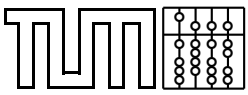


Final Report: Interdisciplinary Project in Medicine

Cardiac PET - SPECT Registration

Brian Jensen

March 19, 2009

Advisor: Prof. Dr. Navab Nassir
Supervisor: Axel Martinez-Moller, Dr. Stephan Nekolla, Darko Zikic

Technische Universität München
Departmentof Computer Science
Chair for Computer Aided Medical Procedures and Augmented Reality
wwwnavab.cs.tum.edu

University Hospital Rechts der Isar
Department of Nuclear Medicine
www.nuk.med.tu-muenchen.de

Abstract

Nuclear cardiology is a subfield of cardiology that uses nuclear imaging modalities to assess a patient's cardiac function. Positron Emission Tomography (PET) and Single Photon Emission Computed Tomography (SPECT) are two nuclear imaging modalities which are often used to show complimentary functional aspects of the myocardium. Current clinical protocol for the assessment of cardiac viability and myocardial perfusion often requires that the scans be taken at different times or even using the two different modalities, which then must be registered into the same physical space. This report investigates various methods of automatic registration of nuclear cardiac images and analyzes several concrete implementations.

Contents

1 Introduction **1**

2 Background Information **2**
2.1 Nuclear Imaging Modalities . . . 2
2.1.1 Positron Emission Tomography . . . 2
2.1.2 Single Photon Emission Computed Tomography . . . 4
2.2 Nuclear Cardiology . . . 5
2.2.1 Myocardial Perfusion Imaging . . . 5
2.2.2 Myocardial Viability . . . 5
2.3 Medical Image Registration . . . 6
2.3.1 Intensity Based Registration . . . 8

3 Project Description **10**
3.1 Project Goals . . . 10
3.2 Previous Related Work . . . 13

4 Methods **14**
4.1 Equipment . . . 14
4.1.1 Test Data Set . . . 15
4.2 Image Registration . . . 15
4.2.1 Interpolator . . . 15
4.2.2 Transform . . . 16
4.2.3 Optimizer . . . 18
4.2.4 Similary Measure . . . 19
4.3 Frameworks . . . 22
4.3.1 CAMP Library . . . 22
4.3.2 ITK Framework . . . 23
4.3.3 Qt Framework . . . 28

5 Results **31**
5.1 Evaluation of the Camp Library . . . 31
5.2 Evaluation of the ITK Framework . . . 31
5.2.1 Viola-Wells Mutual Information . . . 31
5.2.2 Mattes Mutual Information . . . 35
5.2.3 Multi-resolution Registration . . . 37

6 Conclusions **40**
6.1 Discussion . . . 40
6.2 Final thoughts . . . 40

A PetSpectFusion Reference **42**
A.1 Introduction . . . 42
A.2 Installation Requirements . . . 42
A.3 Usage Modes . . . 42
A.4 Command Line Reference . . . 43
A.4.1 General . . . 43
A.4.2 Command Line Parameters . . . 44
A.5 Graphical User Interface Reference . . . 45
A.5.1 Features . . . 45
A.5.2 Basic Usage . . . 46
A.5.3 Advanced Functions . . . 48
A.6 Building PetSpectFusion from source . . . 50

Bibliography **52**

1 Introduction

The amount of computer aided diagnostic techniques used in the medical field today is quite extensive and ever growing. Three dimensional scans of a patient, showing information that would not otherwise be visible without surgery, are now one of the standard diagnostic tools for many branches of medicine. Especially in the area of nuclear cardiology are digital imaging tools of high value.

Nuclear cardiology designates the subfield of cardiology that uses nuclear imaging modalities to assess a patient's cardiac function. The nuclear imaging modalities applied in nuclear cardiology are able to show complementary physiological aspects of the tissue being examined. This can lead to problems though, as current clinical protocols can require patients to be scanned using different radio tracers or even using different scanners or at different times.

In order for the images to be of maximum diagnostic value, they need to be correctly aligned to one another, a process called registration, so that equivalent physiological structures correlate to one another. Only when the images are correctly registered can the information contained in both be correctly compared. Standard clinical procedure is for the images to be registered manually, which introduces a potential bias into the system. More desirable are automatic image registration methods, which could serve to improve reproducibility and potentially eliminate a bias in the registration process.

This report covers an interdisciplinary project conducted between the main field of computer science and the applied field of medicine. The goal of an interdisciplinary project is to gain insight into the problems and methods of an applied field and to use knowledge from the main field of study to deal with these problems. The project supervision was carried out as a partnership between the chair for computer aided medical procedures in the department of computer at the Technical University of Munich, and the department of nuclear medicine at the university hospital Rechts der Isar. The main goal of the project was to research and implement methods for fully automatic registration of nuclear cardiac images. It was found that this could be achieved satisfactorily and a program was written, called PetSpectFusion, to carry out the automatic registration. In addition an abstract was published and an oral presentation was held about the results achieved at the Society of Nuclear Medicine's annual conference in 2008.

2 Background Information

This chapter contains the necessary background information in order to understand the rest of this report. It was written for people versed in the field of computer science, but who do not necessarily have any prior knowledge of medical image processing or the imaging modalities used therein.

2.1 Nuclear Imaging Modalities

This section briefly explains the two nuclear imaging techniques used in this report. As the main purpose of this project was image processing, the physical background information will be kept short. In Nuclear Imaging patient scans generate physiological information about the scanned area, in contrast to CT or MR scans which show anatomical information. This physiological information can be used to determine how much blood flow a certain tissue receives or how high the metabolism rate is, as examples.

The two imaging techniques described below differ both in their physical properties and the type of physiological information they gather. As a result they are often used in combination in order to gather complimentary information about the region of interest in the patient. Although they both work upon different physical properties, they do share some common characteristics.

In both imaging modalities the patient is initially administered a substance containing a radioactive isotope called a radiotracer or just tracer for short. The radioactive isotope used in the tracer is integrated into a biologically active molecule that accumulates and in some cases is metabolized in specific parts of the body. The isotope has a known rate and method of decay, so that the areas of concentration can be detected using specialized equipment that is designed to detect the particular type of radiation. Depending upon the type of substance used, different kinds of physiological information can be observed.

The main difference between the two nuclear imaging modalities lies in the types of radiotracers that can be used with them. Each can detect different forms of radiation, and as a result of this have to be used with different radiotracers. This is due to the obvious limitation on the types of substances that can be safely administered to a patient, and the limited ability to integrate radioactive isotopes into such substances. Depending upon the physiological information that needs to be assessed, the patient may have to undergo a scan in both modalities, as will be discussed in subsection 2.2. The following subsections briefly describe the physical characteristics of both nuclear imaging modalities used in this report.

2.1.1 Positron Emission Tomography

In PET, after the patient has been administered the tracer, the patient is placed in a detector ring, with the region of interest centered under the middle point of the ring [21]. The

detector ring contains special radiation detectors called scintillators, which are materials that emit visible light when they absorb high energy particles, such as gamma rays. Scintillators are used in combination with photon transducers to generate electric signals, that in turn are processed by a computer.

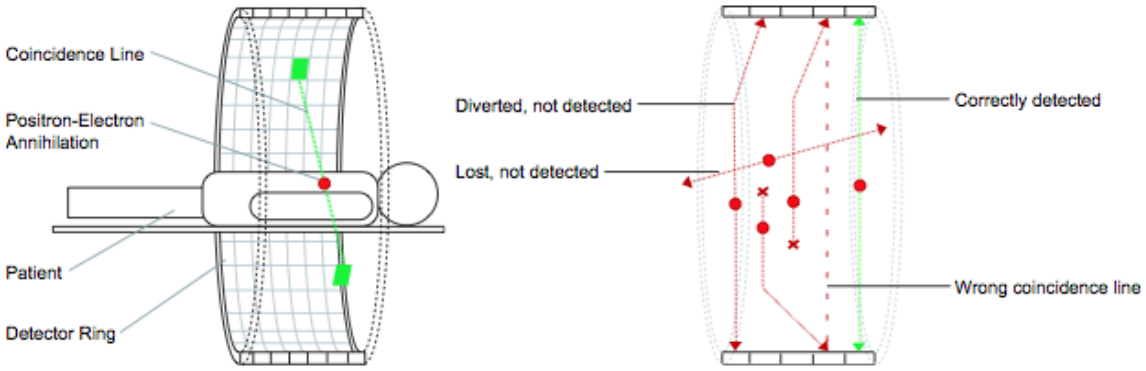


Figure 2.1: The basic principles behind PET imaging. The green lines represent correct coincident events, and the red lines represent ignored and undetected events. Because PET relies on the detection of coincident events at specific energy thresholds, it is more robust against scatter [19].

The radiotracers being used in PET imaging all decay emitting a subatomic particle known as a positron. A very short period after being emitted, a positron will annihilate itself with a nearby electron. This event causes two photons to be sent out at 180 degrees to one another with an energy of 511 keV, which can in turn be registered by the detector ring, see figure 2.1. If two photons are detected by the ring within an extremely short period of one another, usually on the order of a couple of nanoseconds, and have a high enough energy threshold, a so called coincidence event is recorded. Otherwise when a single photon is detected without a paired photon within the alloted time frame, or if the energy of one of the photons is under a certain threshold, such as 300 keV, then a coincidence is not assumed. The single photon event is still recorded, as this information is used to aid in the reconstruction of the tomographic data.

A coincidence event implies that a positron annihilation occurred along the line between the two points of the ring, referred to as the line of response. When enough coincidence events have been recorded, a set of tomographic data can be reconstructed using linear equations for all intersections of the lines of response, or if the photon detection is is fast enough then using timing information for the individual photons in a coincidence event.

One widely used radiotracer in PET imaging is ^{18}F-Fluoro-Deoxyglucose (^{18}F-FDG), which is a chemical analog to glucose. This molecule is treated by the body as it were glucose and as such is transported to parts of the body that metabolize glucose, like muscle or brain tissue. Depending upon how much is being metabolized different tissue parts will have varying concentrations of ^{18}F-FDG, which can be used to show the presence of tumors, or more important for this report show muscle metabolism rates, for example of the heart.

2.1.2 Single Photon Emission Computed Tomography

Single Photon Emission Computed Tomography, or SPECT for short, is an imaging modality that uses a similar idea as PET, but is based on a different physical principle and can detect different forms of radiation. Information about SPECT was taken from [14], [8], and [5]. The radiotracers used in SPECT emit gamma radiation, which are also detected using scintillators as in PET imaging. In SPECT imaging, each radiotracer doesn't have to emit radiation with the same energy level, in a contrast with PET imaging.

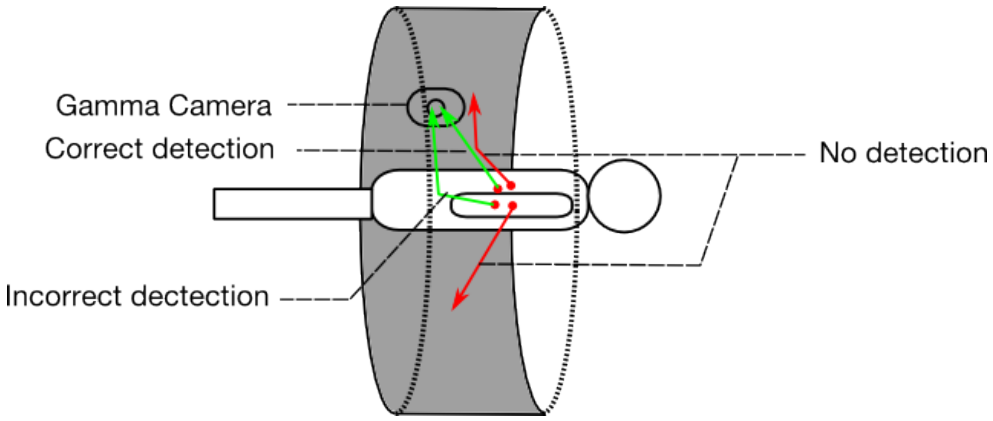


Figure 2.2: The detection scheme used in SPECT imaging. Here the gamma camera is rotated around patient, taking 2-D projects every few degrees. Due to the single point of capture, SPECT is not as robust against scatter as other nuclear imaging modalities.

Instead of registering the activity from the radiotracer using a ring of detectors, in SPECT a gamma camera consisting of a singular or multiple heads is rotated around the patient taking $2D$ images, also called projections, every few degrees. These $2D$ projections are generated by measuring the amount of gamma ray radiation detected from each angle. After enough projections have been acquired, in myocardial imaging this is typically projections over 180 degrees, they can be combined together and processed with a tomographic reconstruction algorithm to generate a $3D$ data set.

Some typical radiotracers used in SPECT imaging include ^{201}Tl and ^{99m}Tc. Both of these tracers have been shown to distribute themselves throughout the myocardium tissue of a patient proportional to the blood flow of the tissue. This makes them well suited as a diagnostic tool for the field of cardiology, which will be discussed in the next subsections. An advantage of SPECT over PET is that these radiotracers have half lives that are an order of a magnitude longer than typical radiotracers used in PET, meaning they don't need to be produced in the immediate vicinity of the scanner. Also SPECT systems tend to be less expensive than comparable PET systems. However, due to the technical limitations of the detection scheme SPECT offers lower resolution images and is more susceptible to scatter than PET.

2.2 Nuclear Cardiology

This section briefly describes certain areas of nuclear cardiology, the subfield of cardiology that deals with using nuclear imaging modalities for diagnostic purposes. Nuclear cardiology has proven itself as a highly reliable and essential aid in assessing the function and state of a patient's myocardium. Using the two nuclear imaging modalities discussed in the previous subsection, clinicians are able to diagnose with a high specificity a number of myocardial problems, chief among them being ischemic heart disease. Further upon detection of ischemic heart disease, or after infarction, clinicians are able to decide upon the appropriate strategy for treating the problem, while avoiding any unnecessary risks. This is accomplished by using the two applications of nuclear imaging described in the following subsections. The information for the next two sections can be found in greater detail in [13], [2].

2.2.1 Myocardial Perfusion Imaging

Myocardial perfusion imaging refers to a diagnostic method in nuclear cardiology that is used to detect ischemic heart disease. Ischemic heart disease is a cardiac disease where certain areas of the heart receive reduced blood flow. This is often due to coronary artery disease, which can cause symptoms ranging from diminished exercise endurance to sever chest pain, and can lead to infarction.

The diagnosis of ischemic heart disease usually involves imaging the blood flow, or perfusion of the main heart tissue, the myocardium, using a nuclear imaging modality under conditions of rest and stress. The rest and stress images are then examined to determine if a deficiency in the perfusion of any parts of the myocardium exist. This can often require that the rest and stress images are correctly aligned with each other, so that difference in perfusion between the two can be correctly assessed.

Myocardial Perfusion can be assessed using images generated by either PET or SPECT. Radiotracers typically used for perfusion imaging with PET are ^{15}O-H_2O, ^{13}N-NH_3, or ^{82}Rb. In SPECT radiotracers using either of the isotopes ^{201}Tl or ^{99m}Tc can be used. All of these radiotracers have the shared property that they distribute themselves throughout the patient's myocardium proportional to blood flow of the tissue. Any deficit in the circulation to a part of the myocardium will be reflected in the resulting image.

Current clinical protocol can vary a bit in the rest and stress testing methodology. Generally the rest scan is taken while the patient is in a normal resting state. The stress image is then taken after the patient has been put in a stress state, either by performing physical activity, or after having been given a stress inducing chemical. The order in which the scans are taken is not necessarily everywhere the same, and the time frame between the two can also be as short as one directly after the other or as long as several days between the two.

2.2.2 Myocardial Viability

In patients that have suffered an infarction of the myocardium, it is of extreme importance to determine the state of the tissue that was affected. Some areas of that are affected by a

residual ischemia may not be infarcted tissue, but instead be in a dormant state that is capable of being revived into fully functional myocardial tissue, if the circulation is restored. Because of the risks involved in revascularization of myocardial tissue, it is extremely important to determine if the revascularization will be affecting hibernating myocardial tissue or infarcted tissue. The process for assessing this is known as myocardial viability.

