchanged through the network [4].

The DICOM Standard is changed continuously and it is conformed with the procedures of the DICOM Standards Committee. Proposals for developments are suggested by the DICOM Committee member organizations evaluating the feedback from the implementors of the Standard. These suggestions are evaluated for implementing in future versions of the Standard. A prerequisite in updating the Standard is to supply effective compatibility with previous versions. This Standard has been developed with an emphasis on diagnostic medical imaging as practiced in radiology, cardiology and related disciplines; however, it is also applicable to a wide ran of image and non-image related information exchanged in clinical and other medical environments [68].

Figure 2.6 presents the general communication model of the standard which spans both network (on-line) and media storage interchange (off-line) communication. Applications may relay on either on of the following boundaries, the Upper Layer Service, which provides independence from specific physical networking communication support and protocols such as Transmission Control Protocol/Internet Protocol (TCP/IP) and the Basic DICOM File Service, which provides access to Storage Media independently from specific media storage formats and file structures.

2.6   DICOM Structured Report Format

In the medical imaging usually images are the input, reports are the output. The importance of diagnostic procedures is coming from the reporting physician's expertise
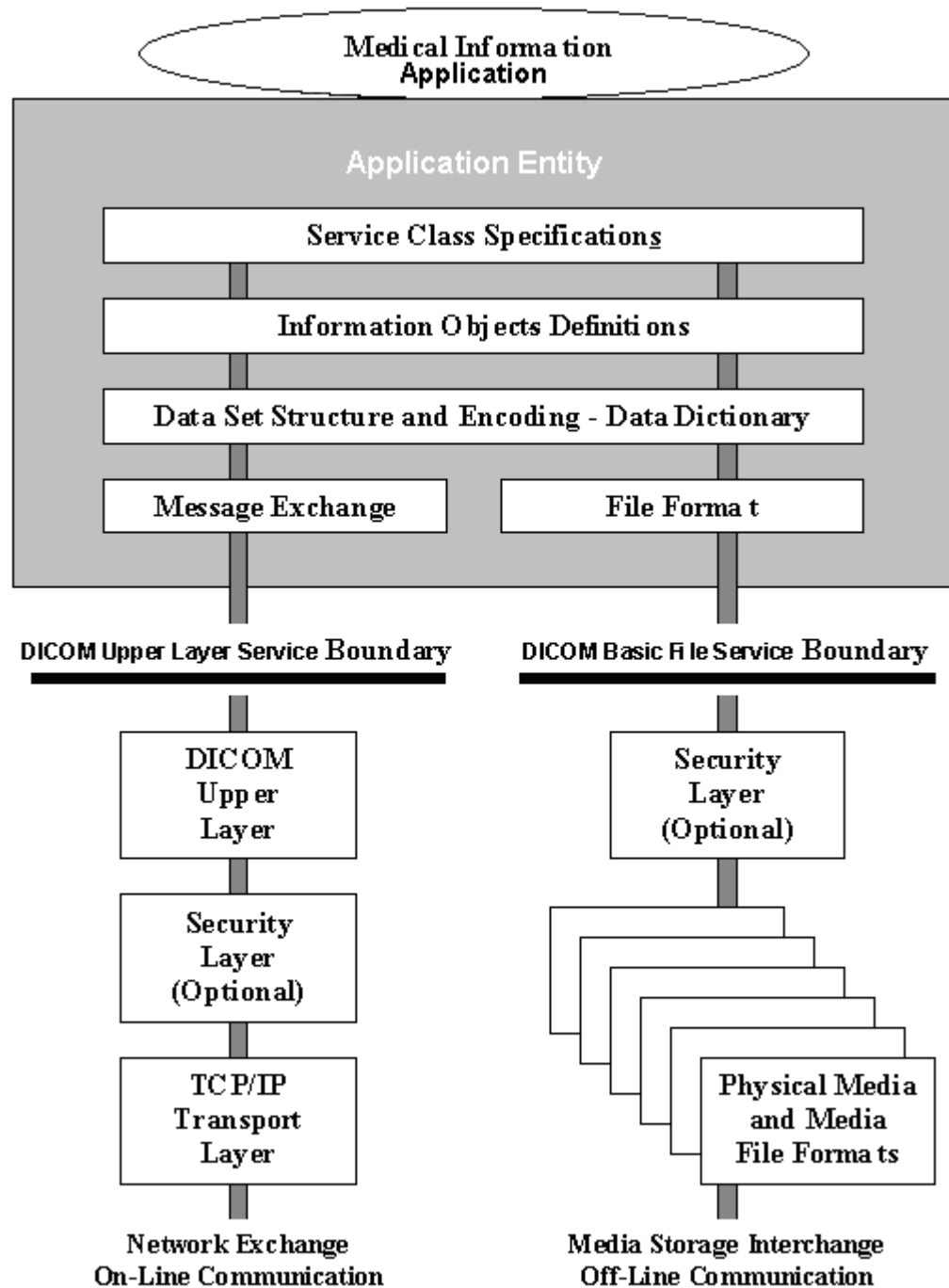
Figure 2.10 DICOM General communication model [4].

rather than the source images. Traditional reports typed on paper are explanatory in themselves, but they should ideally be combined with the corresponding images. At the past when hardcopy films were merely used, the films were annotated with pencil to draw regions of importance to the reader of the report. As softcopy images and PACS are widely used and reports migrate online, the link between the report and region of interest in certain images are currently much more required. DICOM structured report format is formed primarily to cover this requirement.

DICOM SR includes the presence of lists and hierarchical relationships, the use of coded or numeric content in addition to plain text and the use of relationships between concepts. DICOM SR can have any kind of structured content, not just text reports. SR documents can be implemented wherever there is a need for lists or structured content, or a need for coded concepts or numeric values, or a need for references to images, waveforms or other composite objects. In a DICOM SR document all data is carried by individual content items. Each content item is a name-value pair. The value may be of various types, including plain text, coded values, numeric values (with units), person names, dates and times, references to DICOM images, waveforms and other composite objects and spatial and temporal coordinates in referenced objects [69].

References to images and waveforms are restricted to DICOM objects only. An instance of a DICOM CT Image Storage SOP class can be referenced, but not any outer GIF or TIFF file. Likewise, a DICOM Basic Voice Audio Waveform Storage SOP class instance can be referenced, but not an arbitrary MP3 or WAV file. Preformatted text, like HTML, Word, Postscript or PDF cannot be included. Only unformatted text may be included, however including embedded new lines are permitted [69].

The contemporary PACS and information systems attempt to unite reports and images. Such a report is composed of report and images. There exist rare systems that support for the links between findings in the report and specific features within images. DICOM SR fills this gap by providing such links. The value of supplying any kind of access to both the report and the relevant images simultaneously should not be

underestimated. Many software environments, also commercial products of big medical companies maintain reports and images completely separately. So the procedures are selected manually on two different terminals. Usually, not even old reports are available on the imaging workstations.

DICOM SR provides features to reference individual images, specify a presentation state to be applied to the referenced images, reference other DICOM objects such as time-based waveforms (Electrocardiogram, etc.), specify spatial coordinates within an image, specify temporal coordinates within an image or waveform, specify the purpose of the reference to the coordinate, image or waveform [69].

There are various tools for DICOM SR editing and functional MRI data analysis. DICOMscope by OFFIS DCMTK is one of the best well known for the SR Editing. It is a free DICOM viewer which can visualize uncompressed, monochrome DICOM images from all modalities and which supports display calibration according to DICOM part 14 as well as presentation states. DICOMscope also makes a print client available. The development of this prototype was commissioned by the "Committee for the Advancement of DICOM" and demonstrated at the European Congress of Radiology (ECR) 1999. An enhanced version was developed for the "DICOM Display Consistency Demonstration" at Radiological Society of North America (RSNA) InfoRAD 1999. The current release 3.5.1 has been demonstrated at ECR 2001 and contains numerous extensions, including a print server, support for encrypted DICOM communication, digital signatures and structured reporting [70].

2.7 Tree Structure in DICOM SR

Data in a DICOM SR is kept in a tree of content items. This tree is encoded in a DICOM data set as recursively-nested different length sequence items. This kind of DICOM representation is intended for change and storage purposes. This is not normally an appropriate representation, utilizing within an application in which the tree needs to be transformed or manipulated.

As to encode an SR tree, a node structure with child and siblings needs to be defined, and a root node as a parent. Such a basic representation is enough to both compose a tree in a top-down orientation, and to parse it the same way.

Given the root node, processing the tree becomes a simple issue of pursuing the forward linked lists of siblings and children, waiting when a null pointer is got. If it is needed to go back to previous siblings or ancestors appropriate links can be provided in each node. This may be implemented to reach the content items in an SR format [69].

2.8   Current State of Implementation

fMRI of the Brain (FMRIB) Software Library (FSL) and Analysis of Functional NeuroImages (AFNI) are the most popular tools for fMRI studies. FSL is a comprehensive library of analysis tools for FMRI, MRI and DTI brain imaging data. FSL is written mainly by members of the Analysis Group, FMRIB, Oxford, UK. FSL runs on Apple and PCs (Linux and Windows), and is very easy to install [71]. Most of the tools can be run both from the command line and as graphical user interfaces [72, 38].

AFNI is a set of C programs for processing, analyzing, and displaying functional MRI data. It runs on Unix + X11 + Motif systems, including SGI, Solaris, Linux, and Mac OS X. It is available free (in C source code format, and some precompiled binaries) for research purposes [73].

Although there are very useful tools and options like FSL and AFNI, these Neuro Imaging tools lack a standard reporting tool where radiologist can either view the images or enter text input for the relevant images. This study is intended to supply this functionality that is not present at the moment.

DICOM SR format is, since it is part of the DICOM standard, well defined and documented, can work across platforms supporting DICOM and has the capabilities both to refer the images and to contain the report. So the format is found to be suitable

for this study to be used for further studies platform independently and also enable to attach to the patient image data folders.

DICOM SR is mainly used in modalities like ultrasound where obstetrics findings are classified and put into report in a standard way. In Neuro MR Imaging, also findings of the radiologist in fMRI studies can be standardized and put into report. This study aims to enable this process to be more automated and structured.

Besides providing a DICOM SR editing tool and references to images, this study also aims to supply fMRI data visualization and analysis tools for lateralization and Z score calculations. Wada or intracarotid amobarbital procedure is invasive but still routine for determining the lateralization of language prior to surgery. fMRI as a non-invasive technique is correlated with the intracarotid amobarbital procedure as the gold standard and in agreement of about 90%. Significant correlation between presurgical fMRI testing and postsurgical outcome for fMRI activations in frontal language areas [74]. Maximum Z scores in specific anatomical regions and mean Z scores at the anatomical regions are used in motor fMRI studies [75].

In this study Python programming language is used as the development platform. The operating system that program runs on is Ubuntu Linux 804 release. Python is a powerful dynamic programming language that is used in a wide range of applications. Python syntax is quite similar to Tcl, Perl, Ruby, Scheme or Java. Some of its key differentiating features from the other programming languages are clear and readable syntax, highly object oriented and natural expression of procedural code. Python is fully modular and iy supports hierarchical packages, exception based error handling, high level dynamic data types, extensive standard libraries and third party modules. Extensions and modules are easily written in C, C++, Java for Jython, or .NET languages for IronPython and embeddable within applications as a scripting interface. Python enables writing the necessary code quickly and also with its highly optimized byte compiler and support libraries, Python code runs fast enough for most applications [6].

The GUI design is made by utilizing Glade 3.4.5 release as a user interface designer for GTK+. Glade enables fast and easy development of user interfaces for the GTK+ toolkit and the GNOME desktop environment. The user interfaces designed in Glade are saved as XML, and by using the GtkBuilder GTK+ object, these can be used by applications dynamically as required. Via GtkBuilder, Glade XML files can be used in many programming languages including C, C++, C#, Vala, Java, Perl and Python [7].