There are methods for assessing myocardial viability using both SPECT and PET, or a combination thereof. In SPECT an imaging protocol known as redistribution can be used, but for this report only PET or PET / SPECT combined imaging protocols were used, and so this is not discussed in further detail. The most common PET protocol involves taking a perfusion image using ^{13}N-NH_3 and another image using ^{18}F-FDG which shows the metabolism rate of the myocardial tissue. Areas with significantly less ^{13}N-NH_3 and ^{18}F-FDG uptake predominantly represent infarcted regions, whereas areas with reduced ^{13}N-NH_3 uptake but preserved or increased ^{18}F-FDG uptake represent regions of hibernating tissue. Another possibility is to use a ^{99m}Tc SPECT scan instead of the ^{13}N-NH_3 PET scan for the perfusion image. In either configuration, it is necessary that the two images are correctly aligned to each other, so that the clinician can correctly asses the difference between corresponding areas.

2.3 Medical Image Registration

Image registration is a subfield of image processing that deals with calculating the correct alignment of image pairs. Figure 2.3 illustrates the general idea behind medical image registration. The methods used to achieve this can be quite varying and are highly dependent upon the information present in the images. There are, however, some basic characteristics that are present in all image registration techniques.

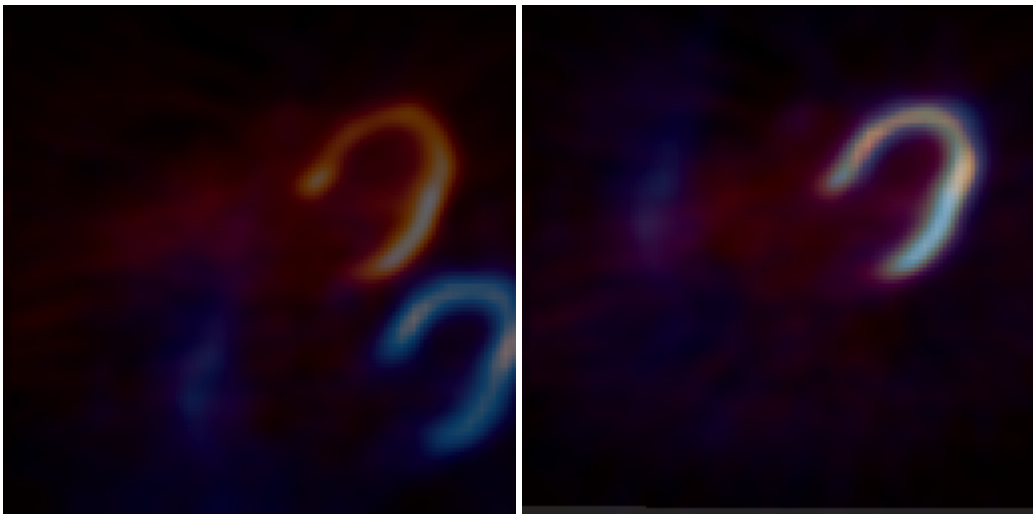
The input to the registration algorithm is two images, one image known as the fixed image F, and one image known as the moving image M, that will be mapped into the fixed image's space. In general we can define the images as mappings of points within a field of view of the patient, the domain Ω, to intensity values

$$F : x_F \in \Omega_F \mapsto F(x_F)$$
$$M : x_M \in \Omega_M \mapsto M(x_M)$$

Furthermore, because we will be dealing with PET and SPECT images, which are based upon the reconstruction of tomographic data from measured physical events, we can further classify our input images as discrete images. A discrete image is a sampling of a continuous domain $\tilde{\Omega}$ using a discrete grid Γ with spacing $S = (S_x, S_y, S_z)$. This means that the domain of the images Ω is only defined at discrete points, evenly distributed along each axis, although the spacing between points is not necessarily equal for each axis, and empty in between points. The image values at these discrete points are referred to as pixels in two dimensional space, and voxels in three dimensional space.

The result of an image registration is a transform mapping

$$T : x_M \mapsto x_F \Leftrightarrow T(x_M) = x_F$$



(a) (b)

Figure 2.3: This figure shows two image slices taken using different modalities (PET in red and SPECT in blue) in the same field of view (a). Without any intervention the images are naturally misaligned because they were taken at different times, by different technicians, and by different scanners. However the images need to be aligned in order to accurately compare for differences, (b) shows the two images correctly aligned after registration was performed.

that maps every point x_M in the moving image M to its corresponding point x_F in the fixed image F. There are many types of transforms that can be applied, ranging from simple transforms the translate points a fixed amount along each axis, to transforms that do not preserve geometric similarity between their input and output images by warping or skewing their input. The choice of transform is entirely dependent upon the nature of the difference between the fixed and moving image's domain. The transform has to be able to reconcile this difference for the registration to be of any use.

In medical image registration there are three main classes of algorithms used for determining the correct transform [9]. The simplest class is a group of algorithms called point based methods. In point based registration methods it is assumed there is a set of known corresponding points, known as fiducial points or fiducials, present in both the fixed and moving images. The fiducial points can be determined either interactively, by selecting known anatomical landmarks, or automatically by attaching markers to the patient that uniquely identifiable in the image. Generally the transform can be inferred from the corresponding fiducial points after accounting for fiducial localization and registration errors. Assuming that a rigid transform is used to map the points, the problem of finding the transform even has a closed form solution [7]. However, due to the constraints of this project, point based methods were not applied and are not discussed in any further detail.

Another approach to medical image registration is a class of algorithms called surface based methods. Instead of using singular landmark points for determining the alignment, in surface based methods corresponding surfaces from both images, such as the skin air

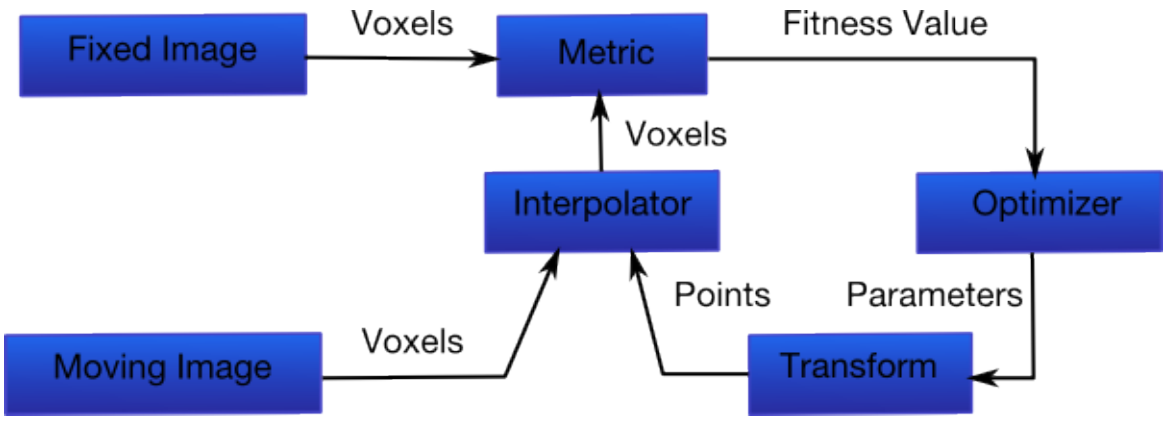


Figure 2.4: The main components used in intensity based registration, together with their place in the general registration workflow.

border), are aligned to compute the transform. The surfaces can be represented as simple point sets, or more advanced structures such as faceted or metric surfaces (e.g. a B-spline surface). The main difference between point based and surface based methods is the lack of exact point correspondence information in surface based methods. It is due to this missing correspondence information that surface based methods use iterative search for calculating the transform. Examples of surface based methods can be found in [17] and [1]. Because the main characteristic surface in cardiac PET and SPECT images, the myocardium, is not guaranteed to be always geometrically similar from the same patient between the two images, surface based methods were not used and are not further elaborated upon.

2.3.1 Intensity Based Registration

The third class of registration algorithms are intensity based methods. In this class of algorithms, the intensity of value of each voxel in the images are used to determine correspondence information. Intensity based methods are well suited for both inter- and intra-modality registration cases.

The basic structure of any algorithm performing intensity based registration is illustrated in figure 2.4. The core of intensity based registration can be broken down into four main pieces: the transform, the interpolator, the optimizer and the similarity measure, also known as the metric. Just like with surface based methods, intensity based registration also relies upon iterative search to the find the correct transform. The algorithmic flow of the registration process can be summed up as follows: beginning with a set of initial transform parameters, the optimizer maps the moving image to several new positions in the fixed image using new transform parameters, each of which is in someway derived from current parameters. At each new position the similarity measure is then evaluated. The similarity measure determines how well the images correlate and returns a scalar value proportional to their correlation. The optimizer then uses these values calculated by the similarity measure to determine the next position to advance to, where the process is repeated. The process is ended once the optimizer has reached one of its stop criteria, this can be after a set number of iterations have elapsed, or when the optimizer determines it

has reached its optimum.

As mentioned earlier, for this project we will be working only with discrete images, which adds a challenge for performing the transform correctly. Because the domain for both images is only defined at certain points, the transform may map points from the moving image to locations in the fixed image that are undefined. To accommodate this an interpolator is needed, that estimates the image values at the points that lie outside of the grid. There are several different types of interpolators, each increasing in quality and computational complexity, which are detailed in section 4.2.1.

The key component of intensity based registration is the similarity measure. Although there are quite a few different kinds of metrics, the choice of metric is directly dependent upon what types of images need to be registered. The main characteristic of any similarity measure is its ability to generate a scalar value from voxel intensity values of corresponding points in the fixed and moving images, which indicates the fitness of the current pose. Some simple metrics, such as the sum of absolute differences (SAD) or the sum of squared differences (SSD) are useful when images from the same modality need to be registered. These metrics work on the assumption that the same structures will have the same absolute intensity values. This assumption however, does not hold true for images of different modalities. In this case the metric needs to be able to tolerate absolute intensity differences that cannot be eliminated by simply rescaling the intensity values. For such cases a class of metrics that utilize techniques from information theory, the most popular of which is mutual information, have proven themselves as fairly effective for inter-modality registration. This class of metrics will be discussed in detail in chapter 4.2.4.

3 Project Description

This section describes the main goals of this project and relevant previous work performed by other research groups.

3.1 Project Goals

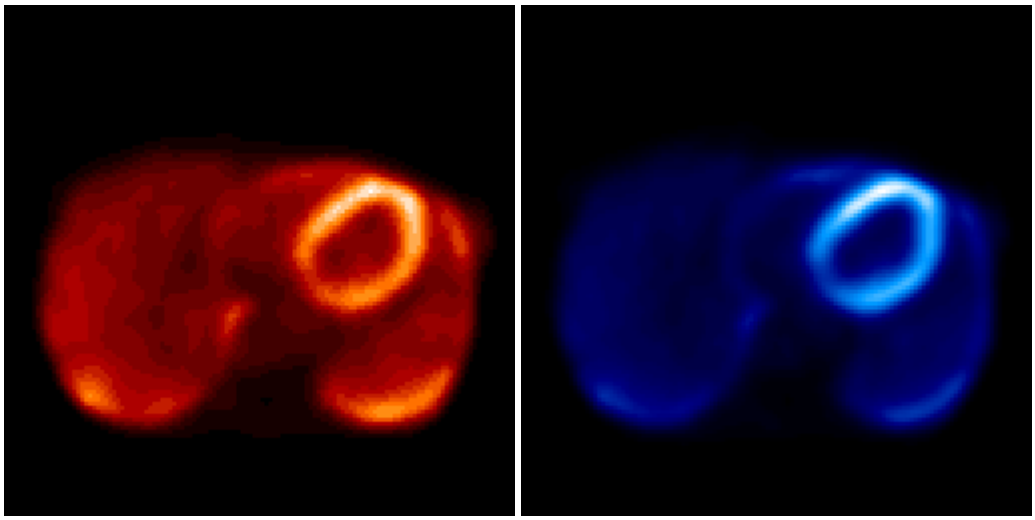
The general goal behind this project was to develop methods to assist the department of nuclear medicine at the university hospital Rechts der Isar with their automated medical image processing routine. More specifically, the idea was to develop software for the automatic registration of intrapatient cardiac PET and SPECT images. Clinicians need to register the different images in order to evaluate the physiological condition of the tissue. Automatic image processing greatly improves the quality of the information obtained by removing varying biases and adding reproducibility, when compared with manual image processing.

The concrete objective of this project was to implement a robust tool for the automatic registration of cardiac PET / SPECT images using either the Chair for Computer Aided Medical Procedures (CAMP) internal software library or using another appropriate framework. For the image registration there were four different cases that needed to be evaluated, listed below in order of their assumed difficulty:

Type 1: PET rest / PET stress, same radiotracer, no motion between scans: This is the easiest type that needed to be evaluated. In this scenario the myocardial perfusion is evaluated by having a PET scan in a state of rest followed by having another PET scan taken after a state of stress is chemically induced in the patient. In both cases the same radiotracer, ^{13}N-NH_3, is administered, so the intensity distribution between both images should be almost equal, except if the patient has a deficit in stress perfusion. This method has extremely little or no motion between scans because the patient does not leave the scanner. See figure 3.1.

Type 2: SPECT rest / SPECT stress, same radiotracer, motion between scans: This is the next most difficult type that needed to be registered. This case also deals with myocardial perfusion, this time using SPECT imaging. In this case a radiotracer containing ^{99m}Tc is administered. The difficulty presented in this case comes from the fact that there is motion between scans. Current protocol requires the patient to run on the treadmill after the rest scan, thus the two images won't necessarily contain the same field of view. See figure 3.2.

Type 3: PET ^{18}F-FDG / PET ^{13}N-NH_3, no motion between scans: This type is the second most complicated because it involves two different radiotracers. The purpose of this case is to test the patient's myocardial viability. In this scenario one rest image



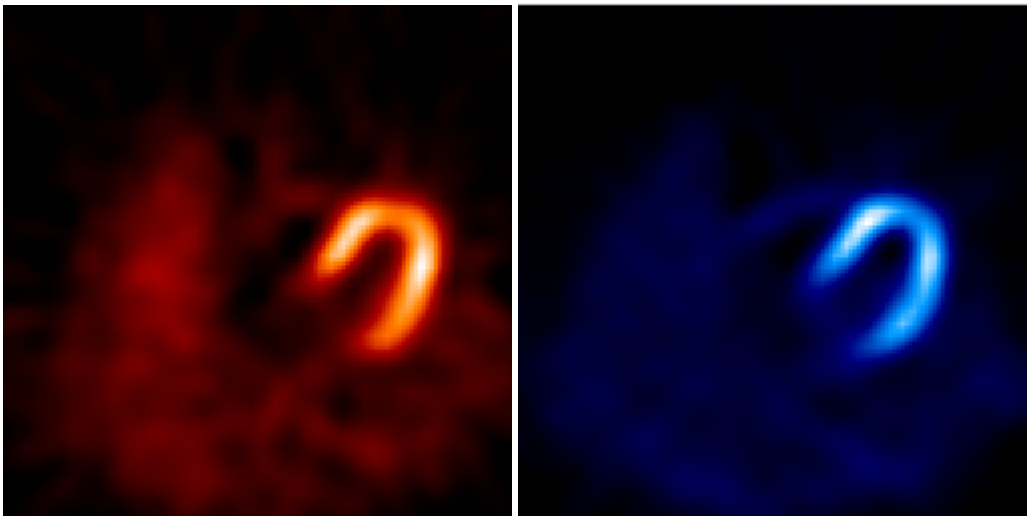
(a) (b)

Figure 3.1: Example of a type 1 registration case. Image (a) represents the rest image and image (b) the stress image. There should be little motion and little difference between the images.

is generated out using ^{13}N-NH_3 as a radiotracer, whereas another rest image uses ^{18}F-FDG. The rest image using ^{13}N-NH_3 reflects the patient's myocardial perfusion, in contrast to the other image using ^{18}F-FDG that shows the myocardial metabolism rate. Although there is no motion between the two scans, the two radiotracers reflect different physiological information, in addition to having different uptake distributions. In particular ^{13}N-NH_3 accumulates not only in myocardium but also in the liver, so both will show up strongly in an image, while ^{18}F-FDG will have uptake mostly in the myocardium. Because of this the resulting images from both scans will have very different intensity distributions and not contain 100 percent similar information. See figure 3.3.

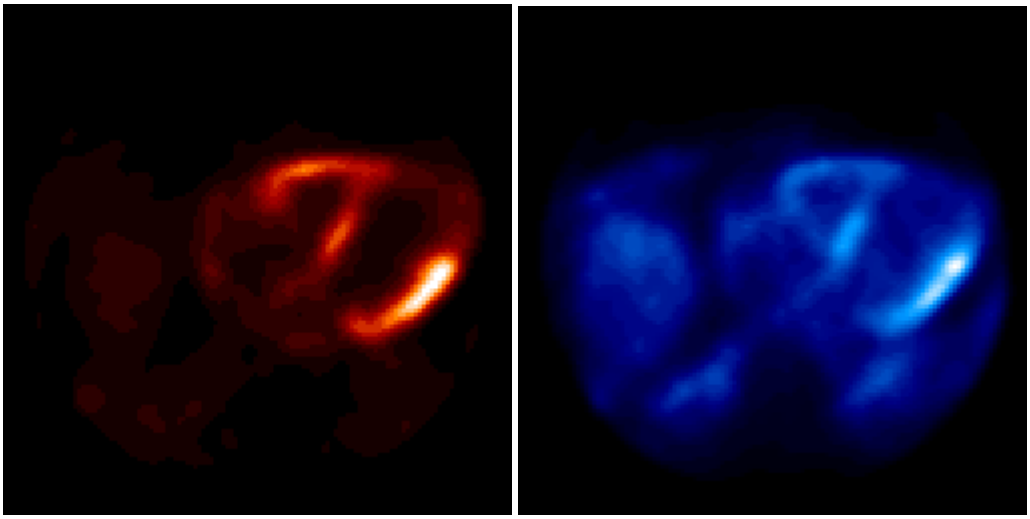
Type 4: PET ^{18}F-FDG / SPECT ^{99m}Tc, motion between scans: This is the most difficult type, not only because there is motion between the two scans, but also because the scans come from different modalities. In this configuration a rest ^{99m}Tc image is registered to a ^{18}F-FDG PET image for the assessment of myocardial viability. See figure 3.4.

It should be noted that there was no auxillary information available to aid the registration in any of the four cases, such as image markers. The program had to calculate the correct registration using only the information present in the images themselves. In order to be successfully integrated into the clinical workflow, the program had to be able to seamlessly handle all the four aforementioned cases. There also needed to be different access methods, automatic for most cases, and manual in case the registration needed to be corrected. In addition the automatic mode needed to have a live observation mode, where the current progress of the registration could be interactively monitored.



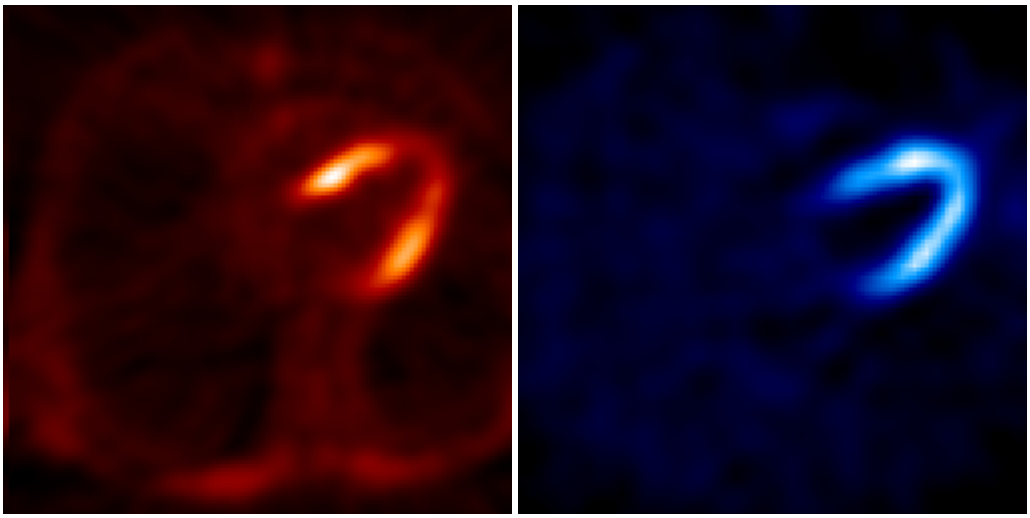
(a) (b)

Figure 3.2: Example of a type 2 registration case. Image (a) is rest SPECT image and image (b) is stress image. There will likely be motion between the image pairs, but little difference in intensity distribution.



(a) (b)

Figure 3.3: Example of a type 3 registration case. Image (a) contains the ^{18}F-FDG scan and image (b) the ^{13}N-NH_3 scan. There should be no motion between the image pairs, but there is difference in intensity distribution.



(a) (b)

Figure 3.4: Example of a type 4 registration case. Image (a) is from a PET ^{18}F-FDG scan and image (b) from a ^{99m}Tc SPECT scan. There will likely be motion between the image pairs, and there will be intensity distribution differences.

3.2 Previous Related Work

In [6] a team from the university of Milan performed a study on the feasibility of cardiac PET / SPECT registration using surface based methods. In that study the researchers registrered the transmission scans of ^{201}TI SPECT images to the transmission scans of ^{13}N-NH_3 PET images, using both phantom and patient data. The resulting registration parameters where then applied to actual PET and SPECT images. They reported an accuracy on the of 3 mm and 5 mm along the axial and transaxial planes respectively.

Another team performed a similar study for brain PET / SPECT and cardiac ^{201}Tl SPECT / SPECT images in [4]. The researchers used a rigid 3-D transformation with no rotational component and the sum of absolute differences as the similarity measure. The mean accuracy achieved was 2.1 $\pm$ 1.2mm for the cardiac studies when compared with known landmark positions in both images.

As part of a wider strategy for the automatic detection of coronary artery disease, ^{99m}Tc SPECT stress images were automatically registered to a normalized template specific for each gender in [16]. The registered image was then quantified with the template, and if the difference exceeded a threshold then the presence of CAD was assumed. The researchers stated the overall sensitivity and specificity of the automatic detection was 73% and 92% respectively.

In [20] registration of SPECT rest and SPECT stress images was performed as part of an algorithm for automatic perfusion deficit detection. The researchers registered ^{201}Tl SPECT rest images to ^{99m}Tc stress images using a rigid 3-D transform with a scaling parameter and the sum of absolute differences as the metric. The registration was deemed excellent in 177 of 204 cases, satisfactory in 24 cases, and poor in 2 cases.

4 Methods

This chapter describes the specific methods used to implement the automatic image registration, covering not only the theory behind the tools used in the registration, but also the frameworks that implemented them.

4.1 Equipment

Important for quantifying the results of any research project is to know exactly what specific equipment the project made use of. Most important for this project were the nuclear imaging devices. There were two different models used for the acquisition of the images, one PET scanner and one SPECT scanner:

PET Scanner :
Siemens ECAT HR+
Image matrix of 128x128x63
Voxel Spacing 2.3 mm x 2.3 mm x 2.43 mm

SPECT Scanner :
Siemens E.CAM
Image matrix 64x64x32
Voxel spacing 6.6 mm x 6.6 mm x 6.6 mm

There were two main computing environments that were used to carry out the development and evaluation of the registration methods:

Main Computer :
Core 2 Duo @ 2.4 Ghz
4 GB DDR2 sdram
Nvidia Geforce 8600M GT 256 MB
Apple OS X 10.5

Secondary Compuer :
PowerPC G4 @ 1.3 Ghz
1.5 GB DDR sdram
ATI Radeon Mobility 9600 32 MB
Apple OS X 10.4

Other computer configurations were used to test for reproducibility of results across all platforms, most important among were computers supporting the windows platform. Because the frameworks used to implement the automatic registration supported multiple platforms by default, special effort was made to ensure that cross platform nature was maintained.

4.1.1 Test Data Set

Just as important as the equipment used, if not more important, is the test data used to evaluate the registration. There were a total of 26 test data sets provided by the university hospital using anonymized patient data. They can be broken down into the following groups based upon the categorization in 3.1:

Case 1 2 data sets

Case 2 12 data sets

Case 3 2 data sets

Case 4 12 data sets

The data sets represent a random sample of actual clinical data, and serves as a good indicator of how well the automatic registration process would work in real clinical workflow conditions.

4.2 Image Registration

Before the specific registration frameworks are discussed, it is important to understand the theory behind the methods these frameworks offer. The basic idea behind image registration was discussed already in in section 2.3. In general the registration was limited to intensity based image registration. This has more than one reason, foremost being that no easily extractable fiducials, such as markers, are expected to be present, and the main characteristic surface of the images, the myocardium, is not expected to have exactly geometry in both images for all the registration types. It is assumed that in the case of hibernating myocardium in a type four registration case, the myocardium in one of the two input images will show diminished radiotracer uptake when compared to the other.

The following subsections will cover in detail the methods used in the major registration components.

4.2.1 Interpolator

As mentioned earlier, both the fixed and moving images have discrete domains $\tilde{\Omega}_F$ and $\tilde{\Omega}_M$ respectively. When the transform maps points from the fixed image to moving image that are undefined in $\tilde{\Omega}_M$, the interpolator estimates the value of the image at these points. There are several strategies for interpolation image values, each increasing in computational complexity and relative quality of the estimate. Simplest among them is the nearest neighbor approach, in which the intensity value of the voxel nearest is chosen. This approach is simple to implement, but lacks in precision and was undesirable for the goals of this project.

A better approach is linear interpolation, or in our case trilinear interpolation. The idea behind trilinear interpolation is to combine the intensity values of neighboring voxels proportional to their distance to the point. The intensity value of the point (x, y, z) denoted

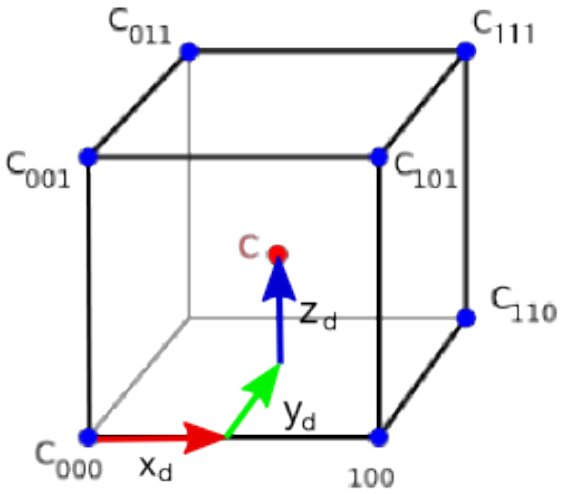


Figure 4.1: The labels of the coordinate grid for trilinear interpolation of a point located at coordinate C [25].

C_{xyz}, where x_d, y_d, z_d represent the distance between the point and the next lowest coordinate respectively, is given by the formula

$$\begin{aligned} C_{xyz} =& C_{000}(1-x_d)(1-y_d)(1-z_d)+ \\ & C_{100}x_d(1-y_d)(1-z_d) + C_{010}(1-x_d)y_d(1-z_d) + C_{001}(1-x_d)(1-y_d)z_d+ \\ & C_{101}x_d(1-y_d)z_d + C_{011}(1-x_d)y_dz_d + C_{110}x_dy_d(1-z_d)+ \\ & C_{111}x_dy_dz_d. \end{aligned}$$

This can be thought of as successively adding the voxel intensity from each of the eight neighbors proportional to their distance to the point. Figure 4.1 shows the neighbor vertex labels. A further improvement would be through the usage tricubic interpolation, which is much more computationally intensive, but does not necessarily offer better results. Thus only trilinear interpolation was used.

4.2.2 Transform

Before specific transforms are discussed, it is important to cover how the transform is applied to the moving image. The direct method maps from the moving image to fixed image, known as forward mapping. This has many problems with interpolation and voxel assignment, especially if the transform modifies the proportions of the moving image. Since we are mapping the moving image onto the fixed image's field of view, we are also only concerned with values that defined in fixed image's domain. Thus an inverse mapping is normally performed, which maps from the fixed to moving image. This principle can be seen in figure 4.2.

Since we are dealing exclusively with 3-D images, we will have to use a transform that can perform transformations in 3-D space. Of particular interest are transforms that offer both a rotation and a translation component. Because significant deformations of the

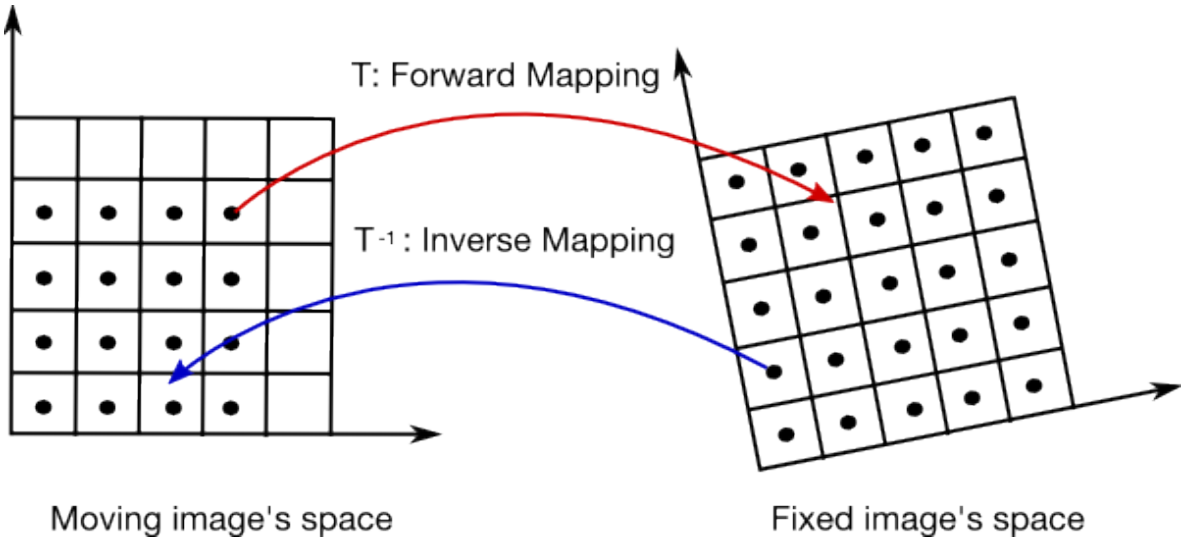


Figure 4.2: The fixed and moving images in their own discrete domains, when viewed in common reference frame. Both a forward and inverse mapping are displayed. Notice that the mapped points don't necessarily lie on the grid.

heart were not expected, and because of the complexity in evaluating the correctness of deformable transformations, it was decided to work only with rigid 3-D transformations. A rigid 3-D transform has 6 input parameters, 3 parameters for translations along the x,y and z axes, and 3 parameters for a rotation around each axis.

With the available frameworks there were only two viable options for a 3-D rigid transform, an Euler rigid 3-D transform and a Versor rigid 3-D transform. An Euler rigid 3-D transform performs a rotation around three reference axes, typically the x, y and z axes , using so called the Euler angles, and afterwards a translation along each axis. The rotational component of the transform can be represented using a 3x3 rotation matrix that is premultiplied to the input point, and the translation can be represented as vector that is added to the point afterwards. As an example, a rotation around the x, y and z axes by an angle of ϕ, θ, and ψ respectively, is represented by the orthogonal matrix

$$[R] = \begin{pmatrix} \cos\theta\cos\psi & \cos\psi\sin\phi\sin\theta - \cos\phi\sin\psi & \cos\phi\cos\psi\sin\theta + \sin\phi\sin\psi \\ \cos\theta\sin\psi & \cos\phi\cos\psi + \sin\phi\sin\theta\sin\psi & \cos\phi\sin\theta\sin\psi - \cos\psi\sin\phi \\ -\sin\theta & \cos\theta\sin\phi & \cos\phi\cos\theta \end{pmatrix}$$

which can be easily premultiplied with a point in column vector form.

Euler angles are a fairly intuitive concept to work with, nonetheless they include a number of undesirable traits for image registration, such as gimble lock. The condition known as gimble lock is used to describe the situation where a degree of freedom is lost when certain combinations of angles are chosen. Also Euler angles do not uniquely identify an orientation in three dimensional space, more than one set of angles can describe the same orientation.

A Versor rigid 3-D transform is similar to Euler rigid 3-D transform, except that a versor is used to represent the rotational component. A versor is the rotational part of a quater-

nion, which can also be classified as a unit quaternion [10], [11]. Versors are a more convenient method of representing orientations and rotations in three dimensional space. Generally they can be though of as defining a rotational axis in three dimensional space with an accompanying rotation around that axis. They do not suffer from gimbal lock, while at the same time they are more numerically stable for interpolating between rotations when compared with Euler angles. Combinations of rotations using versors is significantly faster than using rotational matrixes.