The software developed in this study with Python as the programming language and Glade Toolkit as GUI has a DICOM SR Viewer and Editor. The DICOM SR tag values can be viewed and the report context can be edited. fMRI and anatomical images are overlayed and functional data is registered on Harvard-Oxford Cortical and Subcortical anatomical atlasses to view anatomical regions. Lateralization maximum Z value and location analysis can be done. Mean Z values throug the anatomical regions are analyzed and presented as a table in the software. Finally after analyses report can be prepared and saved both in DICOM SR format and in rich text format.

The software is designed to be open source and will be shared under General Public License (GPL) for research purposes. The software will be firstly used in Kozyatagi Acibadem Hospital by Radiologist Alp Dincer,MD. with images acquired from Siemens Trio 3 Tesla MR. This site is working on many patients for presurgical planning, brain lateralization experiments and fMRI studies.
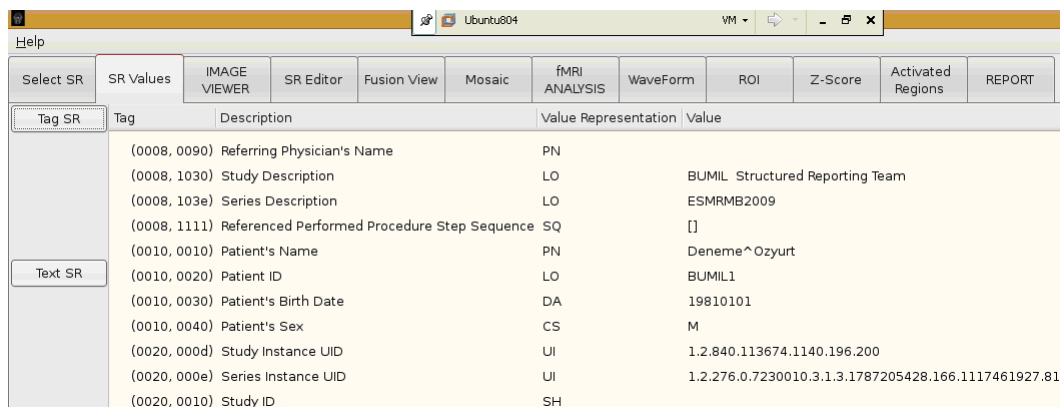
3. DESIGN OF THE SR GUI

3.1 DICOM Tag Analysis and Selection

Each DICOM tag has a basic definition consisting of Tag, Description, Value Representation and value as described in Appendix A.1. This can be seen in the Figure 3.1. Instead of using an external DICOM Tag viewer, a built in function in the program is developed in order to view these values that is used afterby in decoding DICOM and programming the software to read and write these values. It is described how all the tags and values can be taken into a treeview in Appendix A.2.

3.2 Edit SR Text Values

As it has been discussed in the previous chapter, the DICOM SR Text is embedded in the "containers" where the text value is stored. It is described how to view and store text field container in Appendix A.3.

The buffer is taken from the text edit area, then the text in the buffer is set to a text variable which is set to the report section content. The screenshot in Figure 3.2 shows where the report text content is edited and set. Then the report is viewed as seen in Figure 3.3.

Figure 3.1 DICOM SR tags and their values.

Figure 3.2 SR text report editor module.

Figure 3.3 SR view report function.

Table 3.1
Image reference container in DICOM SR.

(0040, a010) Relationship Type CS: 'CONTAINS'
(0040, a040) Value Type CS: 'IMAGE'
(0040, a043) Concept Name Code Sequence 1 item(s) —-
(0008, 0100) Code Value SH: 'IR.02'
(0008, 0102) Coding Scheme Designator SH: '99_OFFIS_DCMTK'
(0008, 0104) Code Meaning LO: 'Best illustration of finding' (0008, 010c)
Coding Scheme UID UI: 1.2.276.0.7230010.3.0.0.1
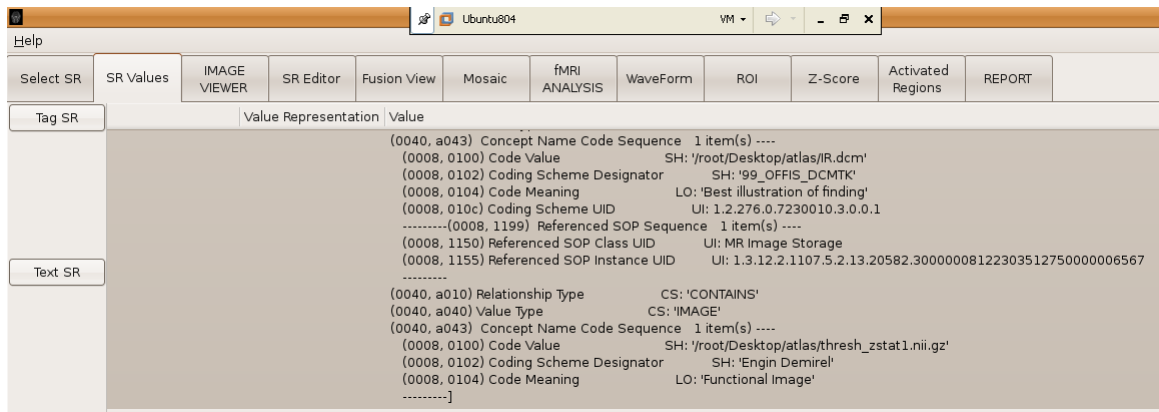
Figure 3.4 Image references in a DICOM SR.

3.3   Image References

Image references are just like the text areas in the DICOM SR documents, are present in the container part. The difference from text storage is that the image reference contains only the reference to the image with the unique identifier of the coding scheme of image.

In the software developed for this thesis work the DICOM image of which the patient data is taken and the functional image path is referenced in the DICOM SR file in which the report is saved after all the analysis process is finished. These references are seen in Figure 3.4.

The referenced DICOM image is conveyed with the SR file where by means of

a DICOM SR viewer the content of the file and the referenced DICOM image can be visualized. The referenced functional image is in Neuroimaging Informatics Technology Initiative (NifTi) format which is widely used and supported in fMRI tools like FSL and AFNI used for scientific researches. Since DICOM SR does not support viewing of image references other than formats DICOM, it is out of the scope of this study to make NifTi image viewable via an SR Viewer by means of data conversions. The path where the functional image is stored andd can be retrieved by neuroimaging tools is stored. The methods are described in Appendix A.4.

4. POSTPROCESSING OF fMRI IMAGES

4.1 View and Fuse Functional and Anatomical Images

A DICOM image file has a header and an image part. In order to view and postprocess the Dicom files in Python, first the image data has to be read and converted to an array. This is described in Appandix B.1.

The overlay of two images and changing the contrast of one image releative to the other one is another challenge in visualizing array data with Python. The matplotlib module has a pylab submodule having useful fuctions as imshow like the matlab analog to view the array data. The alpha property of imshow function which enables visualization of purely one image to a mixture of overlay with the other to a pure second image. This property of alpha blending of the images are used here. One important class "IndexTracker" is needed to be adapted for this functionality and has been used in many of the other modules of this software development. This function is explained in Appendix B.2. It can be seen that the alpha value can be changed in the GUI is shown in Figure 4.1.

4.2 View Mosaic Images Slice by Slice

It is not that straight forward to figure out if a DICOM file is a mosaic image or not. It is not written in any tag of the header. To find out if the DICOM file is mosaic, the definition made by Douglas N. Greve, Ph.D., MGH-NMR Center at Harvard University is implemented [76]. The implementation of this method is described in Appendix B.3. The mosaic image can be seen in Figure 4.2.

Once the single arrays are taken from the mosaic images they are viewed by the imshow function, and slices can be changed via mouse scroll by implementation of IndexTracker function in the module to change the slice numbers viewed. The separated view and the ability to change slice numbers is depicted in Figure 4.3.
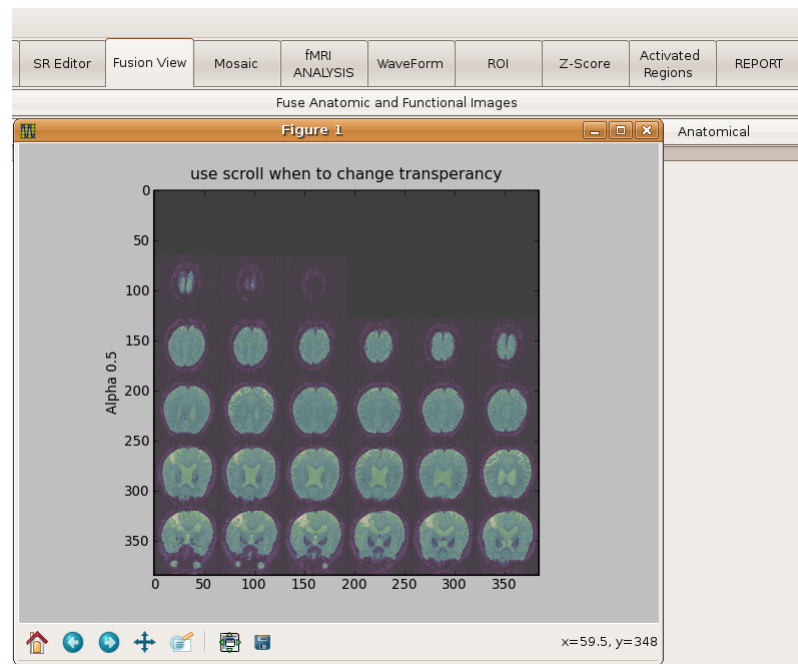
Figure 4.1 Fusion of images and Alpha blending.
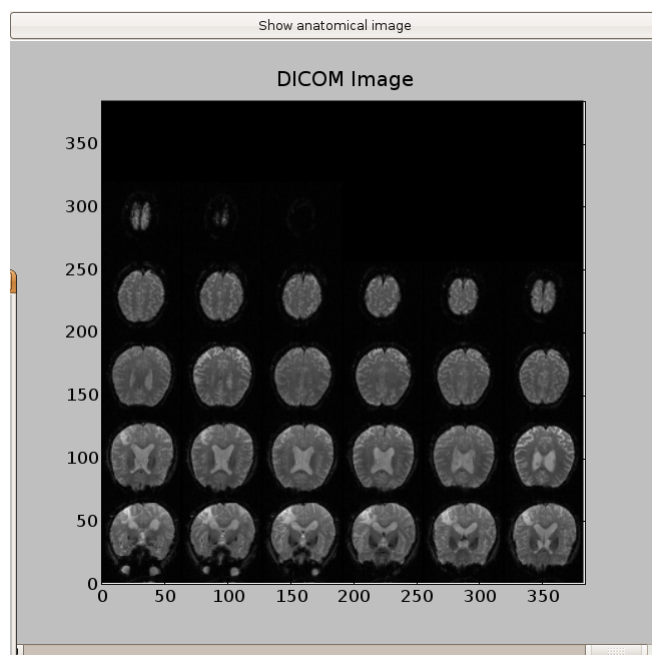
Figure 4.2 A mosaic image with show anatomical image function.
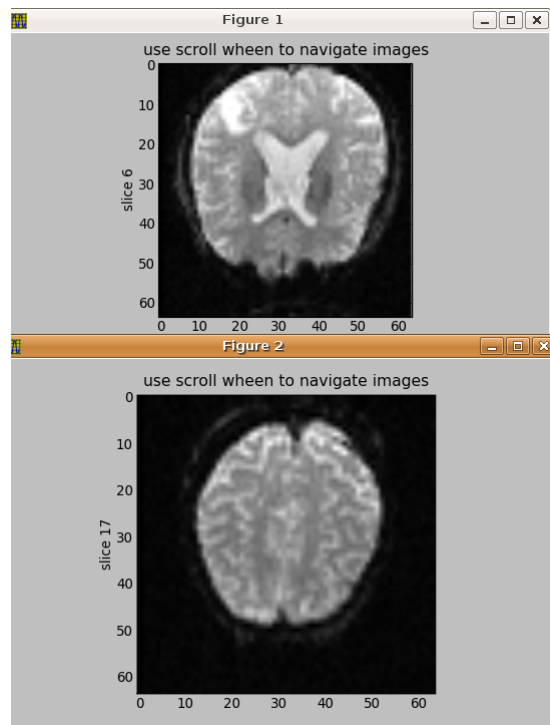
Figure 4.3 Mosaic images are shown separated slice by slice.