The translational component of the transform can be represented by column vector. Versors are described using four parameters: q_0, q_x, q_y, q_z. The components q_x, q_y, and q_z are used to describe an axis in space, and q_0 to describe a rotation around that axis. The orthogonal rotation matrix of quaternion is represented by

$$[R] = \begin{pmatrix} q_0^2 + q_x^2 - q_y^2 - q_z^2 & 2q_xq_y - 2q_0q_z & 2q_xq_z + 2q_0q_y \\ 2q_xq_y + 2q_0q_z & q_0^2 - q_x^2 - q_y^2 - q_z^2 & 2q_yq_z - 2q_0q_x \\ 2q_xq_z - 2q_0q_y & 2q_yq_z + 2q_0q_x & q_0^2 - q_x^2 - q_y^2 + q_z^2 \end{pmatrix}$$

and because a versor is unit quaternion with the additional property

$$q_0^2 + q_x^1 + q_y^2 + q_z^2 = 1$$

the matrix entries can be rewritten in the more computationally friendly form

$$[R] = \begin{pmatrix} 1 - 2(q_y^2 + q_z^2) & 2(q_xq_y - q_0q_z) & 2(q_xq_z + q_0q_y) \\ 2(q_xq_y + q_0q_z) & 1 - 2(q_x^2 + q_z^2) & 2(q_yq_z - q_0q_x) \\ 2(q_xq_z - q_0q_y) & 2(q_yq_z + q_0q_x) & 1 - 2(q_x^2 + q_y^2) \end{pmatrix}.$$

The conversion from versor components to Euler angles can be accomplished with the equation

$$\begin{pmatrix} \phi \\ \theta \\ \psi \end{pmatrix} = \begin{pmatrix} \arctan\frac{2(q_0q_x + q_yq_z)}{1-2(q_x^2+q_y^2)} \\ \arcsin(2(q_0q_y - q_zq_x)) \\ \arctan\frac{2(q_0q_z + q_xq_y)}{1-2(q_y^2+q_z^2)} \end{pmatrix}.$$

Because the advantages of versors over Euler angles, especially for automatic image registration, the Versor rigid 3-D transform was chosen as the standard transform.

4.2.3 Optimizer

The optimizer is one of the key components of the registartion process that has a large impact on the success of the registration. Under the assumption that the similarity measure has its global optimum when the two images are perfectly registered, the ideal registration would simply search the entire parameter space of the transform. Obviously this is not feasible for several reasons, alone the computation time would greatly exceed any reasonable tolerance. Also the assumption that the similarity measure has its global optimum when the two images are perfectly registered does not necessarily hold true for the image pairs used in this report, due to their different intensity distributions. Instead a reduced assumption is accepted, where the similarity measure has a local optimum when the images are perfectly registered.

The optimizer is responsible for performing an intelligent search of a subset of the transform's parameter space. Starting with a set of initial transform parameters, the optimizer evaluates several new parameter sets, each located a distance from the current parameter set called the step size. The optimizer then advances to the next most optimal set, repeating the process iteratively until a local optimum has been reached. Although there are a few different types of optimizers, the choice of optimizer is determined by the transform type and the image types. The appropriate optimizer for working with a versor rigid 3-D transform is a versor rigid 3-D transform optimizer.

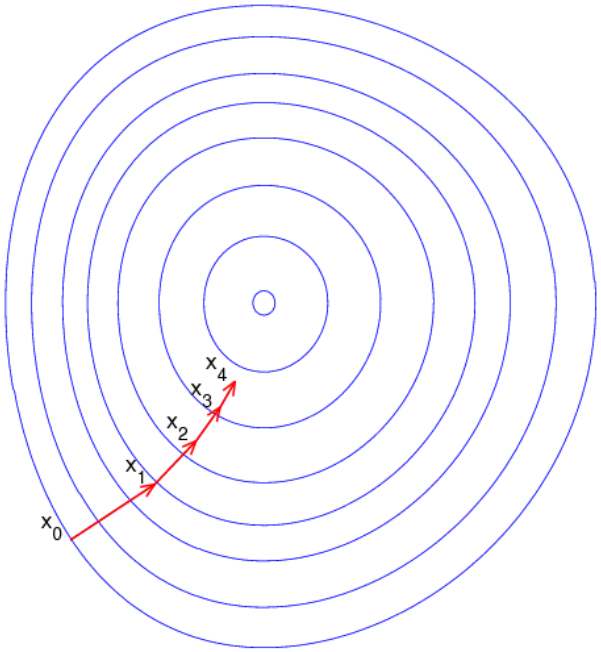


Figure 4.3: Illustration of the process of gradient descent optimization [23].

A versor rigid 3-D transform optimizer is a variant of a regular step gradient descent optimizer that has been modified to work with the versor component of the transform. Regular step gradient descent optimizers take step sizes proportional to the gradient of the transform at the current parameters. This concept is illustrated in figure 4.3. Versors in particular do not form a vector space, due to the fact that versor multiplication is not commutative. The versor rigid 3-D transfrom optimizer differentiates itself from a regular step gradient descent optimizer in that it aware of the special properties of the versor rotational component.

4.2.4 Similary Measure

The real core of the registration process is the similarity measure, also referred to as the metric. The success of the registration hinges on the ability of the similarity measure to have a local optimum when the images are correctly registered. Some simple metrics, such as the sum of absolute differences (SAD)

$$SAD = \frac{1}{N} \sum_{x \in \Omega_F^T} |F(x) - M^T(x)| \quad \text{where} \quad N = |\Omega_F^T|$$

or sum of squared differences (SSD)

$$SSD = \frac{1}{N} \sum_{x \in \Omega_F^T} |F(x) - M^T(x)|^2 \quad \text{where} \quad N = |\Omega_F^T|$$

are useful when images from the same modality need to be registered. These similarity measures are well suited for certain cases of intramodality registration, but are not able to adequately deal with intermodality registration cases. Currently the best adapted similarity measures are based upon a property from information theory known as mutual information.

Mutual information operates on the idea of maximizing information (intensity value) that is shared between images, as discussed in [18]. The basic principle is based on random variables. The entropy of a random variable A is given by the following equation

$$H(A) = -\int p_A(a) \log p_A(a) da$$

and the joint entropy of two random variables A and B is given by

$$H(A, B)) = \int p_{AB}(a, b) \log p_{AB}(a, b) da db.$$

If the random variables are independent from one another then

$$H(A, B) = H(A) + H(B).$$

However, if they share information then

$$H(A, B) < H(A) + H(B)$$

and the difference is know as the mutual information

$$I(A, B) = H(A) + H(B) - H(A, B).$$

Applied to image registration, in mutual information each image would be viewed as a random variable. A similarity measure based upon mutual information will give a measure of how much randomness (unique information) is present in the combined registered image compared to the two input images. Mutual information increases as the images become more aligned because there is less randomness in the registered image. One unique advantange of mutual information is that it not dependent upon absolute pixel value correlations between the input images. This is important because the intensity differences between a PET image using ^{18}F-FDG and a SPECT image using ^{99m}Tc cannot be compensated for using simple intensity rescaling.

There were two main implementations of mutual information evaluated for this project, each of which is explained below.

Viola-Wells Mutual Information

The first implementation of mutual information that was tested was based upon the algorithm from Viola and Wells [22]. Typically the marginal and joint probabilities densities cannot be calculated directly and must be estimated. Instead they can be estimated using kernel density estimators, or Parzen windows [3]. This is accomplished by taking intensity samples from the image S and superimposing centered kernel functions $K()$ on them. Thus the density can be estimated with

$$p(a) \approx P^*(a) = \frac{1}{N} \sum_{s_j \in S} \log K(a - s_j).$$

Often the gaussian density function is used as the kernel function in the Parzen window estimate, although this is not a requirement, any differentiable function could be used. In addition to the Parzen window estimate, the viola-wells implementation also takes a second sample set R and calculates the mean image entropy using

$$H(A) = \frac{1}{N} \sum_{r_j \in R} \log P^*(r_j).$$

The mean entropy is calculated once for each image, and then reevaluated after every new transform position for the combined image. The Viola-Wells implementation also requires that the images be normalized so that they have a mean of zero and a standard deviation of one. The calculation of the mean entropy estimate is computationally expensive, as it involves an N x N loop.

Mattes Mutual Information

Mattes mutual information is an alternative implementation for the mutual information metric proposed by Mattes et al. [15]. In Mattes mutual information only one set of spatial samples is taken per image. These samples are then used to calculate the marginal and joint probability density function at discrete intervals, known as bins, that are spread out over the entire dynamic range of the image.

Let L_F and L_M represent a specified number of uniformly sized bins along their respective dimensions of the joint histogram. Two integer values are used to index the bins, l and k, with $0 \leq k \leq L_F$ and $0 \leq l \leq L_M$. Let $\beta^{(3)}$ represent a cubic spline Parzen window and $\beta(0)$ represent a zero order Parzen window, where each satifies the partition of unity constrain. Then the smoothed joint histogram is given by

$$p(l, k|T) = \alpha \sum_{x \in \Omega_F^T} \beta^{(0)} \left(k - \frac{F(x) - \lfloor F \rfloor}{\Delta b_F} \right) \times \beta^{(3)} \left(l - \frac{M^T(x) - \lfloor M \rfloor}{\Delta b_M} \right)$$

where α is a normalization factor so that the joint histogram normalizes to 1, $\lfloor F \rfloor$ and $\lfloor M \rfloor$ are the minimum intensity values of the fixed and moving images respectively, and Δb_F and Δ_M are the intensity range of each individual bin necessary to fit the number bins into the intensity ranges of the fixed and moving images. The marginal entropy of the moving image is can be taken from the joint histogram by

$$P_M(l) = \sum_k p(l,k)$$

The marginal probability density can be calculated independently of the current transform parameters by

$$P_F(k) = \alpha \sum_{x \in \Omega_F^T} \beta^{(0)} \left(k - \frac{F(x) - \lfloor F \rfloor}{\Delta b_F} \right)$$

This implementation is computationally cheaper than the implementation by Viola and Wells. Because each sample only affects a small number of bins, the probability density function does not need a N x N loop to be computed. In practice the Mattes et al. implementation is significantly faster than the Viola-Wells implementation, while at the same delivering the same quality results.

4.3 Frameworks

The next sections cover the frameworks that were investigated during this project. The main application developed for performing the automatic registration, PetSpectFusion, was created using both the ITK and Qt frameworks, and was deployed in the clinical workflow upon completion of the project. More information on this can be found in appendix A.

4.3.1 CAMP Library

The initial approach for this project was to investigate how easily the Chair for Computer Aided Medical Procedures' (CAMP) internal software library would allow the implementation of a tool for the four registration cases. To begin, an existing software program that used the CAMP libraries named MAPFusion was analyzed. MAPFusion was designed as GUI for using and observing various registration methods available in the library.

The program, as well as the CAMP library, is written using C++, with Qt or FLTK for graphical components. The program offered various options for the registration, the most important of which was the choice of cost function between mutual information, normalized cross coefficient, and sum of squared difference. There also was the ability to adjust the initial transformation applied to the moving image.

One large deficit in MAPFusion was the complete lack of automation. The goal of this project was to create an automatic registration program that could be scripted from the command line, as well from a GUI. Another disadvantage of the CAMP library was the lack of portability. The CAMP library uses several microsoft specific C++ extensions, and any application that makes use of it cannot be easily ported to other platforms. As work progressed on creating a custom program, further problems were encountered when working with images of different resolutions, which will be explained in more detail in section 5.1.

4.3.2 ITK Framework

The alternative to rewriting MAPFusion and possibly parts of the CAMP library was to implement the required functionality using the Insight Segmentation and Registration framework (ITK). ITK is an open source cross platform framework for developing segmentation and registration applications written in C++, sponsored by the nation library of medicine and developed by Kitware Inc.. ITK has several advantage over the CAMP library in that it is cross platform, offers more control over the registration process, has far more image processing tools, and most importantly, is able to handle many image sizes and formats. On the advice of my supervisor, it was decided to implement the automatic registration program using ITK. Information about ITK was taken from [12].

ITK presented some unique initial difficulty due to its cross platform nature. ITK and all programs that link to it have to use CMake, the cross platform make file generator. CMake reads in a user created configuration file and generates build files appropriate for the platform it is running on, on Mac OS X these are Xcode project files and on windows its visual studio projects. This process is controlled by a project specific file, `CMakeLists.txt`, that contains the information about what files are used by the project, what libraries to link with, and what kind of executable to generate. A short example is presented in listing 4.1.

Listing 4.1: An example of a simple CMakeLists.txt file

```
PROJECT(PetSpectFusion)

FIND_PACKAGE(ITK REQUIRED)
IF(ITK_FOUND)
        INCLUDE(${ITK_USE_FILE})
ENDIF(ITK_FOUND)

FIND_PACKAGE(Qt4)
IF(QT_FOUND)
        INCLUDE(${QT_USE_FILE})
ENDIF(QT_FOUND)

#if we are compiling on osx, generate an application bundle
IF(APPLE)
        SET( MACOSX_BUNDLE_ICON_FILE PetSpectFusion.icns )
        SET( CMAKE_OSX_ARCHITECTURES x86_64;i386;ppc)
        SET( MACOSX_BUNDLE_COPYRIGHT Brian Jensen 2008-2009)
ENDIF(APPLE)

CMAKE_MINIMUM_REQUIRED( VERSION 2.4.0 )

SET(SRCS src/Main.cxx src/ImageIO.cxx src/RegObserver.cxx
        src/Registration.cxx src/PetSpectFusionViewer.cxx)

SET(INCS inc/typedefs.h inc/ImageIO.h inc/RegObserver.h)

ADD_EXECUTABLE(${PROGNAME} MACOSX_BUNDLE
        ${SRCS} ${MOC_SRCS} ${INCS} )

TARGET_LINK_LIBRARIES(PetSpectFusion ITKCommon
```

```
        ITKIO ITKAlgorithms ${QT_LIBRARIES})
```

Since ITK is cross platform by nature, it was decided with the new implementation to preserve this behavior as much as possible. A new program was created, PetSpectFusion, to begin exploring the ITK framework's possibilities. Most of the development was carried out in an OS X environment, but windows support was always maintained as this was the target platform for deployment at the university hospital.

ITK itself is a framework strictly for medical image processing, because of this it offers no utilities for creating a GUI to visibly observe the registration process or verify the results. It does, however, allow a callback function to be hooked into the optimizer, so that the optimizer executes the callback function every time another step was taken. This was used to create a rudimentary method of tracking the path the optimizer chose, but it was somewhat insufficient for gaining any real detailed insight into the registration process. To better evaluate the registration process a GUI was necessary, which is described in 4.3.3

General Implementation

Implementation of a simple registration program using ITK was fairly straight forward. ITK provides functions for loading images of various formats, although in this project strictly meta header image files were used. ITK supports multiple voxel types for image data and for internal representation. This is accomplished through the heavy usage of C++ templates. Generally, every single ITK object has to be casted to the appropriate type using a `typedef` statement, including the image objects, the transform, the interpolator, and various others. This is illustrated in listing 4.2, which shows some of the key types for the registration.

Listing 4.2: Shows the definitions for some of the key components in the registration

```
//setup internal image types
typedef float InternalPixelType;
const unsigned int Dimension = 3;
typedef itk::Image<InternalPixelType, Dimension> InternalImageType;

//now set up the types for the registration
typedef itk::VersorRigid3DTransform< double > TransformType;
typedef itk::VersorRigid3DTransformOptimizer OptimizerType;
typedef itk::ImageRegistrationMethod<InternalImageType,
        InternalImageType> RegistrationType;
typedef OptimizerType::ScalesType OptimizerScalesType;
```

Although the choice of the voxel type for input images is dictated by their internal data type, in most cases after the images are loaded, they are cast to an internal image type for the remainder of the registration process. Both `float` and `double` were found to have adequate accuracy, and in most cases `float` was used.

Another special property of the ITK framework is that it implements the smart pointer design pattern for most objects. Smart pointers simulate real pointers while tacking on additional features such as automatic garbage collection through reference counting. The advantage of this method is that the programmer doesn't have to worry about memory leaks. Every time a new object is created, instead of receiving a pointer to the object,

the programmer receives a reference to a smart pointer object that encapsulates the desired object. The initial reference count is set to 1. Every time the reference to the smart pointer is duplicated, the object count is incremented. Once an object containing a reference to the smart pointer object goes out of scope, the reference count of the target object is decremented. When the reference count reaches zero, the actual object is also deallocated. Despite the lack of a pointer to the actual object, it can accessed directly using the smart pointer as if it were a pointer to the actual object.

Registration Initialization

The first step in performing a registration is to create the necessary objects for each of its components. Listing 4.3 shows the main components.

Listing 4.3: Shows the creation of the main registration components

```
MetricType::Pointer       metric      = MetricType::New();
OptimizerType::Pointer    optimizer   = OptimizerType::New();
InterpolatorType::Pointer  interpolator = InterpolatorType::New();
RegistrationType::Pointer  registration = RegistrationType::New();
TransformType::Pointer     transform    = TransformType::New();

//add the components to the registration object
registration->SetMetric(metric);
registration->SetOptimizer(optimizer);
registration->SetInterpolator(interpolator);
registration->SetTransform( transform );
```

The next step is loading the two input images. This is the accomplished with the use of the class `ImageFileReader`. This class reads in the input image and casts it to the internal image type used in the registration. Listing 4.4 shows the necessary code for loading an image.

Listing 4.4: This code reads in an input image and casts it to the internal image type

```
typedef itk::ImageFileReader< InternalImageType > MhdImageReaderType;
MhdImageReaderType::Pointer mhdReader = MhdImageReaderType::New();

mhdReader->SetFileName(fileName);
try
{
        mhdReader->Update();
}
catch(itk::ExceptionObject & err)
{
        std::cerr << "Error reading the image from the disk!" << std::endl;
        return -1;
}
InternalImageType::Pointer image = mhdReader->GetOutput();
```

The last general step in the registration initialization is calculating the initial transform parameters. The initial transform used to start the registration has a large influence on the success of the registration. In ITK one class responsible for initializing a transform

is `CenteredTransformInitializer`. This class takes the center of the moving image and aligns it with the center of the fixed image, where the center can be the geometric center or the center of mass. For this project only the geometric center was taken because the SPECT images using ^{99m}Tc show heavy uptake in the liver and the heart, whereas the PET images using ^{18}F-FDG only show heavy uptake in the heart, causing the initializer to pick an disadvantageous initial pose.

Optimizer

As mentioned earlier in this chapter, the `VersorRigid3DTransformOptimizer` was chosen as the apporpriate optimizer. The behavior of the optimizer can be controlled using several parameters including the minimum and maximum step sizes, the maximum number of iterations, and a scaling parameter for each degree of freedom, which are used to weight each one differently. The scaling parameter can be used to restrict the optimizer from taking steps in one of its degrees of freedom. Listing 4.5 shows the optimizer's control parameters being set.

Listing 4.5: Shows the initialization of the optimizer

```
//create the optimizer scaling weights
optimizerScales[0] = DEFAULT_XROTATION_SCALE;
optimizerScales[1] = DEFAULT_YROTATION_SCALE;
optimizerScales[2] = DEFAULT_ZROTATION_SCALE;
optimizerScales[3] = DEFAULT_XTRANSLATION_SCALE;
optimizerScales[4] = DEFAULT_YTRANSLATION_SCALE;
optimizerScales[5] = DEFAULT_ZTRANSLATION_SCALE;

//set the optimizer's parameters
optimizer->SetScales(optimizerScales);
optimizer->SetMaximumStepLength(maxStepSize);
optimizer->SetMinimumStepLength(minStepSize);
optimizer->SetNumberOfIterations(maxIterations);
optimizer->AddObserver( itk::IterationEvent(), observer );
```

The optimizer maintains a list of observers, that it calls after each iteration. These observers can be used to perform actions or track the registration process. For this project the observer was used to update the GUI and to print the current transform and metric values to the console.

The last required component in order to start the registration is the metric. Because both Viola-Wells and Mattes' implementation differ in some key points, they are presented separately.

Viola-Wells Mutual Information

There are several parameters that govern the ITK implementation of Viola-Wells mutual information. The first is the standard deviation being used for calculating the gaussian density kernel of the two images. Another parameter required is the spatial sampling rate. This parameter is heavily dependent on the nature of the registration being performed, as well as how much noise is present in the images. In combination with the required

parameters it is also often necessary to use an image normalizer that will scale the images to that they have a zero mean and a unit variance. Another useful step for increasing registration accuracy is to preprocess the images with a gaussian blur.

Listing 4.6: Shows the initialization of Viola-Wells implementation of Mutual Information

```
//set the metric's parameters
metric->SetFixedImageStandardDeviation( 0.4 );
metric->SetMovingImageStandardDeviation( 0.4 );

//set the smoother's parameters
fixedSmoother->SetVariance( 2.0 );
movingSmoother->SetVariance( 2.0 );

//read in images and normalize them
fixedNormalizer->SetInput( fixedImageReader->GetOutput() );
movingNormalizer->SetInput( movingImageReader->GetOutput() );
//smooth the images
fixedSmoother->SetInput( fixedNormalizer->GetOutput() );
movingSmoother->SetInput( movingNormalizer->GetOutput() );
//add them to the registration process
registration->SetFixedImage( fixedSmoother->GetOutput() );
registration->SetMovingImage( movingSmoother->GetOutput() );

//set the absolue number of samples, where 0 < sampleRate < 1
const unsigned int numberOfVoxels = fixedImageRegion.GetNumberOfPixels();
const unsigned int numberOfSamples =
        static_cast< unsigned int >(sampleRate * numberOfPixels);
metric->SetNumberOfSpatialSamples(numberOfSamples);

//now start the registration ....
```

Listing 4.6 shows the steps necessary for initializing the Viola-Wells implementation. The individual steps involved in the initialization are be self explanatory.

Mattes Mutual Information

The configuration of Mattes mutual information is very similar to that of viola-wells, although a quite bit simpler. As well as setting the spatial sampling rate used for both images, the number of bins needs to be specified, that are used for calculating the entropy. This value has just as much an impact on the quality of the registration as the number of samples. There is no no need to blur the images, and there is no need to renormalize the images.

Listing 4.7: Shows the initialization of Mattes' implementation of Mutual Information

```
//set the number of bins
metric->SetNumberOfHistogramBins(bins);

//read in images and add them to the registration
registration->SetFixedImage( fixedImageReader->GetOutput() );
registration->SetMovingImage( movingImageReader->GetOutput() );
```

```
//set the absolue number of samples, where 0 < sampleRate < 1
const unsigned int numberOfVoxels = fixedImageRegion.GetNumberOfPixels();
const unsigned int numberOfSamples =
        static_cast< unsigned int >(sampleRate * numberOfPixels);
metric->SetNumberOfSpatialSamples(numberOfSamples);

//now start the registration ....
```

Multi-resolution Registration

Multi-resolution registration is the process of performing the image registration with images resampled to lower resolutions before the main registration is performed. The results of each registration is used as the starting point for the next higher resolution registration, starting with the lowest resolution. The idea is that as the registration is performed with images at lower resolutions the overall registration time will decrease while the stability should increase, because the best initial transform for the main registration should the results of a previous registration.

The implementation in the ITK framework allows the use of image pyramids for performing multi-resolution registration. The image pyramids can contain arbitrary resampling values for every dimension of each image. The class responsible for carrying out the multi-resolution is `MultiResolutionImageRegistrationMethod`.

4.3.3 Qt Framework

There were several possibilities for adding a GUI to PetSpectFusion. One possibility was a companion framework to ITK, the Visualization Toolkit (VTK), that offers libraries for creating 2D and 3D image views, but does not offer any sort of widgets for allowing user input or controlling the the program. Qt was picked as the best choice for creating a GUI on top of the ITK base program. Qt is a full featured graphics toolkit that is also cross platform and integrates nicely with the CMake build environment of ITK.

The basic approach was to create GUI similar to that present in MAPFusion. PetSpectFusion needed to have the ability to easily load the fixed and moving images, to change the transform and registration parameters, and most importantly to view the registration in action. The first step was to create a view panel for viewing the images in the same field of view.

A three slice view model containing the transversal, coronal, and sagittal planes was chosen for this purpose. The implementation of the view panel presented several difficulties because of the difference between how ITK and Qt deal with image data. ITK offers methods for resampling image data into a different voxel space and then extracting user specified regions of interest from it. The first problem encountered was the lack of default color lookup table for single channel images in Qt. A color lookup table is mapping of image intensity values to RGB triplets, so that the image can be displayed on a computer monitor in a purposeful way. The principle behind color lookup tables is shown in figure 4.4. Each color table used had to be implemented by hand.

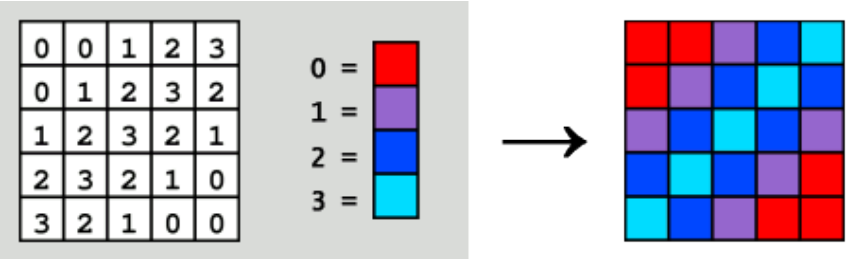


Figure 4.4: An example of a color lookup table for an image. Each of the intensitvy values present in the image is mapped to specific color value, giving the image a different appearance when displayed [24].

Although the choice of color lookup table to use with each image is arbitrary, some color lookup tables are better suited for working with nuclear cardiac images. Initially linear color lookup tables were used for the fixed and moving images, which can be seen in figure 4.5. Linear color lookup tables make it difficult to discern individual image features when both images are displayed in the same field of view, mainly because they distribute the color evenly amongst each image's intensity values. Better suited are gradient based color lookup tables that distribute color unevenly amongst the image's intensity values. An example of this can also be seen in figure 4.5.

Another problem was encountered with Qt's handling of 8 bit single channel images. Most of Qt's standard image classes expect input data to be 24 to 32 bit (3 - 4 channels respectively), so none of Qt's internal image processing functions could be directly used without modification. Alpha blending, the process of fusing two images together, had to be implemented by hand as an example. This was accomplished with the formula

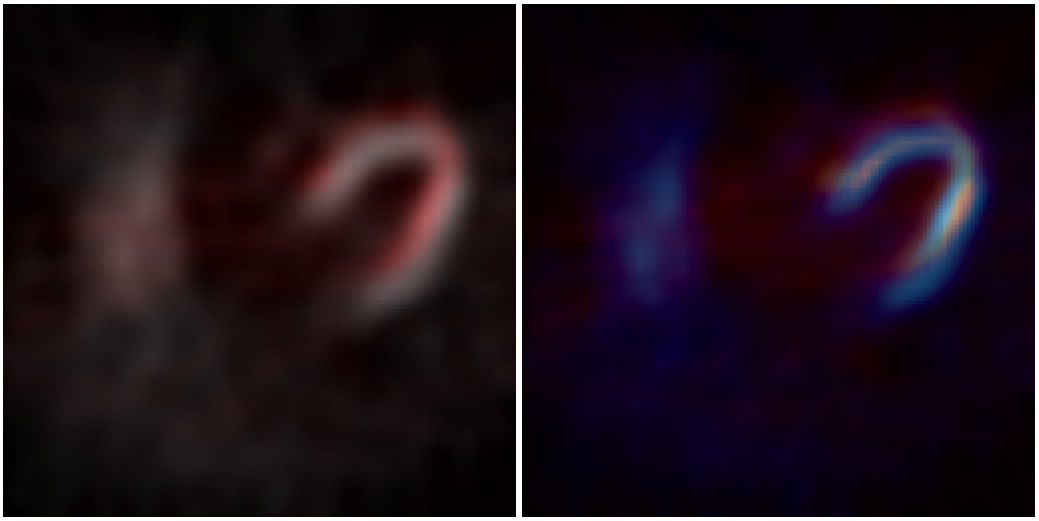
$$C = \alpha A + (1 - \alpha)B, \alpha \in [0, 1]$$

where A is the input pixel value for the fixed image, B is the pixel value for the moving image, α is the blend value, and C is the result value.

As an alternative method of verifying the registration quality, another viewing mode was created using checkerboard composite images. This is different from the alpha blended viewing mode in that each image is divided up into 4 x 4 x 4 sections. These sections are then used to create composite image, where sections are alternatingly taken from the fixed and moving images. An example of this is presented in figure 4.6.

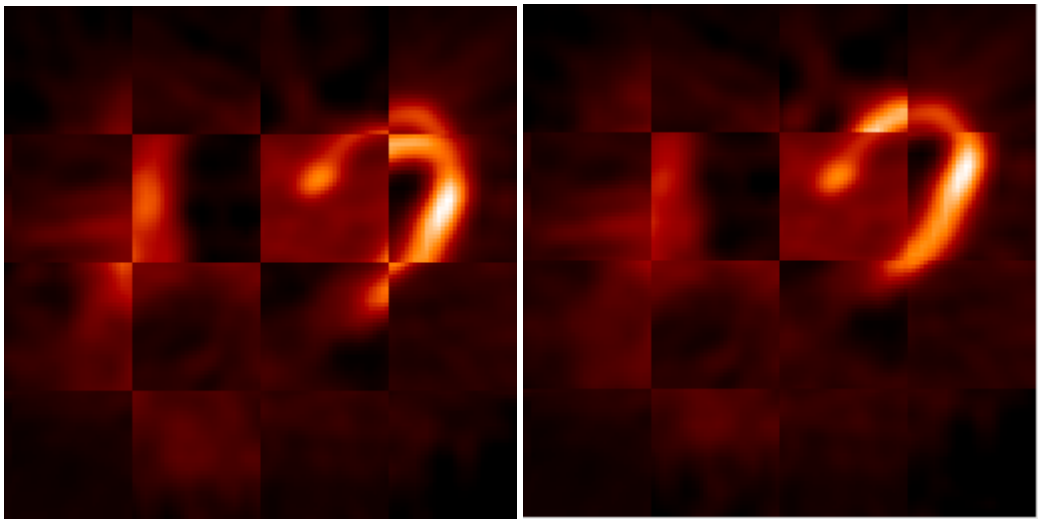
On top of being able to view alpha blended image slices, PetSpectFusion also needed to be able to manually apply transforms to the moving image. This has two reasons, the first being that many different initial poses needed to be tested. One of the most important parameters for the registration is the initial transform. It was crucial to the success of the project to be able to test the robustness of the registration against many different initial transforms. The other reason was the necessity to manually follow the changes in the metric value for a given transform. This was necessary for understanding the steps taken by the optimizer, especially when the optimizer takes a path that leads to a mis-registration.

For more information about PetSpectFusion and its features see section A.



(a) (b)

Figure 4.5: Images using linear color lookup tables (a), and using gradient based lookup tables (b). Notice that the slight mis-registration is easier to identify using gradient based based color lookup tables.



(a) (b)

Figure 4.6: Checkerboard composite images. Image (a) shows the two images with same mis-registration as in image 4.5, and (b) shows the composite when correctly registered.

5 Results

This section discusses the results of the different registration methods, and lists the parameters used to achieve them.

5.1 Evaluation of the Camp Library

Initial tests showed that the automatic registration could be successful for the first and third registration types. There were four images pairs tested, two for the first registration type and two for the second. The tests were successful for both cases when the mutual information and normalized cross coefficient metrics were used, but were unsuccessful for the third case when the sum of squared differences was used. In the successful tests the mean deviation of the generated transform parameters from the manually determined parameters was less than one millimeter. In the unsuccessful tests the registration would calculate a transform with a z axis component of 202 mm, effectively placing the two images on top of one another with no overlap.