4.3 Labeling Using Anatomical Atlas

Within this study the functional MRI data is overlayed on to standard atlases called Harvard-Oxford cortical and subcortical structural atlases. Probabilistic atlases covering 48 cortical and 21 subcortical structural areas, derived from structural data and segmentations are developed by the Harvard Center for Morphometric Analysis. These atlases are also used in FSL program which is widely used and popular in neuro imaging. The atlases are loaded to arrays in order to get the data later to which the location refers to as a label in the atlas. The loading of the standard atlases and registration of functional image data on anatomical atlases are described in Appendix B.5.

4.4 Visualizing Waveform of Images

It is sometimes a very useful data for the clinician or the radiologist to see how a data pixel on the image is changed through a timeseries. To make this possibe, the point selected on the image should be taken in all the time series and plotted. The
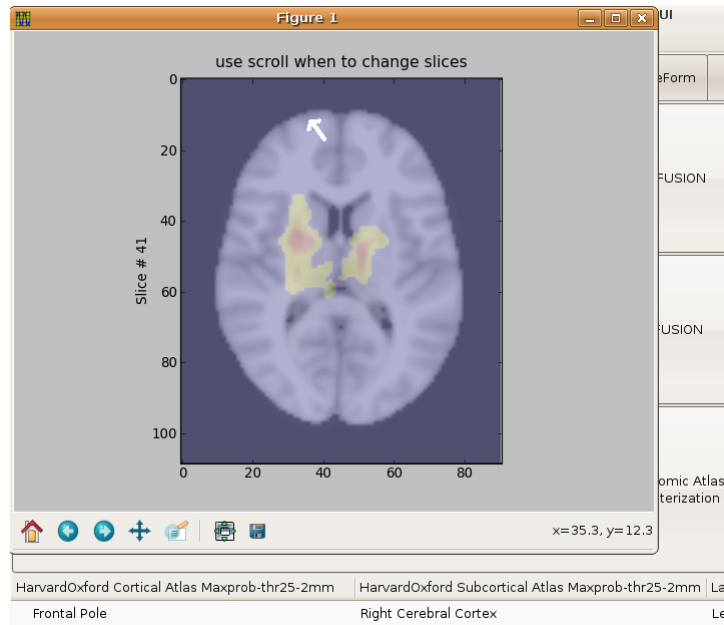
Figure 4.4 Functional data overlayed on anatomical atlas.

details on programming this module can be found in Appendix B.6.

For all timepoints in the NifTi file the pixel value is taken and plotted. User can click and see different locations waveform on the screen. The same buton pressed and released functions like anatomical atlas registration is used here. As the result user can get the data in the Waveform tab of the program as seen in Figure 4.6.

4.5   ROI Visualization and Save

A basic rectangular ROI tool is also developed in scope of the work done. The imshow function from pylab has a ROI tool in itself but the pcolor function just plots the image data on the canvas. In order to get the selected region of the canvas to the screen button pressed and released events are utilized. The x, y points on the screen is taken and normalized according to the data dimensions. As a known feature, the canvas may be different in different computers, so an arrangement may be needed in order to get the correct region with ROI. It is described how this is implemented in Appendix B.7.

Here the method the region is taken via maximum and minimum values of the
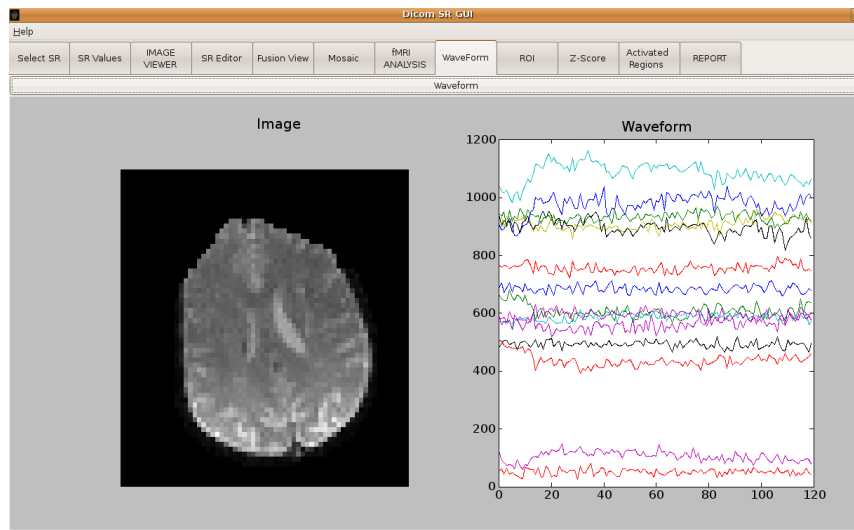
Figure 4.5 Waveform of different locations through timepoints.
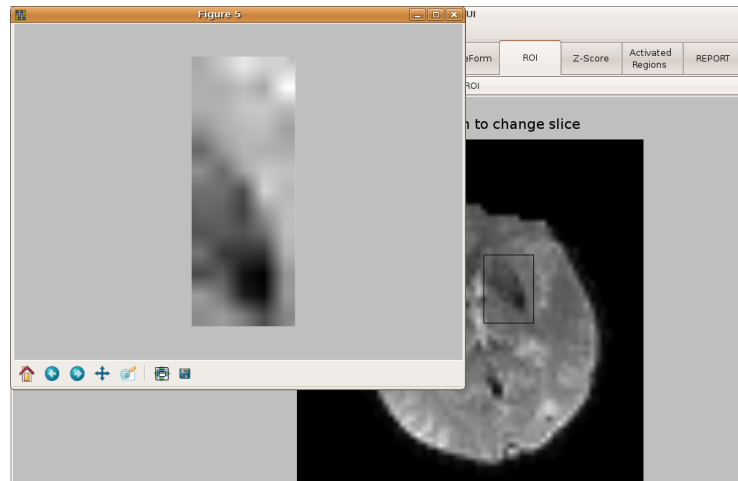
Figure 4.6 Select ROI and view.

coordinates was a challenging part of the development. Again from the coordinate orientation of pyNifTi data function, a rotation was necessary so as to show the canvas correctly [77]. As also seen the coordinates, y and x are assigned in opposite order. You can see in Figure 4.6 an example from the software with rectangular ROI.

5. fMRI ANALYSIS

In this chapter, the analysis tools essential for the reporting is covered namely lateralization index calculation, maximum Z Template preperation and the active anatomical region analysis. After all processes are covered report creation in rich text format which is editable and printable and the DICOM structured report creation is covered in the last part of the chapter.

5.1 Lateralization Index Calculation

The functional data overlayed on the anatomical template has a tresholded data which the active ones are above a predefined z value. This data as discussed in the previous chapter can be visualized on the anatomical template and the anatomical labels can be shown.

LI is used in language based neuroimaging procedures to determine which side of the brain, left or right is more active. This is then used as an input for surgical planning decision. The value is between -1 and 1, negative values mean left sided, positive values right sided brain for linguistic functions, bigger magnitude represents a more unilateral brain function [66]. The programming details on lateralization index calculation can be found in Appendix C.1. LI is calculated as in Eq.5.1 [64].

$$LI = \frac{rightactivevoxelcount - leftactivevoxelcount}{rightactivevoxelcount + leftactivevoxelcount} \tag{5.1}$$

The count and the calculation that the program returns is inserted into the fMRI analysis page as seen in Figure 5.1.

5.2 Maximum Z Template

As lateralization calculation is an indicator for linguistic fMRI studies, maximum Z value and location detection is an important indicator in motor neuro studies in MRI
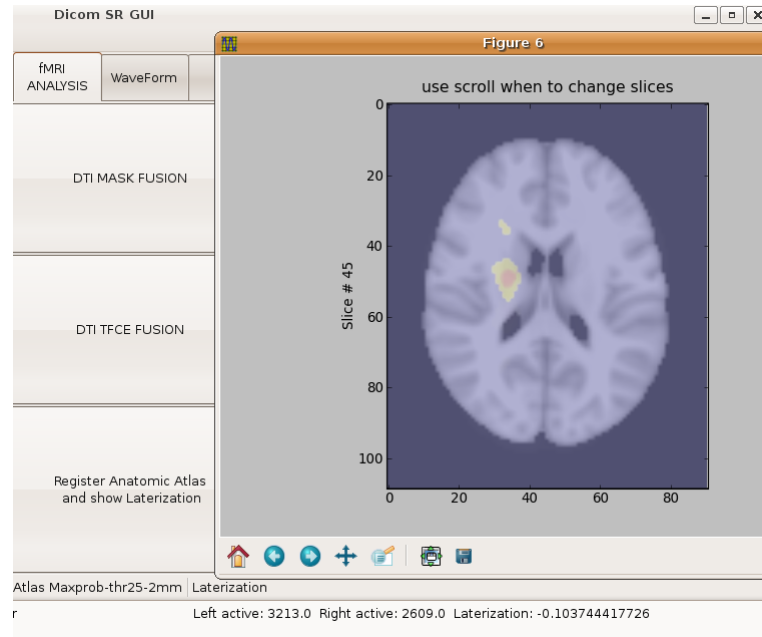
Figure 5.1 Lateralization value and active voxel counts on hemispheres.

[75]. The detection of the maximum z value, anatomical region and plotting the region in the program is essential for a useful neuro analysis tool. The maximum z value and the location is found by employing the procedures described in Appendix C.2. After the process of the functions in this module, the maximum Z point localization in sagital, coronal and axial planes as well as waveform of the functional image in this locale are visualized. Also the anatomical region and the maximum Z value are written on screen with exact axial location. This can be seen in Figure 5.2.

5.3   Active Anatomical Region Analysis

From the forty eight (48) cortical anatomical regions, and twenty one (21) subcortical regions in HarvardOxford anatomical atlas, not all regions are and need to be activated in an fMRI study. To know about which slices as well as which anatomical regions in a study is very useful for the radiologists.

The mean z values in the anatomical regions containing active voxels are analyzed. These are given as a table in the module "Activated Regions" where the regions are seen as cortical and subcortical, two tables. Also number of active voxels in that region
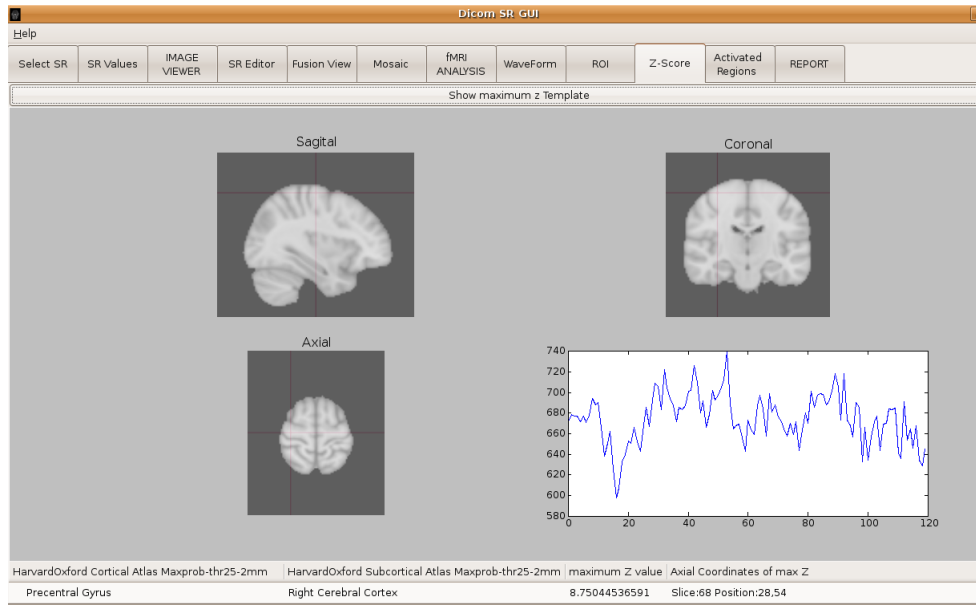
Figure 5.2 Maximum z value localization and value.

and the mean z values of those active voxels is calculated and depicted in these tables. The description of procedures developed to enable this are explained in Appendix C.3.