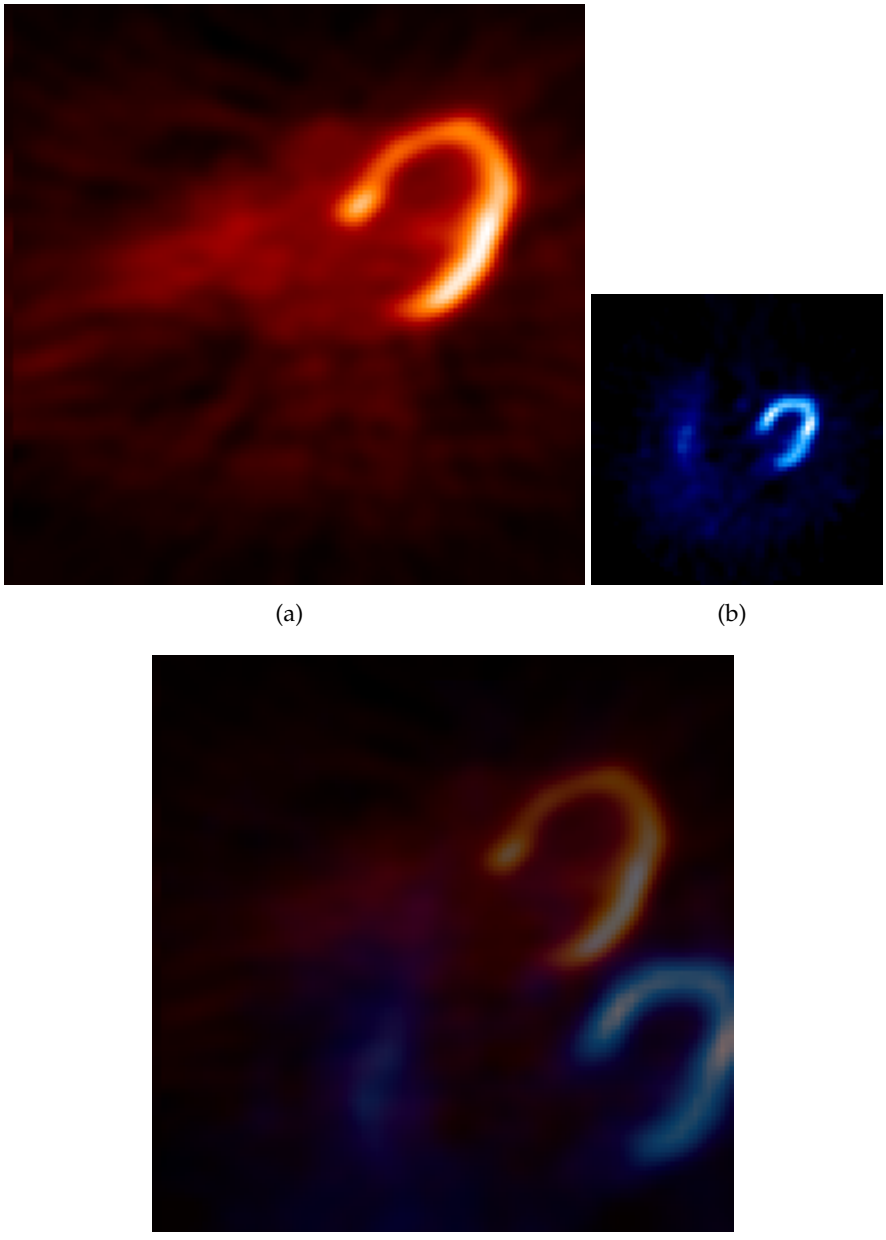
Tests with the other two cases at the time were not possible because the PET and SPECT images occupy both different voxel and physical spaces. SPECT images used in this report represent a physically larger area than the PET images, despite the fact that the PET images have almost eight times the number of voxels. The exact image matrix and spacing sizes are listed in 4.1. MAPFusion was unable to correctly handle image pairs that weren't exactly the same size in voxel space, and as result no registration could be performed. For an example of this concept see figure 5.1

Upon closer inspection, it appeared that the CAMP library at the time had always performed the registration in voxel space and not in physical space. So in essence the registration was trying to align two severely geometrically distorted images, which of course could never work. As long as the images occupied roughly the same voxel space with the same voxel spacing, everything was alright, which is why the first and third registration types were successful. It was up to the program developer to handle all the resampling of images necessary to perform the registration. At this point the development of the automatic registration tool was continued using the ITK library instead of the CAMP library, in part due the resampling problem, but mostly due to ITK's other advantages mentioned in 4.3.2

5.2 Evaluation of the ITK Framework

5.2.1 Viola-Wells Mutual Information

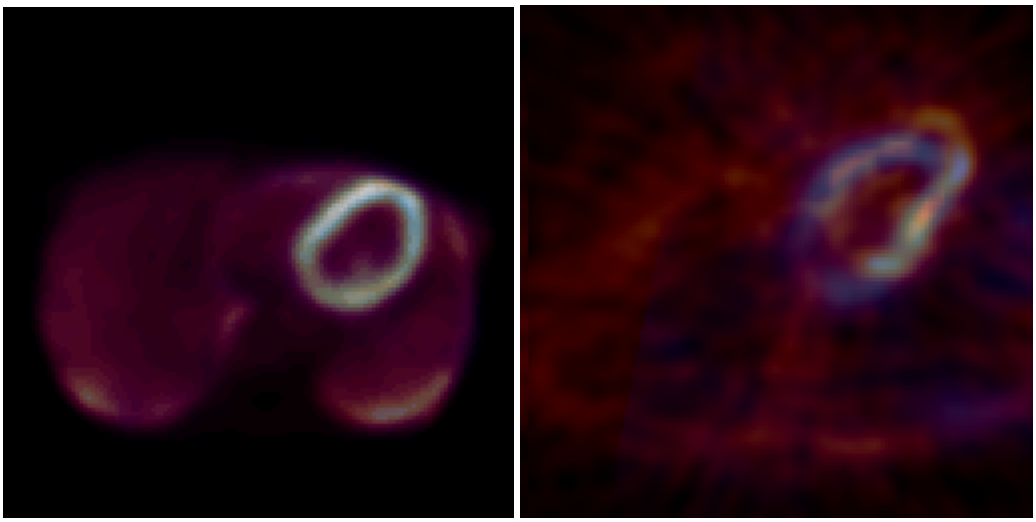
ITK recommends starting with an initial sample rate of $20 - 50\%$ of the voxels of the fixed and moving images, and then scaling down to as low as 1%. Because of the large difference



(a) (b)

(c)

Figure 5.1: This figure shows a PET image (a) and a SPECT image (b) slice from the same patient in each image's voxel space. It is clear that a registration performed solely in voxel space on these images will fail. In (c) the two images are resampled into physical space with the SPECT image placed into the fixed images field of view. Notice that the bottom and right side of the SPECT image is truncated, this is because the SPECT image occupies a larger physical space than the PET image.



(a) (b)

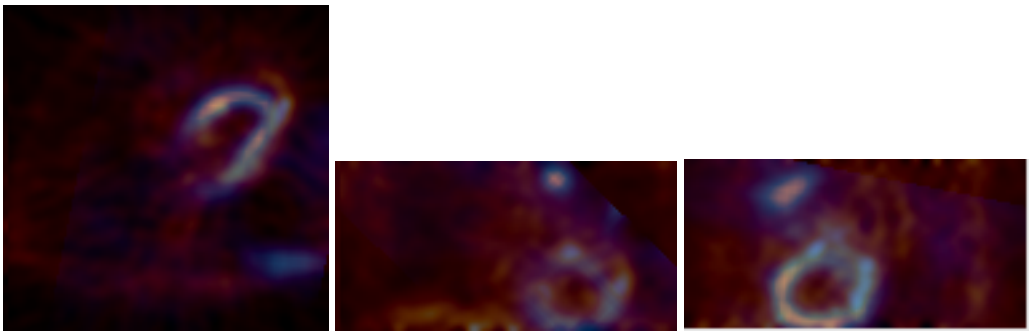
Figure 5.2: This figure shows some initial results achieved using the viola wells implementation with a sample rate of 70%, min step size of $0.01mm$ and max step size of $0.5mm$. In (a) is successful PET (FDG) / PET (FDG) registration while (b) shows a unsuccesful case four PET (in red) / SPECT (in blue) registration.

between the images generated caused the different radiotracers used in types 3 and 4, in addition to the generally noisy state of the images, it was found that a sampling rate between $40 - 70\%$ is ideal. Anything lower than that value and the registration quality will suffer, anything higher and the registration time increases considerably, recall that Viola-Wells use a N x N loop to calculate the probability density function.

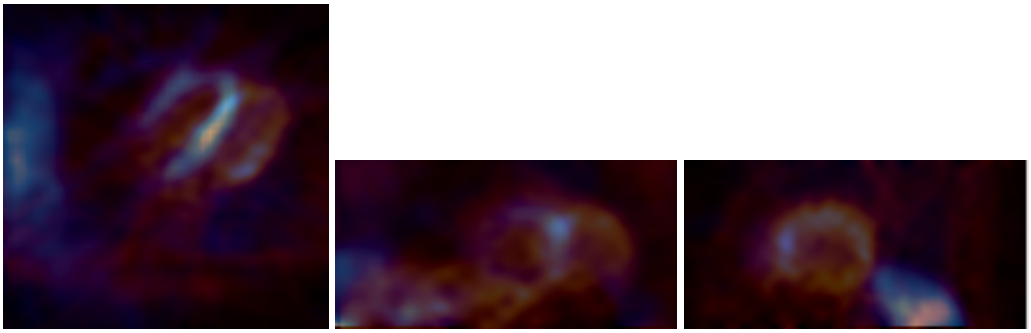
For all tests performed using Viola-Wells mutual information, a gaussian blur was applied to both images using a variance of 2.0, after they were normalized. Initial results using the recommended paramters from the ITK handbook were very positive. Tests were conducted on types 1, 3, and 4. In figure 5.2 can be seen a successful type 1 registration case, and an unsuccessful type 3 case, although the mis-registration is not very severe. In early testing very conservative values were chosen for the optimizer step sizes (min step size of $0.01mm$, max step size of $0.5mm$) as well as very restrictive optimizer scales for the rotation part of the transform ($1/100,000$ weighted in all directions).

The final results achieved using Viola-Wells' implementation indicated that type 1 and 2 cases can be successfully registered, with the mean error compared of less than one millimeter in each translation direction, when compared with manual registration. Type 4 cases, however, showed unsatisfactory results, which can be seen in figure 5.3. The mean registration error in the translation for these cases compared to manual registration was 5.1, 5.9 and $8.5mm$ in X, Y and Z direction respectively, while the mean rotation error was 0.09, 0.08 and 0.02 radian respectively.

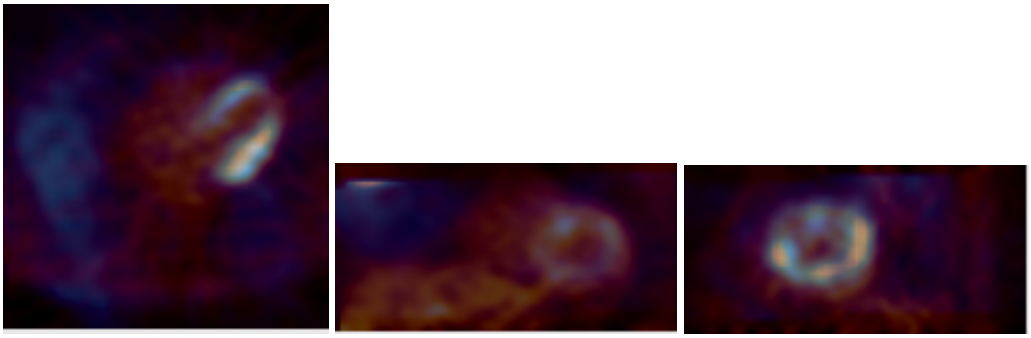
(a)



(b)



(c)



(d)

Figure 5.3: Shows the before, (a) and (c), and after registration, (b) and (d), of two type 4 cases using Viola-Wells mutual information where mis-registration occured. Each image group contains a transversal, coronal, sagittal slice respectively.

5.2.2 Mattes Mutual Information

The speed advantage of Mattes et al. implementation made itself apparent right from the beginning. Each iteration required half the time in comparison to the Viola-Wells implementation. An additional advantage of this implementation is that it has a much smoother distribition. It allows the optimizer to be more aggressive without the increased risk of landing in a false local optimum. In figure 5.5 you can see the optimizer path taken plotted against the metric value with fairly conservative values (min step size of $0.01mm$ and a max step size of $0.5mm$), while 5.5 shows the same case with more reasonable values (min step size of $0.1mm$ and a max step size of $3.0mm$)

The implementation was able to successfully register all four of the type 1 and 3 cases, just as with Viola-Wells' implementation . Of the type 2 cases tested, 11 of 12 were able to be registered successfully. Table 5.1 shows the final results achieved using Mattes' mutual information for all 12 of the type 2 cases, compared to the gold standard manually determined parameters.

Table 5.1: **Case 2 results: automatic and manual registration difference**

Case Name	**Translation**			**Rotation**			**Iterations**
	x	**y**	**z**	**x**	**y**	**z**	
AdBr	0	0	0	0	0	0	6
AlKh	0	0	0	0	0	0	7
AnBe	0	0	0	0	0	0	12
AnHe	0	0	0	0	0	0	20
AuWa	0	0	4	0	0	0	4
BaKo	0	0	0	0	0	0	6
BaRu	0	-4	0	0	0	0	10
BeHe	0	0	0	0	0	0	13
BiBa	0	0	0	0	0	0	3
ExPe	0	0	0	0	0	0	7
FeXa	0	0	0	0	0	0	13
ObSi	12	4	0	0	0	0	39
Average	**1.0**	**0.3**	**0**	**0**	**0**	**0**	**11.5**

The type 4 cases tested also had similar results, with 10 of 12 cases being registered successfully. In figure 5.6 you can see the two type 4 cases that mis-registered in addition to one of the cases from figure 5.3 that correctly registered using Mattes mutual information. Table 5.2 shows the final results of the type 4 registration cases, again compared with parameters determined through manual registration. Figure 5.4 shows the result metric value plotted together with the average deviation from the manual parameters for each case. As is evident in the plot, there seems to be no correlation between absolute metric value and mis-registration of the case. For both cases where the registration was deemed unsuccessful, the metric value was lower at the correct position than at the final mis-registered position, independent of start position. This obstacle will not be able to be overcome by simply changing the registration parameters alone, other methods will have to be investigated in order to mitigate this effect.

Table 5.2: **Case 4 results: automatic and manual registration difference**

Case Name	Translation			Rotation			Iterations
	x	y	z	x	y	z	
Fiho	0	0	2	0	0	0	20
GrPe	0	0	0	0	0	0	11
HoBa	0	0	-4	0	0	0	12
HuHe	1	-2.5	-25	0	0	0	23
KlMa	0	0	0	0	0	0	11
KrEd	0	-4	0	0	0	0	16
MaEl	0	0	0	0	0	0	14
ObSi	12	4	-14	0	0.07	0	27
ScNo	0	0	0	0	0	0	19
SoKa	0	0	0	0	0	0	18
ViFr	0	-2	0	0	0.02	0	20
WiKa	0	0	4	0	0	0	19
Average	**1.1**	**1.1**	**4.1**	**0**	**0**	**0**	**17.5**

The registration time for type 1, 2 and 3 cases spanned from 1 to 3 seconds for each case, with the average being 1.4 seconds. The type 4 cases, however, took significantly longer, although the registration time is still within a tolerable threshold. The registration times for both multithread and single threaded registration mode, as well as 64 and 32 bit modes is listed in table 5.3. For a contrast, the average manual registration time for each case was between two and three minutes, so the automatic image registration still presented a significant time savings.

Table 5.3: **Case 4 Registration times on an intel Core2 Duo @ 2.4 Ghz (seconds)**

Case Name	Multithreaded		Single threaded	
	32 Bit	64 Bit	32 Bit	64 Bit
FiHo	17	12	26	21
GrPe	10	8	17	14
HoBa	11	9	16	14
HuHe	18	14	30	25
KlMa	10	8	17	14
KrEd	14	11	22	20
MaEl	12	10	21	18
ObSi	21	18	36	31
ScNo	17	14	26	22
SoKa	15	11	22	19
ViFr	17	13	27	22
WiKa	16	13	26	22
Average	**14.8**	**11.8**	**23.8**	**20.2**

On average multithreaded mode was 65.5% faster than single threaded mode, due to the

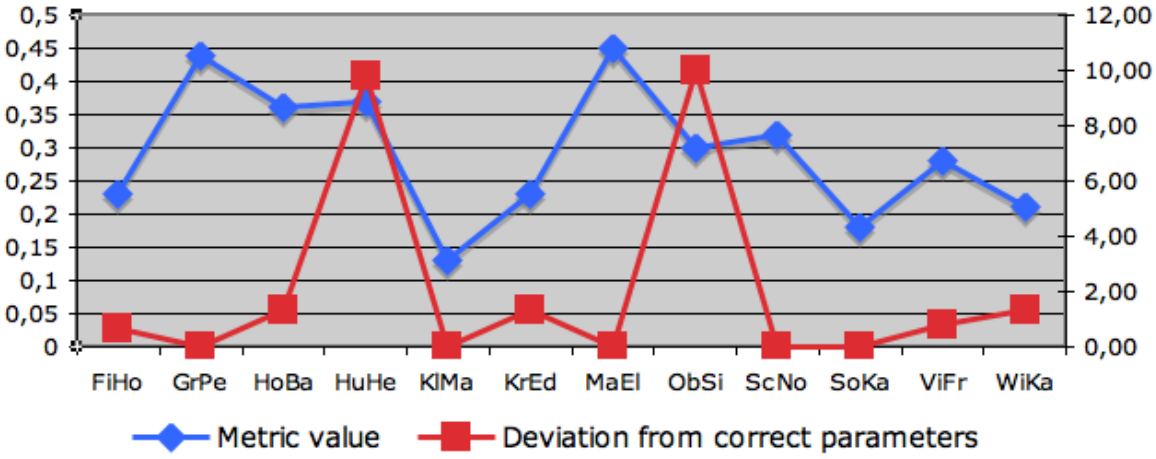


Figure 5.4: Metric value plotted against the average deviation from the correct parameters for each type 4 case.

fact that the Core2 Duo has two processing cores. Since the optimizer and the image processing functions, including transform calculation, are all multithreaded, a similar boost in performance is likely to be enjoyed by systems of up to 8 cores. Also surprising was the 25% performance increase simply by using 64 bit code. This increase was likely due to the additional general purpose and vector (SSE) registers available in x86-64 mode.