As these values are calculated and set to the tree view, the tables in Figure 5.3 and Figure 5.4 are formed where data can be sorted according to voxel count or mean Z.

5.4   Report Preparation and Output

In radiology since the images are the input and the report is the output, the last and most important part of this study was to present the results to the user in our case radiologs and the clinicians in an understandable, usable and printable format. This was a challenging task since the format DICOM SR dealt within the project is not widely used in neuroimaging and the NifTi format, apart from being widely used in neuroimaging scientific studies, was not supported by standard commercial DICOM viewers.

The idea behind using DICOM SR was to keep the report with the images, and also having image references in it. This is achieved with the DICOM image references,

Dicom SR GUI

Help

| Select SR | SR Values | IMAGE VIEWER | SR Editor | Fusion View | Mosaic | fMRI ANALYSIS | WaveForm | ROI | Z-Score | Activated Regions | REPORT |

Analyze Activated Regions

Active Slices

[27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78]

HarvardOxford Cortical Atlas Anatomical Region	Active Voxel Count ▲	mean Z Value	HarvardOxford Sub-Cortical Atlas Anatomical Region	Active Voxel Count ▲	mean Z Value
No Label Found	4653	2.800027	Right Cerebral White Matter	1136	2.731584
Superior Frontal Gyrus	670	3.212719	Left Cerebral Cortex	1045	3.134492
Juxtapositional Lobule Cortex	190	2.877241	No label found!	800	2.995853
Precentral Gyrus	106	2.695280	Left Thalamus	546	2.850472
Paracingulate Gyrus	73	2.582689	Right Thalamus	431	2.644159
Middle Frontal Gyrus	61	2.625721	Right Putamen	422	2.783380
Insular Cortex	52	2.503420	Left Cerebral White Matter	398	2.708796
Frontal Orbital Cortex	14	2.464592	Left Putamen	346	2.790047
Subcallosal Cortex	3	2.505604	Right Cerebral Cortex	240	2.508310
			Left Pallidum	217	2.870575
			Right Pallidum	99	2.612719
			Left Caudate	51	2.513923
			Right Accumbens	40	2.659830
			Right Caudate	29	2.439597
			Left Lateral Ventrical	13	2.521450
			Brain Stem	9	2.587498

Figure 5.3 Mean Z value analysis, regions arranged decreasing active voxel count.

Dicom SR GUI

Help

| Select SR | SR Values | IMAGE VIEWER | SR Editor | Fusion View | Mosaic | fMRI ANALYSIS | WaveForm | ROI | Z-Score | Activated Regions | REPORT |

Analyze Activated Regions

Active Slices

[27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78]

HarvardOxford Cortical Atlas Anatomical Region	Active Voxel Count	mean Z Value ▲	HarvardOxford Sub-Cortical Atlas Anatomical Region	Active Voxel Count	mean Z Value
Superior Frontal Gyrus	670	3.212719	Left Cerebral Cortex	1045	3.134492
Juxtapositional Lobule Cortex	190	2.877241	No label found!	800	2.995853
No Label Found	4653	2.800027	Left Pallidum	217	2.870575
Precentral Gyrus	106	2.695280	Left Thalamus	546	2.850472
Middle Frontal Gyrus	61	2.625721	Left Putamen	346	2.790047
Paracingulate Gyrus	73	2.582689	Right Putamen	422	2.783380
Subcallosal Cortex	3	2.505604	Right Cerebral White Matter	1136	2.731584
Insular Cortex	52	2.503420	Left Cerebral White Matter	398	2.708796
Frontal Orbital Cortex	14	2.464592	Right Accumbens	40	2.659830
			Right Thalamus	431	2.644159
			Right Pallidum	99	2.612719
			Brain Stem	9	2.587498
			Left Lateral Ventrical	13	2.521450
			Left Caudate	51	2.513923
			Right Cerebral Cortex	240	2.508310
			Right Caudate	29	2.439597

Figure 5.4 Mean Z value analysis, regions arranged decreasing mean Z value.

as well as the NifTi file references to know which neuroimaging files are processed and studied in preparation of the report. Report in DICOM SR format is kept with all the four text areas namely request, procedure description, findings and the impression and the image references of the DICOM image of best illustration of finding and the path of the NifTi images used. This is saved as a DICOM SR file to the output reports folder that can also be uploaded to DICOM archives. The patient data in the report is taken from the referenced DICOM image header. The developed source code for this module is described in Appendix C.4.

These patient data and the text part is common to the two main different report output developed at the reporting module of the software. This work has two different report output model in terms of RTF output, one is maximum Z template report, the other is laterization report. In maximum Z report output the location and the waveform plot of the maximum Z point as well as its value is included. In laterization report the laterization index, the count of the left and right active voxel numbers are given. Also common to the both report type, the user's selection of images through his/her study saved with imshow functions save ata predefined folder, in .png and .jpg formats are inserted into the report. Since pyrtf supports jpg and png insertion these two formats will be the standard for the report outputs from this study.

Lastly after all input is done to the report, the report is saved under a predefined folder. Hence the reports are saved in a convention not to mix patients as well as different studies of a patient. This can be seen in Figure 5.5. You can see in Figure 5.6, a maximum Z report example created by this software. Also the other format, laterization report, for the same patient with laterization study and the referenced images are given in Figure 5.7.
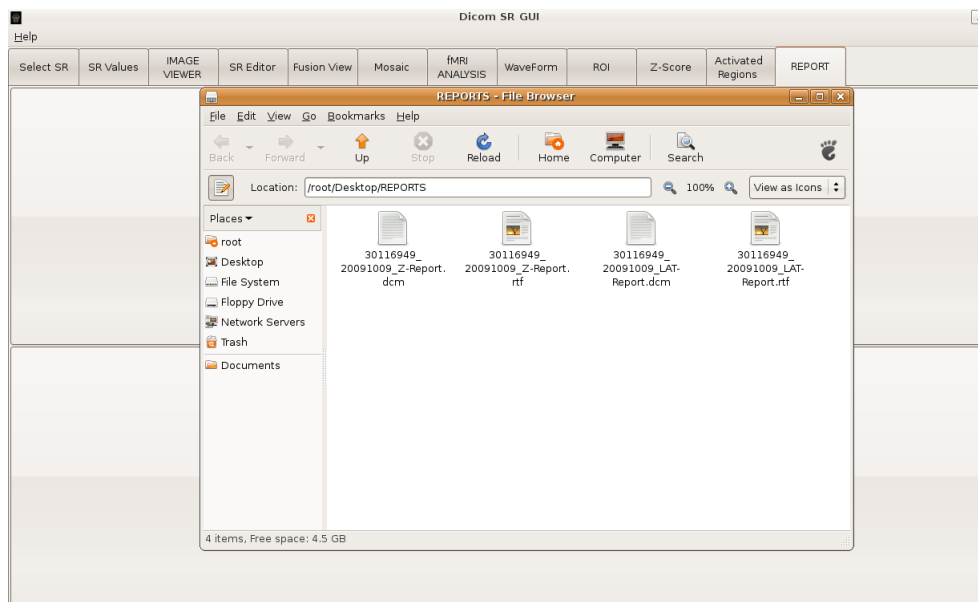
Figure 5.5 Reports folder with the created reports.
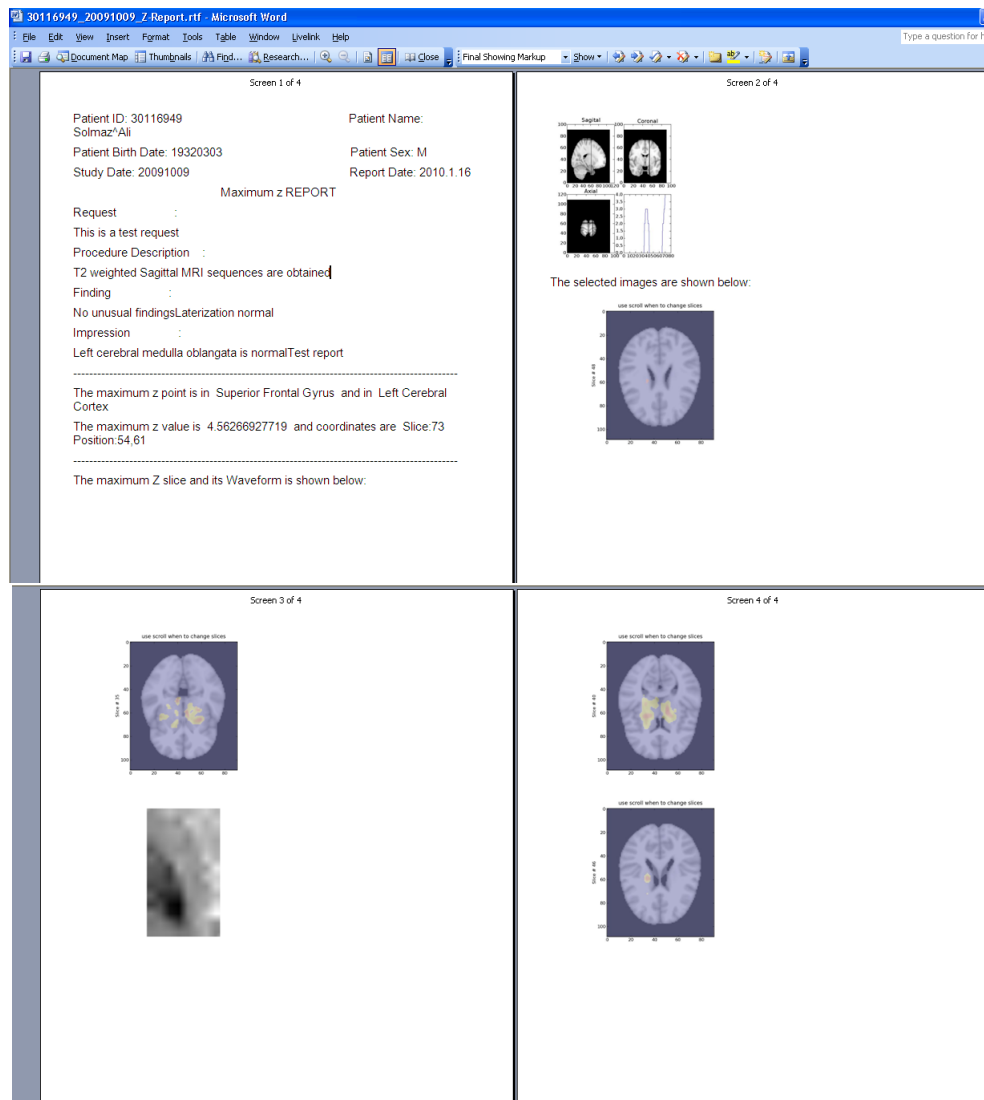
Figure 5.6 Maximum Z report.
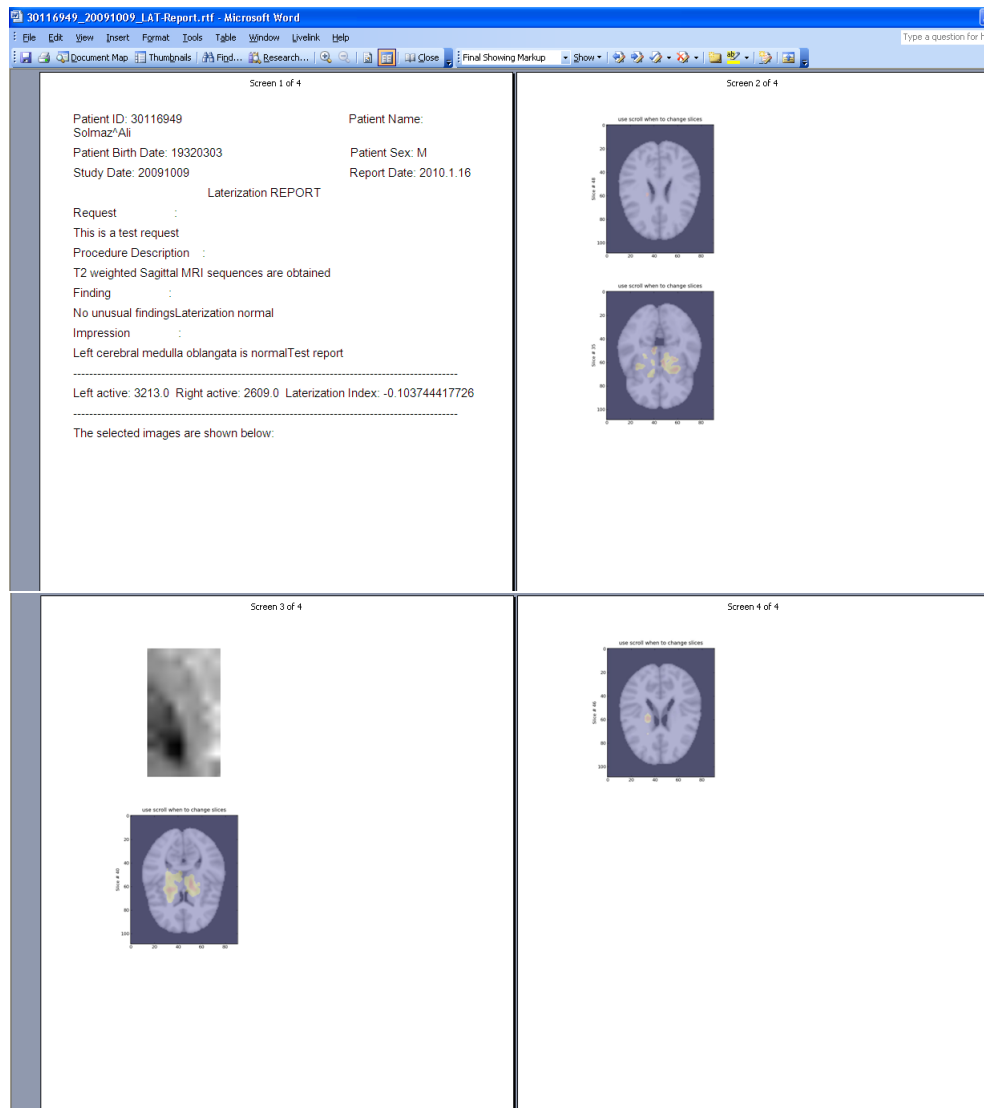
Figure 5.7 Lateralization report.

6. CONCLUSION AND FUTURE WORKS

In this study we aimed to develop useful and reliable reporting and postprocessing tools for neuro MRI witn an initial focus on fMRI studies. The commercial as well as the scientific tools in neuroimaging field was lacking the reporting part and we integrated this capability with the analysis process.

As the first step a GUI is designed to select a DICOM SR template, edit the SR text, view the tag data of the report and reference the images associated with the diagnoses. This part of the thesis can provide a very useful tool in the future for other postprocessing projects standardizing the technical report format.

As the second step, in order to visualize the images evaluated in writing the report and the diagnosis, toolboxes are developed to view the DICOM images, to overlay functional and anatomical images, to analyze NifTi format images and to overlay them with the reference anatomical maps. Slices of DICOM mosaic images are seperated for individual visualization. Region of interest on the anatomical images can be viewed and saved.

The functional images are overlayed on the standard anatomical atlases and the active regions can be viewed. The registration of Harvard-Oxford Cortical and Subcortical probabilistic anatomical atlas is performed, so the radiologist can view the segmentation of the active region mapped to the anatomical one. The quantitative analysis on the functional data including lateralization calculation and maximum z point location and regional analysis of average z values are done.

The DICOM structured report format is preserved for the report to be enabled to upload to the DICOM archive and to enable view it with standard DICOM SR viewers. Within DICOM SR, the rich text format (RTF) report can be saved with references to images for viewing and printing the output. As an output report double format, both DICOM SR and rich text format is preferred because DICOM SR is supporting only references to DICOM images so the NIfTI references are not be possible to be

visualized if they were not included in RTF report.

The software will be used initially at Kozyatagi Acibadem Hospital for fMRI studies, for beta testing and future development. The software will be licensed under GPL and released for open source development and use to the public.

As a future direction, in order to make NIfTI images part of the SR format in a standard way, header of the NIfTI images can be taken, NIfTI image data can be converted to pixel array data and then to DICOM format. Header including patient data can be taken from DICOM images of patient and merged. Further development can be done to support DTI and visualize the tracts of the fibers and the anatomical locations in DTI studies or to serve other neuroimaging postprocessing needs.

Appendix A. DICOM SR GUI DESIGN

A.1 DICOM SR Text Data and Containers

Information in a DICOM structured report is contained in a tree of content items. "(0040, a730)" Tag includes the data for the content sequence. This content sequence has the items of DICOM SR text areas as well as the image references. Below this container come firstly the text containers:

(0040, a010) Relationship Type CS: 'CONTAINS'

(0040, a040) Value Type CS: 'TEXT'

....

Which has as a description (0008, 0104) Code Meaning, has four different text items Request, Procedure Description, Finding and Impression. In this chapter DICOM tag element analysis, setting of text values of DICOM SR files and image references are introduced.

All Text Container items in SR have "(0040, a160) Text Value" fields having the report or finding text inside. Python Dicom module's read_dicom function is used to get the SR contents in an array format which the elements can be reached and modified. If we call ds the dataset we get through dicom_read function, individual content items can be viewed as seen below:

>>> ds.Contents[0]

(0040, a010) Relationship Type CS: 'CONTAINS'

(0040, a040) Value Type CS: 'TEXT'

(0040, a043) Concept Name Code Sequence 1 item(s) —-

(0008, 0100) Code Value SH: 'RE.02'

(0008, 0102) Coding Scheme Designator SH: 'BUMIL'

(0008, 0104) Code Meaning LO: 'Request'

(0008, 010c) Coding Scheme UID UI: 1.2.276.0.7230010.3.0.0.1

————

(0040, a160) Text Value UT: 'Test Request'

To see the text value of the item on the report screen:

>>> ds.Contents[0]. TextValue

Test Request

These individual text items come from the SR Template used to for the SR file. The radiologist has a screen to input the text values in these items. Textview items are used in Glade to hold this data. The buffer in the text view is taken and when the radiologists finished writing, selects 'Set Request' and the relevant report item is set to the text input entered.

For the request case:

>>> buff1 = textview1.get_buffer()

text1 = buff1.get_text(buff1.get_start_iter(), buff1.get_end_iter())

dsedit=dicom.read_file('SR filename')

dsedit.Contents[0].TextValue=text1

filewriter.write_file(ds, dsedit, WriteLikeOriginal=True)

For secure file operation firstly a copy of the SR document is changed, then it is written to the file.

All four of the items in the text is viewed inside a treeview by use of a while loop to show the radiologist what is the actual text for the SR. Since four text items are present:

a, b = 0, 1

while a < 4:

    text = ds.Contents[a].TextValue

    label = ds.Contents[a].ConceptNameCodes[0].CodeMeaning

    c=[]

    c.append(label)

    c.append(text)

    treeview.append(None, c)

    a = a + b

A.2 DICOM Tag Analysis and Selection

The text elements in the DICOM Structured Report are located in (0040, a730) tag of which long text meaning is Content Sequence. This Content Seuence has four different containers for text areas of Basic Text SR Storage.

For the first container the tags are as follows:

(0040, a010) Relationship Type   CS: 'CONTAINS'

(0040, a040) Value Type   CS: 'TEXT'

(0040, a043) Concept Name   Code Sequence

The Text storage in this tag is denoted with (0008, 0104) Code Meaning: 'Request' and the "(0040, a160) Text Value" is taken. This element can be with dicompy module read_file function, utilizing the names of the values:

dcm=dicom.read_file(sr_file)

request=dcm.Contents[0].TextValue

Here the "Contents" stands for Content Sequence. In python pydicom module has such a convention that these abbrevations are done like also in ConceptNameCodes for Concept Name Code Sequence. Also tags with two words are merged like PatientID, which stands for Patient ID.

The tags and values can be taken into a treeview on the GTK with:

```
for i in self.b:

    a=[]

    a.append(i.tag)

    a.append(i.description())

    a.append(i.VR)

    a.append(i.value)
```

self.store.append(None, a)

Then this self.store is written into the treeview as seen below. Patient identification data is also taken through tags' values and written in the DICOM SR file.

self.store = gtk.TreeStore(str, str, str, str)

self.treeview3.set_model(model=self.store)

A.3 DICOM SR Text Field Container View, Edit and Store

To view the SR value in a text field container in python:

>>> ds=dicom.read_file("SR.dcm")

>>> ds.Contents[1].TextValue

'Sagittal T1 and T2 weighted images and fast spin echo and coronal T2 weighted images are acquired.'

Four functions are programmed in the software to set the entered text into the relevant field. Each of these functions gets the text inside the textview box of the GUI in GTK, for procedure description for example, sets it to the occurrence of the DICOM SR document:

self.textview2= self.window1_temel.get_widget("textview2")

buff2 = self.textview2.get_buffer()

text2 = buff2.get_text(buff2.get_start_iter(), buff2.get_end_iter())

#print text1 self.z=dicom.read_file(self.a)

self.z.Contents[1].TextValue=text2

In the end of the function the file is saved by filewriter function of pydicom module.

filewriter.write_file(self.a, self.z, WriteLikeOriginal=True)

A.4 DICOM SR Set Image References

To set the image references elements of DICOM SR document is modified and saved like in the Text areas as seen below where file selected 3 is functional image and 4 is DICOM image of the patient:

addr2=self.filechooserbutton3.get_filename()

self.dicom_file=self.filechooserbutton4.get_filename()

self.vlat.Contents[4].ConceptNameCodes[0].CodeValue=self.dicom_file

self.vlat.Contents[5].ConceptNameCodes[0].CodeValue=addr2

Appendix B. PROCESSING OF fMRI STATISTICAL MAPS

B.1   View and Fuse Anatomical Image

The image needs to be converted to a pixel array as seen from the source code below, since matplotlib accepts to plot array data, so the pixel array data is taken from the dicom file by using the dicom_read function of dicompy, after filename of SR is got as self.aa the image name is taken from the SR file instance self.bb and self.cc the image file is read into self.dd, self.dd.pixel_array with pcolor function is used afterby:

self.aa=self.filechooserbutton1.get_filename()

self.bb=dicom.read_file(self.aa)

self.cc=self.bb.Contents[5].ConceptNameCodes[0].CodeValue

self.dd=dicom.read_file(self.cc)

In Python there are two well fit functions to visualize array data for an image. The module matplotlib has the GTKBackend functions to support visualization of array data within a figure canvas, pcolor function has many basic colormaps from graysclae to spectral to show the image in different colour schema. The module matplotlib is very useful in enabling the visualization for functional neuroimaging files. matplotlib is a python 2D plotting library which produces publication quality figures in a variety of hardcopy formats and interactive environments across platforms as the developers define it. "pcolor" fuction inside a Figure canvas is used to show basic DICOM image files and single functional images associated with them as seen in the source code sample for this study below:

from matplotlib.figure import Figure, figaspect

from matplotlib.axes import Subplot

```
from matplotlib import cm, pylab

. . . . . . . . . . . . . . . . .

w, h = figaspect(self.dd.pixel_array)

self.figure = Figure(figsize=(w,h), dpi=100)

self.axis = self.figure.add_subplot(111)

self.axis.set_title('Functional Image')

self.axis.grid(True)

self.canvas = FigureCanvasGTK(self.figure) # a gtk.DrawingArea

self.canvas.show() builder = gtk.Builder()

self.graphview = self.window1_temel.get_widget("plot")

self.graphview.pack_start(self.canvas, True, True)

self.axis.pcolor(self.dd.pixel_array, cmap=cm.spectral)
```

B.2   Indextracker Function

Index tracker basicly takes the signal of mouse scroll button and adopts in to change some specific value and update the image plotted. The changed value is used as alpha here within this function to fuse anatomic and functional images and the interval was 0.1. The alpha is between 0.0 and 1.0, where 0 is fully first image shown, 1.0 second is shown.