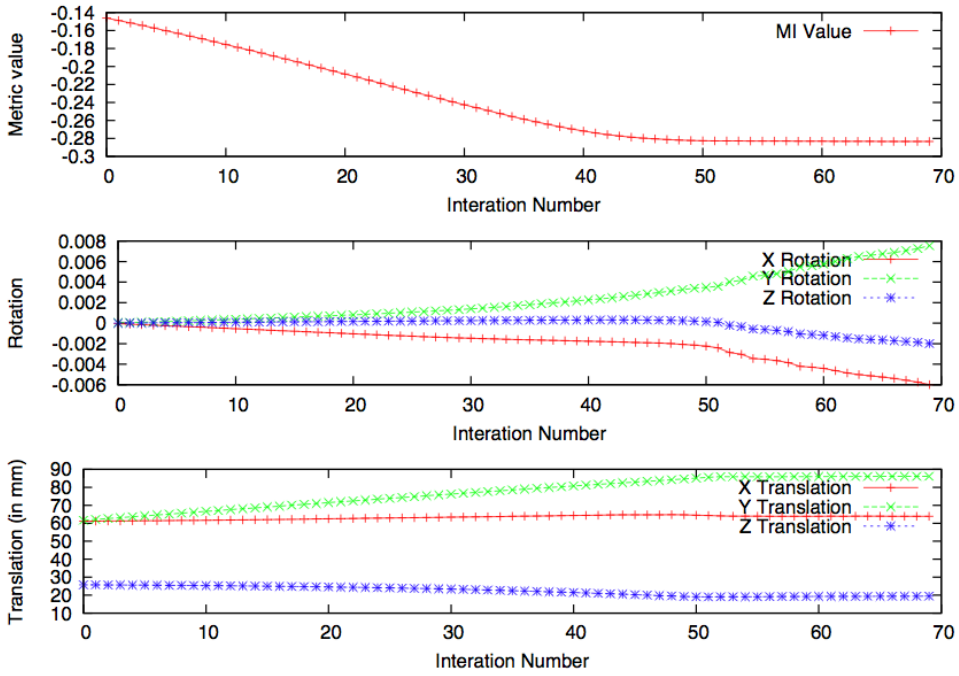
The best parameters for Mattes' mutual information were empirically determined to best match all four registration case types. Table 5.4 lists the parameters along with the standard value and an acceptable range that should still produce acceptable results.

Table 5.4: **Registrattion parameters**

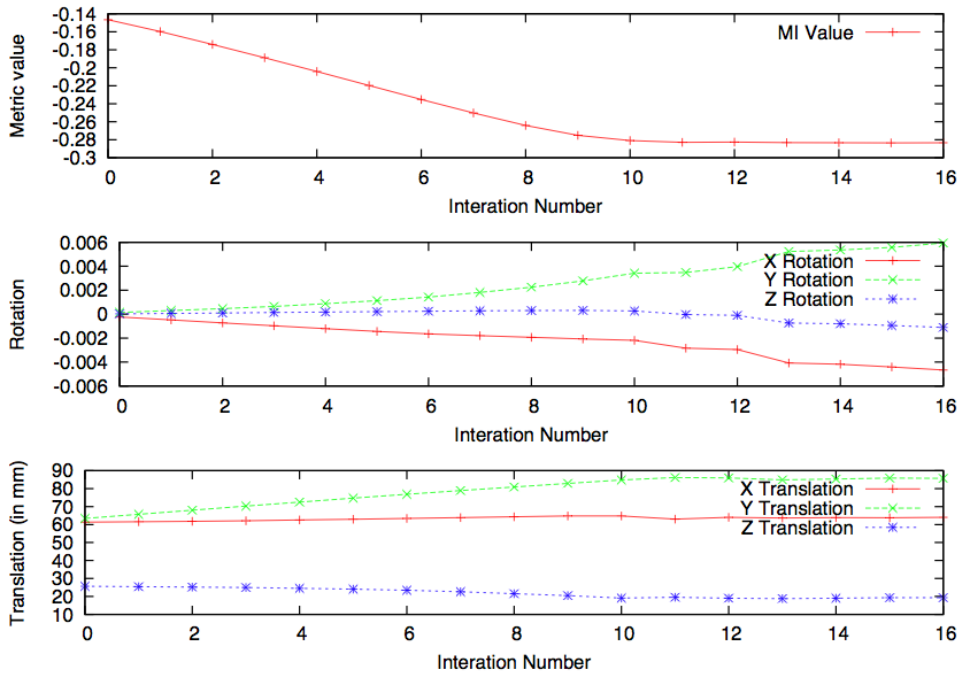
Parameter	Default Value	Acceptable Range
Intensity Bins	50	30 to 80
Max Iterations	250	250 to 300
Min Step Size	0.2 mm	0.001 mm to 1.0 mm
Max Step Size	2.3 mm	0.1 mm to 4.3 mm
Sample Rate	70%	30 to 100%
Rotation weights	1.0	0.01 to 10.0
Translation weights	0.00001	0.000001 to 0.0001

5.2.3 Multi-resolution Registration

For this project image pyramids were used with 4 levels, each one with each dimension half the size of the previous. Tests showed that multi-resolution did not decrease registration time, but increased it by an average of 30%. There was also no advantage discernible when compared to performing a regular registration, the success rate was identical with that of the normal registration for registration cases evaluated.

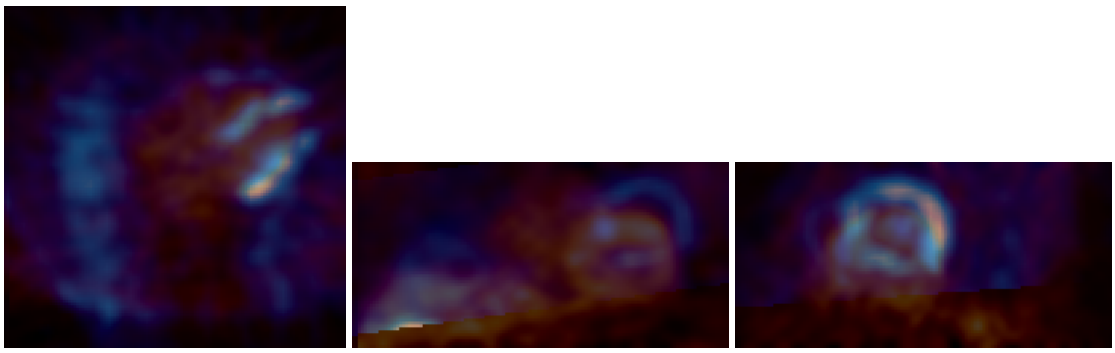


(a)

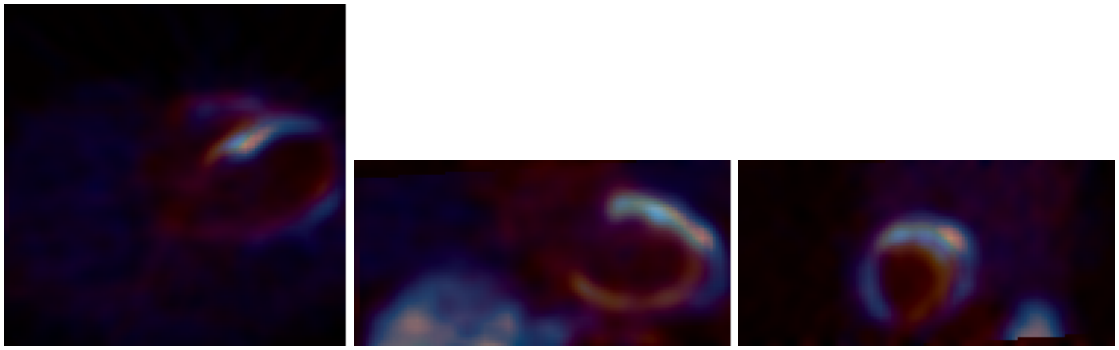


(b)

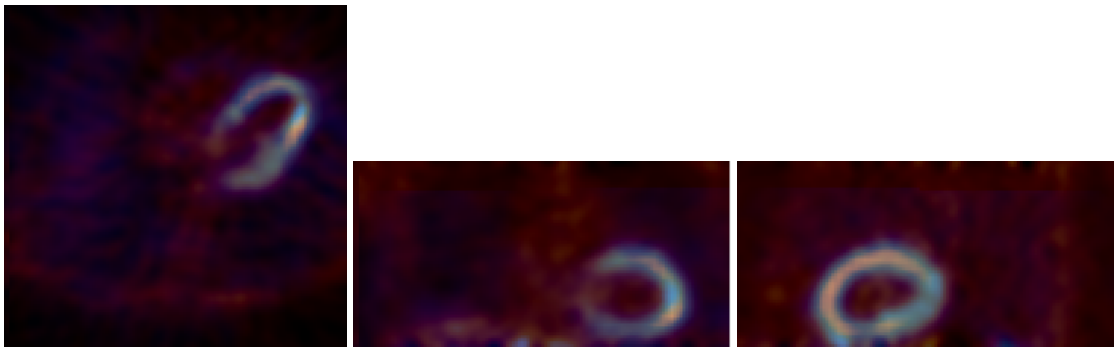
Figure 5.5: Figure (a) shows optimizer path taken using conservative step sizes with mattes mutual information, and (b) shows optimizer path taken when using larger step sizes



(a)



(b)



(c)

Figure 5.6: Shows the two type 4 cases, (a) and (b), that were mis-registered, as well as one case (c) that was previously mis-registered using viola-wells but was successful registered using Mattes implementation.

6 Conclusions

6.1 Discussion

The work performed in this project has shown the effectiveness of applying readily available open source toolkits to the field of image registration. In particular, the combination of ITK for the image processing and Qt for the user interface seems well suited for the rapid development of image registration applications. The cross platform nature of both frameworks offers the ability to write applications that are developed in one environment and deployed into another with minimal effort. With the help of these two frameworks a program was created, PetSpectFusion, that was well adapted for observing the registration process and testing the many parameters that control it.

Intensity based registration using Mattes et al. implementation of mutual information has shown itself to be the best fit for performing the four types of image registration attempted in this project. Using this implementation 100% of type one and three registration cases, 92% of type two cases, and 83% of type four registration cases were able to be successfully registered. The average registration error, when compared with manually determined registration parameters, was 1.1 ± 3.3 mm, 1.1 ± 2.5 mm, and 4.1 ± 10.6 mm for the translation in x, y, and z directions respectively, and less than 1 degree for all rotations. The registration time for each type one, two and three case had an average of 2.1 ± 1.2 seconds, and 11.8 ± 2.7 seconds for type four cases, when using the program in 64 bit multithreaded mode on a current dual core processor. Multi-resolution registration has not presented an advantage over standard registration, while costing on average 30% more time, and is not recommended for the image types used in this project.

Further analysis of the registration cases that failed to be successfully registered indicated that simply modifying the registration parameters would not suffice for eliminating the problem. Additional image preprocessing steps would be necessary to alleviate the error, such as automatically segmenting the myocardium, perhaps with the help gated images. Nonetheless, the program constructed for this project, and the methods discussed herein, are well suited to be integrated in the current clinical workflow.

6.2 Final thoughts

Fully automated image processing offers unique advantages over traditional methods that are only increasing as more sophisticated algorithms are being engineered. Automatic image registration in particular is an area with high potential in the medical field. As was seen in this project, automatic registration of nuclear cardiac images is able to be performed satisfactorily using freely available open source toolkits and current algorithms.

Most of the difficulty in implementing an automatic image registration is finding the right parameters. Parameters that work well for the one set of cases may not work well

for the second set. The choice of registration method is also highly limited by the types of images being registered. Multi-modality registration, as was the case in this project, requires more complicated algorithms and computing power than registering images of the same modality. Nonetheless it is possible to implement the automatic registration of nuclear cardiac images using current techniques and achieve a satisfactory level of quality.

The benefits of automatic registration methods to the current clinical workflow are extensive. Aside from the time savings, which should not be underestimated when dealing with highly skilled specialists, there is an increase in quality. This is mostly through reproducibility, because instead of having the registration performed by several different technicians or specialists, each with a different bias, there is now one program with a single reproducible bias. Even if the registration is performed manually, the automatic registration can be used to get a second result to be used as a comparison.

Based on the results achieved in this project using Mattes mutual information, as described in section 5.2.2, an abstract paper was published in the annual proceedings of the Society of Nuclear Medicine 2008 and an oral presentation was given at the conference.

A PetSpectFusion Reference

This chapter contains information about PetSpectFusion, a program that was written as tool too help analyze the different registration algorithms and parameters. It main function is to visualize the registration process and provide a front end to the ITK functions.

A.1 Introduction

PetSpectFusion is a tool that was written as a GUI and command line interface for the various ITK 3-D registration classes. It is written in C++ and uses the Qt framework for providing a GUI and multithreading. The main purpose of this program is to enable detailed examination of the registration process and its results, while also allowing changes to be made to the registration parameters easily. It can perform registrations fully automaticly or manually, using either the command or the GUI. In addition to this it can also perform set rigid transformations to images, in case you have many images in a series that don't need to be individually registered to each other.

As it is based upon cross platform open source tools PetSpectFusion can be run on all major desktop operating systems: Windows, Linux, and Mac OS X. Currently the program has only been tested with images in meta header format, but other other image formats like the analyze format should also be usable. Although it is designed for the registration of PET and SPECT data sets, it is theorectically possible to use any data sets that can be registered using a 3-D rigid transform.

A.2 Installation Requirements

The only requirement for installing and running PetSpectFusion is a working installing of Qt 4.4 or later, except on windows platforms as the installer package contains the necessary Qt libraries. As a result of this you can install this program on most current desktop operating systems. You can download the latest version of the Qt framework for your OS from http://www.qtsoftware.com/downloads/opensource/appdev. This program was tested on Windows (x86 and x64), Mac OS X, and Ubuntu Linux x86_64. Because of this, there are custom installer packages available for all three of these platforms that can be downloaded from the following location: http://home.in.tum.de/ jensen/downloads/. In order to install the program on a different platform, such as Linux x86, you will need to compile it from source, for this see the section compiling from source.

A.3 Usage Modes

PetSpectFusion supports various modes of operation which allow fully automatic, semi-automatic and completely manual registartion. Manual registration mode starts the GUI

but does not automatically start the registration process or exit once the registration process is completed, allowing the user to modify the results or rerun the registration with different parameters if necessary. Semi-automatic mode starts the GUI and automatically begins registration process but does not exit automatically, allowing the user to thoroughly verify the results. Fully automatic mode starts the registration on program begin and exits once the registration is complete. This mode can be run headlessly, displaying only progress information on the command line, or with the GUI showing progress, in case the registration process needs to be obverved.

In addition to the modes mentioned above another mode of operation exists, transform mode, which simply performs a rigid 3-D transform on an input image and saves the result. This mode can be used when there are many images in series that do not need to be individually registered, for example in a gated study.

A.4 Command Line Reference

A.4.1 General

The command line is used for starting the majority of the programs usage modes. The simplest mode, manual registration mode, is started simply by starting the program without passing any arguments. This will start the GUI with the default arguments set.