Since indextracker implementation is very important and used many times for

this study below it is presented how it is implemented fpr the case of two images with different colormaps first intended to be the anatomic (self.sa.pixel_array) and the second, functional image (self.sa2.pixel_array):

```
Z1 = self.sa.pixel_array

Z2 = self.sa2.pixel_array

class IndexTracker:

    def __init__(self, ax, X1, X2):

        self.ax = ax

        ax.set_title('use scroll when to change transperancy')

        X1=Z1, X2=Z2

        self.X1 = X1, self.X2 = X2

        self.alp=1

        self.ind = int(self.alp/2)

        self.im = ax.imshow(self.X1[:,:], cmap=cm.gray)

        hold(True)

        self.im2 = ax.imshow(self.X2[:,:], cmap=cm.spectral, alpha=self.ind)

        print self.ind

        self.update()
```

```python
def onscroll(self, event):

    print event.button # event.step

    if event.button=='up':

        self.ind = numpy.clip(self.ind+.1, 0, self.alp+.1)

    else:

        self.ind = numpy.clip(self.ind-.1, 0, self.alp-.1)

    self.update()

def update(self):

    self.im.set_data(self.X1[:,:])

    self.im2.set_data(self.X2[:,:])

    ax.set_ylabel('Alpha %s'%self.ind)

    self.im = ax.imshow(self.X1[:,:], cmap=cm.gray)

    hold(True)

    self.im2 = ax.imshow(self.X2[:,:], cmap=cm.spectral, alpha=self.ind)

    self.im.axes.figure.canvas.draw()

    hold(True)

    self.im2.axes.figure.canvas.draw()
```

This class is called triggered by a mouse scroll as seen below, so the alpha value

changes and the plot is blended with the different ratios of an overlay that can be selected by the user.

```
fig = figure()

ax = fig.add_subplot(111)

X1 = Z1, X2 = Z2

tracker = IndexTracker(ax, X1, X2)

fig.canvas.mpl_connect('scroll_event', tracker.onscroll)

show()
```

B.3   View Mosaic Images Slice by Slice

Non-DICOM tags are found in proprietary Siemens fields. While proprietary, these are stored as simple ASCII text surrounded by the strings "### ASCCONV BEGIN ###" and "### ASCCONV END ###", so it is easy to parse them with simple string operations. I This can be also referred as "ASCII header" as explained below.

The Phase Encode FoV ("sSliceArray.asSlice[0].dPhaseFOV") and the Readout FoV ("sSliceArray.asSlice[0].dReadoutFOV") from the ASCII header is used, the Phase Encode Direction (18,1312), and the row and column resolutions (28,30) to compute an expected number of rows and columns. Then these numbers are compared to the number of rows (28,10) and columns (28,11) in the image. If they are the same, then it is not a mosaic. If it is not, it is understood to be a mosaic file [76].

Once the file is checked to be mosaic, slice header info needs to be taken and the relevant values of slice number should be taken from this ASCII header with string operations as seen from the source code part taken below (IR.02 is a mosaic dicom

file):

```
self.rimg='/root/Desktop/IR.02'

self.img=dicom.read_file(self.rimg)

self.arraysize= self.img.pixel_array.shape

self.shdr=self.img[0x29,0x1020].value #Slice Header info

self.pos=self.shdr.find('sSliceArray.lSize = ')

self.inc=len('sSliceArray.lSize = ')

self.slpos = self.pos + self.inc

self.snum = self.shdr[self.slpos]

lim = int(self.snum)
```

Once we know the slice number we need to fit this number into the smallest square of an integer possible, like 2 for 3 slices, 3 for 6 slices. This is found by:

```
cnt=0

while (cnt*cnt) < lim:

    cnt = cnt + 1
```

Now we have cnt as the shape of the square for the mosaic image. We then get the array size and find how much pizels to be taken from here for each slice:

```
(r, c)=self.arraysize
```

amy=int(r/cnt), amz=int(c/cnt)

array_mos= zeros([(cnt*cnt), (amy), (amz)], Int)

This was the easy part to get the dimensions. Now assigning these to different arrays is really hard, all dimensions need to be thought and assigned to frames carefully:

i=0, k=0, l=0

while i < cnt:

    k=0

    while k < cnt:

        for n in range(i*(amy), (i + 1)*(amz)):

            for x in range(k*(amy), (k + 1)*(amz)):

                array_mos[l][n-(i*(amy))][x-(k*(amz))]=self.mos[n][x]

        k = k+ 1

        l = l + 1

    i=i+ 1

Here the complexity of the algorithm can be seen from the indexes of the array_mos which needs to be taken care of the changing frames in a square as 3x3 for example, left most is 0th frame, where one upper is 3rd frame, going firstly from x direction 0,1,2 to y first frame 3,4,5.

B.4 NIfTI File Handling and Visualization

Below it can be seen how the data is taken and assigned to arrays with pynifti:

addr='/root/Desktop/mean_FA.nii.gz'

addr2='/root/Desktop/mean_FA_skeleton_mask.nii.gz'

self.ni=nifti.NiftiImage(addr)

self.ni2=nifti.NiftiImage(addr2)

self.ni_data=self.ni.data, self.ni2_data=self.ni2.data

Afterby these data arrays are visualized just as explained in the image fusion chapter, skeleton ask is overlayed by mean_FA DTI image. Alpha is changed with indextracker class. The overlay part is shown below inside update function of Index-Tracker class, here self.X1 is a slice from self.ni_data and self.X2 from self.ni2_data, imshow function is used to out the figure in the canvas to be shown:

self.im.set_data(self.X1[:,:])

self.im2.set_data(self.X2[:,:])

ax.set_ylabel('Alpha %s'%self.ind)

self.im = ax.imshow(self.X1[:,:], cmap=cm.gray)

hold(True)

self.im2 = ax.imshow(self.X2[:,:], cmap=cm.jet, alpha=self.ind)

self.im.axes.figure.canvas.draw()

hold(True)

self.im2.axes.figure.canvas.draw()

A brief information about pynifti module is that PyNIfTI can read and write any file format supported by libniftiio (provides access to the most important features of the NIfTI-1 data format and libniftiio capabilities). This includes NIfTI (single and pairs) as well as ANALYZE files, both also in gzipped versions. PyNIfTI provides fast and convenient access to the image data via NumPy arrays. This should enable users to process image data with most numerical routines available for Python. The NumPy array automatically uses a datatype corresponding to the NIfTI image data [77].

B.5 Labeling Images Using Anatomical Atlas

addr3='/root/Desktop/atlas/HarvardOxford-cort-maxprob-thr25-2mm.nii.gz'

addr4='/root/Desktop/atlas/HarvardOxford-sub-maxprob-thr25-2mm.nii.gz'

self.ni3=nifti.NiftiImage(addr3), self.ni4=nifti.NiftiImage(addr4)

self.X3=self.ni3.data, self.X4 =self.ni4.data

IndexTracker class is used to overlay functional image onto the anatomical with constant alpha value, this time scroll is used to change slices as seen in the code below:

def onscroll(self, event):

    if event.button=='up':

        self.ind = numpy.clip(self.ind+1, 0, self.slices-1)

    else:

self.ind = numpy.clip(self.ind-1, 0, self.slices-1)

self.update()

In order to show the anatomical region in both of the atlases, in this module a new definition of functions are needed inside the class IndexTracker. Whrn the user presses and releases left mouse button, a signal is sent to these functions to get the location of the pointer so as to match with the anatomical atlas. The so called "events" to get clicked signal is seen in the source code from the software developed below:

self.im.axes.figure.canvas.add_events(gtk.gdk.BUTTON_PRESS_MASK)

self.im.axes.figure.canvas.add_events(gtk.gdk.BUTTON_RELEASE_MASK)

self.im.axes.figure.canvas.connect('button_press_event', self.button_pressed)

self.im.axes.figure.canvas.connect('button_release_event', self.button_released)

The mouse pointer location as taken self.Xabs and self.Yabs is used with self.ind coming from Indextracker (slice number), in order to find the labelvalue of the anatomical atlas as seen below. You would notice that the x and y are swapped in the data taken from pyNifti. This is a typical property that the pyNifti data when converted to array is in (t, z, y, x) orientation.

self.labelvalue1 = self.X3[self.ind][self.Yabs][self.Xabs]

self.labelvalue2 = self.X4[self.ind][self.Yabs][self.Xabs]

self.HOcort=HO_cort[self.labelvalue1]

self.HOsub=HO_sub[self.labelvalue2]

The labels corresponding to these labelvalues are defined in a string array so that it is looked up and assigned to the store variables. Then these stored elements are

taken and viewed in a treeview as seen in Figure 4.5, and the code is below:

```
lab=[]

lab.append(self.HOcort)

lab.append(self.HOsub)

lab.append(self.textlat)

self.storeho.append(None, lab)

self.treeview4= self.window1_temel.get_widget("treeview4")

self.storeho = gtk.TreeStore(str, str, str)

self.treeview4.set_model(model=self.storeho)
```

B.6   Visualizing Waveform of Timepoints

Adrr is the nifti image file name below.

```
self.arraysize2= self.ni_data.shape

self.ni=nifti.NiftiImage(addr)

self.ni_data=self.ni.data

self.W1=self.ni_data

self.arraysize2= self.ni_data.shape

(d, e, f, g)=self.arraysize2
```

k=0

t2 = arange(0, d, 1)

for k in t2:

   self.A[k] = self.W1[k][self.ind][0][0]

axw.plot(t2, self.A)

So for all timepoints in the NifTi file the pixel value is taken and plotted.

B.7   Visualizing Waveform of Images

A new function to see which region is selected on the canvas, draw rubberband, is implemented in this module [78]. The normalization method and the arrangement are as seen below:

def button_pressed(self, axis, event):

if event.button == 1:

   self.x1, self.y1, state = event.window.get_pointer()

   self.y1 = self.axis.figure.canvas.allocation.height - self.y1

   self.Xabs = (self.x1-170)*d/964 #170, 56 is the left corner of canvas

   self.Yabs=(self.y1-56)*c/429 # 170+964 , 429 +56 is the top right corner of canvas

   self.draw_rubberband(event, self.x1, self.y1, self.x2, self.y2)

As these coordinates are taken when the button is pressed, the released coordinates are also necessary to take the difference and to plot the selected region. Button released event is handled as shown here:

```
def button_released(self, axis, event):

    if event.button == 1:

        self.x2, self.y2, state = event.window.get_pointer()

        self.y2 = self.axis.figure.canvas.allocation.height - self.y2

        self.Xabs2 = (self.x2-170)*d/864 #170, 56 is the left corner of canvas

        self.Yabs2=(self.y2-56)*c/429

        self.draw_rubberband(event, self.x1, self.y1, self.x2, self.y2)

        xmin=min(self.Xabs, self.Xabs2)

        xmax=max(self.Xabs, self.Xabs2)

        ymin=min(self.Yabs, self.Yabs2)

        ymax=max(self.Yabs, self.Yabs2)

        self.roi_an= zeros([int(xmax-xmin), int(ymax-ymin)])

        self.roi_cor= zeros([int(xmax-xmin), int(ymax-ymin)])

        self.roi_ni= zeros([int(xmax-xmin), int(ymax-ymin)])

        for x in range(xmin, xmax):
```

```
        for y in range(ymin, ymax): # swapped axis for draw

            self.roi_an[int(x-xmin), int(y-ymin)]=self.X1[self.ind][y,x]

            self.roi_ni[int(x-xmin), int(y-ymin)]=self.X2[y,x]

    print self.roi_an.shape

    print self.roi_ni.shape

self.roi_cor = rot90(self.roi_an) # rotate as to correct axis

self.axis.pcolor(self.roi_cor, cmap=cm.gray)

hold(True)

fig = figure()

axis = fig.add_subplot(111)

im_an = axis.imshow(self.roi_cor, cmap=cm.gray, interpolation=None)

hold(True)

show()
```

Appendix C. fMRI ANALYSIS

C.1   Lateralization Index Calculation

Where lat is the index and the other variables are taken from the data as seen in the source code of the developed module as seen below:

self.filechooserbutton3=self.window1_temel.get_widget("filechooserbutton3")

addr2=self.filechooserbutton3.get_filename()

self.ni2=nifti.NiftiImage(addr2)

self.ni2_data=self.ni2.data

Z2 = self.ni2_data

(s, v, h)=Z2.shape

The Nifti file of the functional images is taken and assigned to Z2 array, then:

for z in range (s):

    for y in range (v):

        for x in range (h):

            if Z2[z][y][x]>0:

                if x>h/2:

                    left_active=left_active+1.0

else:

right_active=right_active+1.0

Inside these three for loops the active regions on the left and right side is determined as their count.

C.2 Maximum Z Value Calculations

In this study the maximum z value and the location is detected as the following loops and the algorithm shows and plotted with the waveform of all timepoints of the maximum Z point. Z2 is the same functional array data as the previous part:

zmax=Z2.max()

for z in range (s):

for y in range (v):

for x in range (h):

if Z2[z][y][x]>0:

if Z2[z][y][x]==zmax:

zz, yy, xx=z, y, x

zz, yy, xx is the coordinates of the maximum z point. Then the label value is found as:

self.labelvalue1 = self.X3[zz][yy][xx]

self.labelvalue2 = self.X4[zz][yy][xx]

Where X3 and X4 are the cortical and subcortical anatomic atlases data taken with pyNifTi. Once the labelvalue is found the procedure is easy, it is looked up from the string array which anatomical region is there.

For the plot of the maximum Z point Axial, sagittal and Coronal planes are plotted with the waveform as seen in Figure 5.2. X1 is the anatomical template array data taken with pynifti:

axis1.pcolor(self.X1[:,:,xx], cmap=cm.gray)

axis2.pcolor(self.X1[:,yy,:], cmap=cm.gray)

axis3.pcolor(self.X1[zz,:,:], cmap=cm.gray)

d=zz, k=0

self.A=zeros(d)

t2 = arange(0, d, 1)

for k in t2:

    self.A[k] = self.X2[k][yy][xx]

axw.plot(t2, self.A)

self.canvas.show()

C.3   Active Anatomical Region Analysis

The algorithm in calculating voxel counts and taking the regions is inside three loops with all indexes taken into account with the active voxels as will be seen in the source code part taken from the programs one of the core modules, analyzeZ:

```
self.labvalc=zeros(49, dtype=float64)

self.labvals=zeros(59, dtype=float64)

for z in range (s):

    for y in range (v):

        for x in range (h):

            if Z2[z][y][x]>0:

                self.active.append((z, y, x))

                if (z, y, x) in self.active:

                    self.labelcor.append(Z3[z][y][x])

                    self.labelsub.append(Z4[z][y][x])

                    Z2[z][y][x]=Z2[z][y][x]+0.0

                    ac=int(Z3[z][y][x])

                    as=int(Z4[z][y][x])

                    self.labvalc[ac]=self.labvalc[ac]+Z2[z][y][x]+0.0

                    self.labvals[as]=self.labvals[as]+Z2[z][y][x]+0.0

            if z not in self.actslice:

                self.actslice.append(z)
```

Here both the active points, then the label numbers of these active points are

taken, anfd the z values of that region is set into the array. Then the number of z values nonzero is found and the means are calculated for cortical and subcortical are as seen below:

```
t=0

for t in range(49):

    self.numlabelcor[t]=self.labelcor.count(t)

    if self.labelcor.count(t)!=0:

        self.labelvalc_mean[t]=(self.labvalc[t]/(self.numlabelcor[t]))

ts=0

for ts in range(59):

    self.numlabelsub[ts]=self.labelsub.count(ts)

    if self.labelsub.count(ts)!=0:

        self.labelvals_mean[ts]=(self.labvals[ts]/(self.numlabelsub[ts]))
```

These values are set to the treeviews as seen below to be visualized in the GUI: For slice numbers with active voxels:

```
self.storeslice.clear()

labslice=[]

labslice.append(self.actslice)

self.storeslice.append(None, labslice)
```

For cortical regions with active slices, active voxel count and voxel's mean values:

for sl in range(49):

    if self.numlabelcor[sl]!=0:

        lablabel=[]

        lablabel.append(HO_cort[sl])

        lablabel.append(self.numlabelcor[sl])

        lablabel.append(self.labelvalc_mean[sl])

        self.storelabel.append(None, lablabel)

And lastly for subcortical regions:

for sls in range(59):

    if self.numlabelsub[sls]!=0:

        lablabels=[]

        lablabels.append(HOsub[sls])

        lablabels.append(self.numlabelsub[sls])

        lablabels.append(self.labelvals_mean[sls])

        self.storelabel2.append(None, lablabels)

C.4   Report Preperation and Output

The source code part seen below from the software developed gets the patient id data from the DICOM image header to be inserted into the report.

self.filechooserbutton4=self.window1_temel.get_widget("filechooserbutton4")

self.dicom_file=self.filechooserbutton4.get_filename()

#GET PATIENT DATA ———————————————————————————

self.dcm=dicom.read_file(self.dicom_file)

self.patid=self.dcm.PatientID

self.patname=self.dcm.PatientsName

self.patbdate=self.dcm.PatientsBirthDate

self.patsex=self.dcm.PatientsSex

self.studydate=self.dcm.StudyDate

#———————————— GET REPORT DATE————————————————-

now=datetime.datetime.now()

self.repdate=str(now.year)+'.'+str(now.month)+'.'+str(now.day)

self.repname='/root/Desktop/REPORTS/'+ str(self.patid)+ '_'+ str(self.studydate)+ '_Z-Report'

After the patient information is readily taken to be inserted into the report, the text part is also retrieved from the SR written:

self.request=self.vlat.Contents[0].TextValue

self.procdesc= self.vlat.Contents[1].TextValue

self.finding= self.vlat.Contents[2].TextValue

self.impression=self.vlat.Contents[3].TextValue

All these preperations are for the second format of report output to be available for the user as printable, readable in every platform, either Linux or Windows, in a rich text format RTF. pyrtf module is used to prepare the report in every platform readable and printable format.

REFERENCES

1. Parry, A., and P. Matthews, "Functional magnetic resonance imaging (fMRI): A window into the brain," *Retrieved July*, Vol. 25, p. 8, 2002.

2. "fMRI Additions." http://users.fmrib.ox.ac.uk, 2009.

3. Smith, S., "Overview of fMRI analysis," *British Journal of Radiology*, Vol. 77, no. Special Issue 2, pp. 167–165, 2004.

4. "Digital Imaging and Communications in Medicine (DICOM)." ftp://medical.nema.org/dicom/2008, 2008.

5. "Eric Integrated Development Environment." http://sourceforge.net/projects/eric-ide/, 2010.

6. "Python Programming Language – Official Website." http://www.python.org/, 2010.

7. "Glade - A User Interface Designer." http://glade.gnome.org, 2010.

8. Finger, S., *Minds Behind the brain - A History of the Pioneers and Their Discoveries*, Oxford University Press, 2000.

9. Zola-Morgan, S., "Localization of brain function: The legacy of Franz Joseph Gall," *Annual Review of Neuroscience*, Vol. 18, pp. 359–383, 1995.

10. Amunts, K., G. Schlaug, L. Jancke, H. Steinmetz, A. Schleicher, A. Dabringhaus, and K. Zilles, "Motor cortex and hand motor skills: Structural compliance in the human brain," *Human Brain Mapping*, Vol. 5, no. 3, pp. 206–215, 1997.

11. Maguire, E., and C. Mummery, "Differential modulation of a common memory retrieval network revealed by positron emission tomography," *Hippocampus*, Vol. 9, no. 1, pp. 54–61, 1999.

12. Gjedde, A., "Brain energy metabolism and the physiological basis of the haemodynamic response," *Functional MRI: An Introduction to methods*, pp. 37–65, 2001.

13. Roy, C., and C. Sherrington, "On the regulation of the blood-supply of the brain," *The Journal of Physiology*, Vol. 11, no. 1, p. 85, 1890.

14. Shulman, R., "Functional imaging studies: linking mind and basic neuroscience," *American Journal of Psychiatry*, Vol. 158, no. 1, p. 11, 2001.

15. Buxton, R., and L. Frank, "A model for the coupling between cerebral blood flow and oxygen metabolism during neural stimulation," *Journal of Cerebral Blood Flow & Metabolism*, Vol. 17, no. 1, pp. 64–72, 1997.

16. Ogawa, S., R. Menon, D. Tank, S. Kim, H. Merkle, J. Ellermann, and K. Ugurbil, "Functional brain mapping by blood oxygenation level-dependent contrast magnetic resonance imaging. A comparison of signal characteristics with a biophysical model," *Biophysical Journal*, Vol. 64, no. 3, pp. 803–812, 1993.

17. Logothetis, N., J. Pauls, M. Augath, T. Trinath, and A. Oeltermann, "Neurophysiological investigation of the basis of the fMRI signal," *Nature*, Vol. 412, no. 6843, pp. 150–157, 2001.

18. Rees, G., K. Friston, and C. Koch, "A direct quantitative relationship between the functional properties of human and macaque V5," *Nature Neuroscience*, Vol. 3, no. 7, pp. 716–723, 2000.

19. An, H., and W. Lin, "Quantitative measurements of cerebral blood oxygen saturation using magnetic resonance imaging," *Journal of Cerebral Blood Flow Metabolism*, Vol. 20, no. 8, pp. 1225–1236, 2000.

20. Schlosser, M., N. Aoyagi, R. Fulbright, J. Gore, and G. McCarthy, "Functional MRI studies of auditory comprehension," *Human Brain Mapping*, Vol. 6, no. 1, pp. 1–13, 1998.

21. Tootell, R., J. Mendola, N. Hadjikhani, P. Ledden, A. Liu, *et al.*, "Functional analysis of V3A and related areas in human visual cortex," *Journal of Neuroscience*, Vol. 17, no. 18, p. 7060, 1997.

22. R. Kurth, K. Villringer, B. M. J. S. J. B. G. C. A. V., and K. Wolf, "FMRI assessment of somatotopy in human Brodman area 3b by electrical finger stimulation," *Neuroreport*, Vol. 9, no. 18, pp. 209–212, 1998.

23. Kim, S., J. Ashe, A. Georgopoulos, H. Merkle, J. Ellermann, R. Menon, S. Ogawa, and K. Ugurbil, "Functional imaging of human motor cortex at high magnetic field," *Journal of neurophysiology*, Vol. 69, no. 1, pp. 297–302, 1993.

24. Nudo, R., and R. Masterton, "Stimulation-induced [* C] 2-deoxyglucose labeling of synaptic activity in the central auditory system," *J Comp Neurol*, Vol. 245, pp. 553–565, 1986.

25. H. H. Batjer, L. R. Caplan, L. F. R. G. G. T. A. K., and W. L. Young, *The Relation Between Brain Function and Cerebral Blood Flow and Metabolism*, pp. 23–40. Lippincott-Raven, 1997.

26. Jezzard, P., P. Matthews, and S. Smith, *Functional MRI: An introduction to methods*, Oxford University Press, 2001.

27. Friston, K., A. Holmes, K. Worsley, J. Poline, C. Frith, R. Frackowiak, *et al.*, "Statistical parametric maps in functional imaging: a general linear approach," *Human Brain Mapping*, Vol. 2, no. 4, pp. 189–210, 1995.

28. McKeown, M., S. Makeig, G. Brown, T. Jung, S. Kindermann, A. Bell, and T. Sejnowski, "Analysis of fMRI data by blind separation into independent spatial components," *Human Brain Mapping*, Vol. 6, no. 3, pp. 160–188, 1998.

29. Beckmann, C., and S. Smith, "Probabilistic independent component analysis for functional magnetic resonance imaging," *IEEE transactions on medical imaging*, Vol. 23, no. 2, pp. 137–152, 2004.

30. Clare, S., M. Humberstone, J. Hykin, L. D Blumhardt, R. Bowtell, and P. Morris, "Detecting activations in event-related fMRI using analysis of variance," *Magnetic Resonance in Medicine*, Vol. 42, no. 6, pp. 1117–1122, 1999.

31. Beckmann, C., M. Jenkinson, and S. Smith, "General multilevel linear modeling for group analysis in FMRI," *Neuroimage*, Vol. 20, no. 2, pp. 1052–1063, 2003.

32. Worsley, K., and K. Friston, "Analysis of fMRI time-series revisited again," *Neuroimage*, Vol. 2, no. 3, pp. 173–181, 1995.

33. Pinheiro, J., and D. Bates, *Mixed-effects models in S and S-PLUS*, Springer Verlag, 2000.

34. Woolrich, M., T. Behrens, C. Beckmann, M. Jenkinson, and S. Smith, "Multilevel linear modelling for FMRI group analysis using Bayesian inference," *Neuroimage*, Vol. 21, no. 4, pp. 1732–1747, 2004.

35. Friston, K., A. Holmes, and K. Worsley, "Comments and Controversies," *Neuroimage*, Vol. 10, pp. 1–5, 1999.

36. Talairach, J., and P. Tournoux, *Co-planar stereotaxic atlas of the human brain: 3-dimensional proportional system: an approach to cerebral imaging*, Thieme, 1988.

37. Collins, D., P. Neelin, T. Peters, and A. Evans, "Automatic 3D intersubject registration of MR volumetric data in standardized Talairach space," *Journal of computer assisted tomography*, Vol. 18, no. 2, p. 192, 1994.

38. Smith, S., M. Jenkinson, M. Woolrich, C. Beckmann, T. Behrens, H. Johansen-Berg, P. Bannister, M. De Luca, I. Drobnjak, D. Flitney, *et al.*, "Advances in functional and structural MR image analysis and implementation as FSL," *Neuroimage*, Vol. 23, pp. 208–219, 2004.

39. "Atlas Descriptions." http://www.fmrib.ox.ac.uk/fsl/fslview/atlas-descriptions.html, 2009.

40. Corbetta, M., E. Akbudak, T. Conturo, A. Snyder, J. Ollinger, H. Drury, M. Linenweber, S. Petersen, M. Raichle, D. Van Essen, *et al.*, "A common network of functional areas for attention and eye movements," *Neuron-Cambridge MA-*, Vol. 21, pp. 761–773, 1998.

41. Toga, A., *Brain Warping*, pp. 385–386. Academic Press, 1999.

42. Miller, M., A. Trouvé, and L. Younes, "On the Metrics and Euler - Lagrange Equations of Computational Anatomy," *Annual Review of Biomedical Engineering*, Vol. 4, no. 1, pp. 375–405, 2002.

43. Thompson, P., R. Woods, M. Mega, and A. Toga, "Mathematical/computational challenges in creating deformable and probabilistic atlases of the human brain," *Human brain mapping*, Vol. 9, no. 2, pp. 81–92, 2000.

44. Van Essen, D., "Windows on the brain: the emerging role of atlases and databases in neuroscience," *Current Opinion in Neurobiology*, Vol. 12, no. 5, pp. 574–579, 2002.

45. Fischl, B., M. Sereno, R. Tootell, and A. Dale, "High-resolution intersubject averaging and a coordinate system for the cortical surface," *Human Brain Mapping*, Vol. 8, no. 4, pp. 272–284, 1999.

46. Van Essen, D., J. Lewis, H. Drury, N. Hadjikhani, R. Tootell, M. Bakircioglu, and M. Miller, "Mapping visual cortex in monkeys and humans using surface-based atlases," *Vision research*, Vol. 41, no. 10-11, pp. 1359–1378, 2001.

47. Binder, J., S. Swanson, T. Hammeke, G. Morris, W. Mueller, M. Fischer, S. Benbadis, J. Frost, S. Rao, V. Haughton, *et al.*, "Determination of language dominance using functional MRI," *Neurology*, Vol. 46, pp. 978–984, 1996.

48. Wada, J., and T. Rasmussen, "Intracarotid injection of sodium amytal for the lateralization of cerebral speech dominance," *Journal of Neurosurgery: Pediatrics*, Vol. 17, no. 2, 1960.

49. Loring, D., K. Meador, G. Lee, and D. King, *Amobarbital effects and lateralized brain function: the Wada test*, Springer-Verlag, 1991.

50. Penfield, W., and H. Jasper, "Epilepsy and the functional anatomy of the human brain," *Southern Medical Journal*, Vol. 47, no. 7, p. 704, 1954.

51. Rasmussen, T., and B. Milner, "The role of early left-brain injury in determining lateralization of cerebral speech functions.," *Annals of the New York Academy of Sciences*, Vol. 299, p. 355, 1977.

52. Woods, R., C. Dodrill, and G. Ojemann, "Brain injury, handedness, and speech lateralization in a series of amobarbital studies," *Annals of neurology*, Vol. 23, no. 5, pp. 510–518, 2004.

53. Pardo, J., and P. Fox, "Preoperative assessment of the cerebral hemispheric dominance for language with CBF PET," *Human Brain Mapping*, Vol. 1, no. 1, pp. 57–68, 2004.

54. Jennum, P., L. Friberg, A. Fuglsang-Frederiksen, and M. Dam, "Speech localization using repetitive transcranial magnetic stimulation," *Neurology*, Vol. 44, no. 2, p. 269, 1994.

55. Dion, J., P. Gates, A. Fox, H. Barnett, and R. Blom, "Clinical events following neuroangiography: a prospective study," *Stroke*, Vol. 18, no. 6, p. 997, 1987.

56. Bandettini, P., E. Wong, R. Hinks, R. Tikofsky, and J. Hyde, "Time course EPI of human brain function during task activation," *Magnetic Resonance in Medicine*, Vol. 25, no. 2, pp. 390–397, 2005.

57. Rao, S., J. Binder, P. Bandettini, T. Hammeke, F. Yetkin, A. Jesmanowicz, L. Lisk, G. Morris, W. Mueller, L. Estkowski, *et al.*, "Functional magnetic resonance imaging of complex human movements," *Neurology*, Vol. 43, no. 11, pp. 2311–8, 1993.

58. Kwong, K., J. Belliveau, D. Chesler, I. Goldberg, R. Weisskoff, B. Poncelet, D. Kennedy, B. Hoppel, M. Cohen, and R. Turner, "Dynamic magnetic resonance imaging of human brain activity during primary sensory stimulation," *Proceedings of the National Academy of Sciences*, Vol. 89, no. 12, p. 5675, 1992.

59. Ogawa, S., D. Tank, R. Menon, J. Ellermann, S. Kim, H. Merkle, and K. Ugurbil, "Intrinsic signal changes accompanying sensory stimulation: functional brain mapping with magnetic resonance imaging," *Proceedings of the National Academy of Sciences of the United States of America*, Vol. 89, no. 13, p. 5951, 1992.

60. Turner, R., P. Jezzard, H. Wen, K. Kwong, D. Le Bihan, T. Zeffiro, and R. Balaban, "Functional mapping of the human visual cortex at 4 and 1.5 Tesla using deoxygenation contrast EPI," *Magnetic Resonance in Medicine*, Vol. 29, pp. 277–277, 1993.

61. Binder, J., S. Rao, T. Hammeke, F. Yetkin, A. Jesmanowicz, P. Bandettini, E. Wong, L. Estkowski, M. Goldstein, V. Haughton, *et al.*, "Functional magnetic resonance imaging of human auditory cortex," *Annals of Neurology*, Vol. 35, no. 6, pp. 662–672, 2004.

62. Binder, J., S. Rao, T. Hammeke, J. Frost, P. Bandettini, and J. Hyde, "Effects of stimulus rate on signal response during functional magnetic resonance imaging of auditory cortex," *Cognitive Brain Research*, Vol. 2, no. 1, pp. 31–38, 1994.

63. Hammeke, T., F. Yetkin, W. Mueller, G. Morris, V. Haughton, S. Rao, and J. Binder, "Functional magnetic resonance imaging of somatosensory stimulation," *Neurosurgery*, Vol. 35, no. 4, p. 677, 1994.

64. Liegeois, F., A. Connelly, J. Cross, S. Boyd, D. Gadian, F. Vargha-Khadem, and T. Baldeweg, "Language reorganization in children with early-onset lesions of the left hemisphere: an fMRI study," *Brain*, Vol. 127, no. 6, p. 1229, 2004.

65. Nagata, S., K. Uchimura, W. Hirakawa, and J. Kuratsu, "Method for quantitatively evaluating the lateralization of linguistic function using functional MR imaging," *American Journal of Neuroradiology*, Vol. 22, no. 5, p. 985, 2001.

66. Staudt, M., W. Grodd, G. Niemann, D. Wildgruber, M. Erb, and I. Krageloh-Mann, "Early left periventricular brain lesions induce right hemispheric organization of speech," *Neurology*, Vol. 57, no. 1, p. 122, 2001.

67. Oosterwijk, H., and P. T. Gihring, *DICOM Basics, 3nd ed.*, OTech, Inc., 2002.

68. Pianykh, and O. S., *Digital Imaging and Communications in Medicine (DICOM) – A Practical Introduction and Survival Guide*, Springer, 2008.

69. Clunie, D. A., *DICOM Structured Reporting*, PixelMed Publishing, 2001.

70. "DicomScope - Dicom Viewer." http://dicom.offis.de/dscope.php.en, 2009.

71. "FMRIB Software Library." http://www.fmrib.ox.ac.uk/fsl/, 2008.

72. Woolrich, M., S. Jbabdi, B. Patenaude, M. Chappell, S. Makni, T. Behrens, C. Beckmann, M. Jenkinson, and S. Smith, "Bayesian analysis of neuroimaging data in FSL," *NeuroImage*, Vol. 45, pp. 173–186, 2009.

73. "AFNI pages – Free software for analysis and display of fMRI data." http://afni.nimh.nih.gov, 2010.

74. Lehericy, S., L. Cohen, B. Bazin, S. Samson, E. Giacomini, R. Rougetet, L. Hertz-Pannier, D. Le Bihan, C. Marsault, and M. Baulac, "Functional MR evaluation of

temporal and frontal language dominance compared with the Wada test," *Neurology*, Vol. 54, no. 8, p. 1625, 2000.

75. Yoo, S., X. Wei, C. Dickey, C. Guttmann, and L. Panych, "Long-term reproducibility analysis of fMRI using hand motor task," *International Journal of Neuroscience*, Vol. 115, no. 1, pp. 55–77, 2005.

76. "Coregistration Siemens DICOM data." http://www.mailarchive.com/ freesurfer-nmr.mgh.harvard.edu/msg08118.html, 2008.

77. "pyNifTI Pythonic Access to NifTI and ANALYZE files." http://niftilib.sourceforge.net/pynifti/contents.html, 2009.

78. "wxPython Graphics Drawing a rubberband over a Canvas." http://aspn.activestate.com/ASPN/Cookbook/Python/Recipe/189744, 2003.