The next mode, semi-automatic mode, is started by passing PetSpectFusion the following command line parameters:

```
PetSpectFusion -g -f fixed image -m moving image -o output image
  [-l logfile name] [-b bins] [-s sample rate] [-k min step size]
  [-l max step size] [-i max iterations] [-p plotfile name]
  [-r Multi-resoltion levels] [-t transform parameters]
  [-tf transform file] [-z optimizer scales]
```

Important to note about the command syntax used above is that optional parameters are contained in square brackets, they are used to change the default values of the registration settings, all others parameters are mandatory. The command listed above starts the PetSpectFusion GUI, loads the fixed and moving images, sets all the parameters specified and automatically starts the registration and automatically saves the result to the file specified. The big difference to the fully automatic mode is that the program does not automatically close, it waits until the user explicitly requests that the program exit.

This next command will start PetSpectFusion in fully automatic mode, but will run headless, that is without the GUI.

```
PetSpectFusion -f fixed image -m moving image -o output image
  [-l logfile name] [-b bins] [-s sample rate] [-k min step size]
  [-l max step size] [-i max iterations] [-p plotfile name]
  [-r Multi-resoltion levels] [-t transform parameters]
  [-tf transform file] [-z optimizer scales]
```

To start PetSpectFusion in fully automatic mode while displaying the registration in GUI you have use the next command.

```
PetSpectFusion  -a -g -f fixed image -m moving image -o output image
  [-l logfile name] [-b bins] [-s sample rate] [-k min step size]
  [-l max step size] [-i max iterations] [-p plotfile name]
  [-r Multi-resoltion levels] [-t transform parameters]
  [-tf transform file] [-z optimizer scales]
```

The final mode of operation, transform mode, is started using this last command

```
PetSpectFusion  -d -f fixed image -m moving image -o output image
  [-tf transform] [-t transform parameters]
```

With this command the last two parameters are not really optional, you have to specify one or the other. PetSpectFusion will then load the two images, resample the moving image into the fixed image's space, apply the transform and then exit saving the result to the output image.

A.4.2 Command Line Parameters

PetSpectFusion allows all of the settings for the registration, as well as internal settings such as logging or plotting output files, to be specified using the command line. The table below lists all acceptable parameters as well as their standard value ranges.

Table A.1: **List of available command line parameters**

Parameter	Standard Value	Description
-a	n.a.	Puts the program into automatic mode, i.e close upon completion
-b	40 .. 80	Sets the number of bins used by the mattes registration algorithm
-c	1 .. 8	Sets the number of threads used for the registration and image processing (defaults to number of processors detected on the system
-d	n.a.	Do transform only. Applies the transform to the moving image, saves and exits
-f	n.a.	Sets the fixed image file name to load
-g	n.a.	Turns on the graphical user interface
-h	n.a.	Display the help message
-i	250 .. 500	Sets the maximum number of iterations used by the registration
-j	0.001 .. 1.0	Sets the minimum step size taken by the optimizer (in millimeter)
-k	0.1 .. 4.0	Sets the maximum step size taken by the optimizer (in millimeter)
-l	n.a.	Turns on logging and sets the log file name

Continued on next page ...

Table A.1 – continued from previous page

Parameter	Standard Value	Description
-m	n.a.	Sets the moving image file name to load
-o	n.a.	Sets the output image file name to save
-p	n.a.	Turns on Gnuplot data file generation with the plot file name
-r	2 .. 4	Turns on multi-resolution registration using the number of levels
-s	0.1 .. 1.0	Sets the image sampling rate
-t	n.a.	Applies the initial transform parameters to the moving image, example: *-t RotX RotY RotZ TransX TransY TransZ* where the rotations are specified in radians and the translations in millimeter. All six values must be specified with this parameter
-tf	n.a.	Applies the initial transform parameters to moving contained in the transform file generated from a previous registration
-z	0.000001 .. 10.0	Sets the optimizer scales used by the registration, example: *-z xRot yRot zRot xTrans yTrans zTrans* where each value specifies the weight of transformation in that direction, the higher the value the more heavily that direction influences the optimizer. All six values must be specified with this parameter

A.5 Graphical User Interface Reference

A.5.1 Features

The PetSpectFusion GUI offers many features for analyzing 3d image registration. The two images are displayed alpha blended on top of one another in the window using three slice views, transaxial, axial, and coronal. Each image is displayed using a different logarithmic color table. The fixed image is initialy displayed using a hot red color table, and the moving image using a hot blue color table, however this can be changed interactively. The user can choose from four different color tables, two linear tables, and two non linear tables. Underneath each slice view is a slider that allows the slice being displayed to be changed, effectively allowing the user to scroll through the particular plane. In addition to that there is also an alpha blending slider, that allows the user to control how strongly the moving image is blended into the fixed.

One of the most valuable features offered by PetSpectFusion is the ability to observe the registration in action. Each time the optimizer takes another step, the GUI is refreshed causing all of the slice views to be redrawn using the current transform parameters. During this time the user still has the ability to move the slice sliders and the alpha slider. This allows the user to dynamically examine different regions during the registration.

Beyond this the GUI also lets the user control various other aspects of how the image

slices are displayed. The user can choose from four levels of magnification for the displayed slices, from one hundred percent to four hundred percent. There is also the ability to change the slice view type from alpha blended mode to checkerboard mode. In Checkerboard mode each slice is divided into a four by four grid of equally sized sqaures. Each of the two images alternatively fills a square, so that no two direct neighboring squares are filled by the image, hence the checkerboard reference. This mode is especially good for viewing the rough alignment of the images, because it allows uses the user to see if curvature in one image is aligned with the curvature of the other.

For manual registration purposes the GUI also allows the user to specify the transform applied to the moving image. Together with this the user can also evaluate the registration metric for the manually entered transform, providing a very useful tool for evaluating the robustness of the automatic registration. Once a correct registration has been performed, it is possible to save the transformed moving image, as well as a variety of other diagnostic images, such as an absolute difference image or a checkerboard composite image.

A.5.2 Basic Usage

The GUI can be started by either double clicking the application's icon or by calling the program from the command line with no parameters or the parameter **-g**. In figure A.1 you can the PetSpectFusion GUI when the program first loads. The GUI has a fairly simply structure, at the top is the menu bar (on OS X the menu bar sits at the top of the screen detached from the actual GUI window), followed by the three slice view panes, then the transform settings box and the progress box and finally the registration buttons.

The fixed and moving images can loaded using the *File* menu. If you load the moving image before you load the fixed image, you might notice that no image is displayed. This is because of the way the registration process works. In image registration, the moving image is always mapped into the fixed image's space. In order to do this, you first need to know the fixed image's physical size, thats why the moving image is only displayed once the fixed image is loaded. Figure A.2 shows what the slice views look like when the fixed and moving images are loaded.

Once the fixed and moving images are loaded, PetSpectFusion will automatically perform a transform to align the geometrical center of the moving with the geometrical center of the fixed image. You will notice the notice non zero transform values in the GUI, depending upon the physical size difference between the moving image and the fixed image. You may also notice that the edges of the moving image have been cropped, if it occupies a large physical space than the fixed image. This is because this program is concerned with aligning information in the fixed image with like information in the moving image. Information in the moving image that does not correspond to anything in the fixed image is thus uninteresting and ignored.

You can start the automatic registration by clicking the *Perform Registration* button or by selecting the menu *Registration / Perform Registration*. Once the registration is running the *Transform Settings* group box become locked as does the *Evaluate Current Transform* button. The GUI will be updated everytime the optimizer has taken another step. During this time you can still change the slices being displayed as well as the alpha blending value. The registration can be stopped at any time by clicking the *Stop Registration* button.

When the registration is completed the *Transfrom Settings* group box will be unlocked

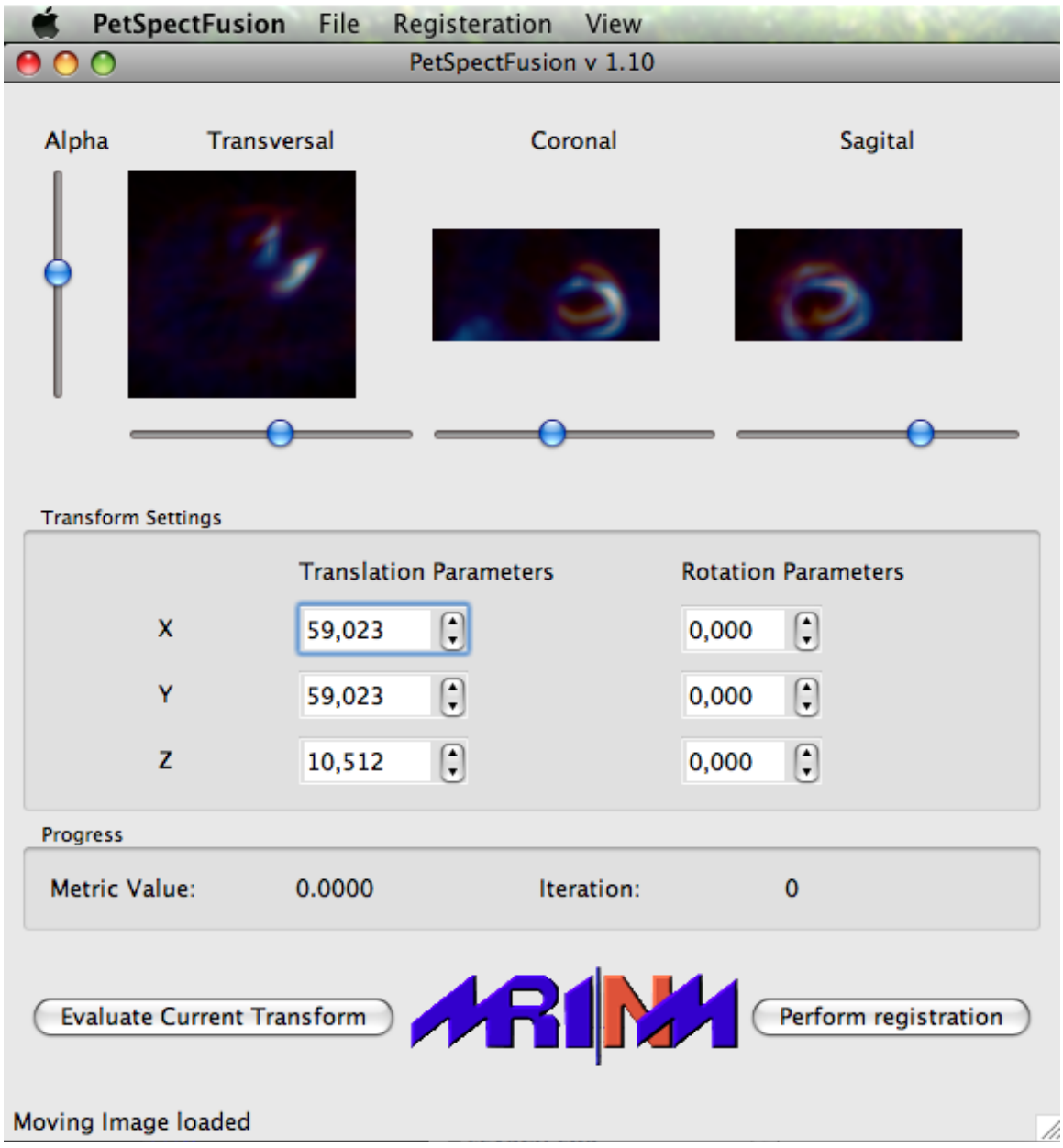


Figure A.1: PetSpectFusion GUI with a fixed and moving image pair loaded

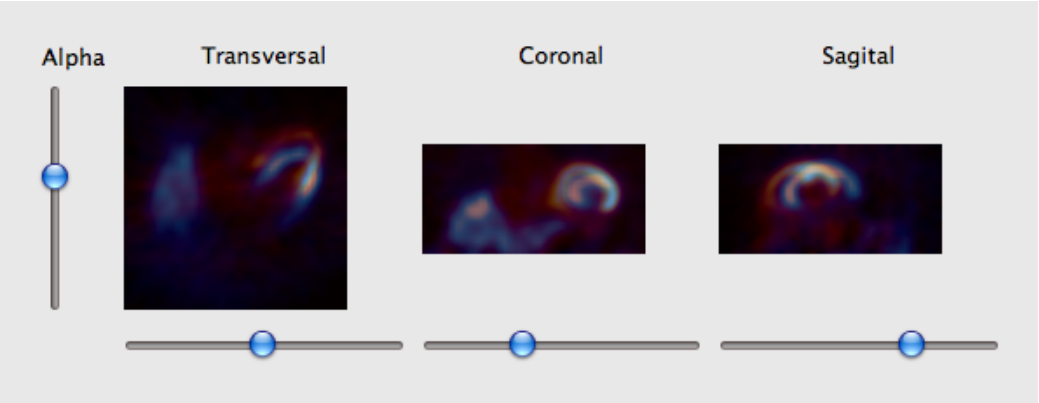


Figure A.2: PetSpectFusion slice views displaying a SPECT image (in the hot blue color table) alpha blended onto a PET image (in the hot red color table) of the same patient

and the GUI will display the final metric value. You can then use the sliders to inspect the correctness of the result. Probably the most important tool for evaluating the correctness is the alpha blending slider because it allows you to fade the images out quickly, allowing you a better view of which parts of the images are aligned.

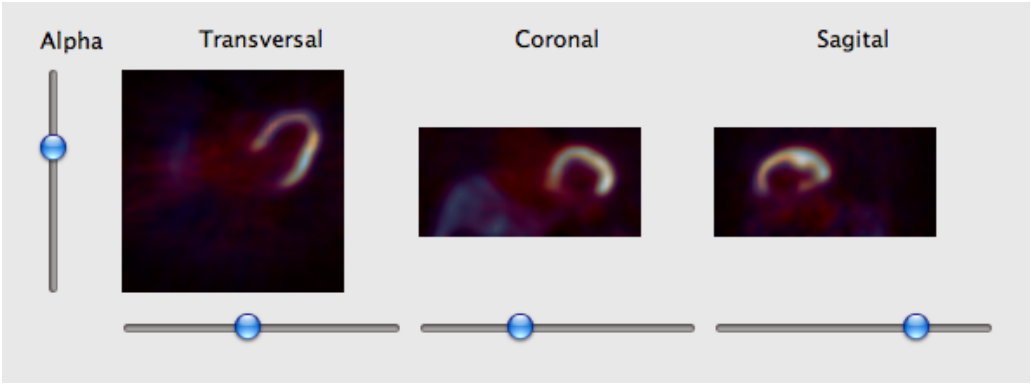
Figure A.3 shows how you can examine a result using alpha blending. The intense regions in both images represent the left atrium of the heart. As you can see, the exact shape and size of the heart in both images not exactly the same, nonetheless the curvature of two is aligned correctly. This is to be expected when registering images from different modalities, anatomical features are not guaranteed to have the same shape between modalities.

When you are ready to save the result there are a few different possibilities to choose from. By selecting *File / Save ...* you can chose the type of image you wish to save, whether it is the registered moving image or a checkerboard composite image.

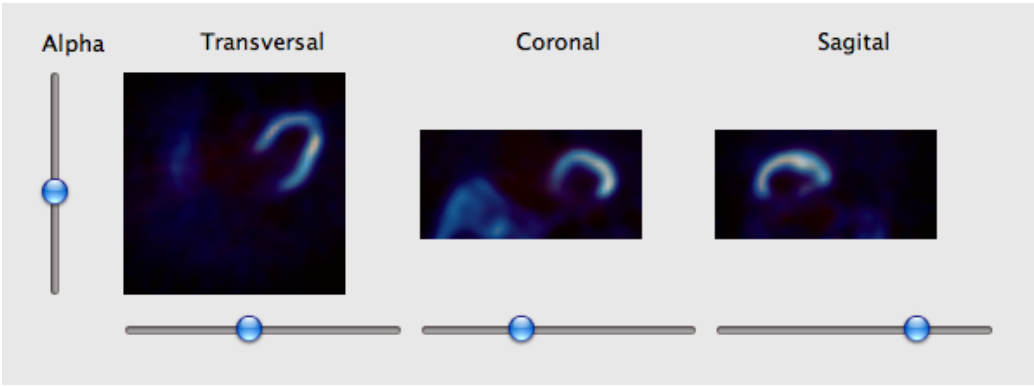
A.5.3 Advanced Functions

The Pet SpectFusion GUI offer several advanced features for controlling the registration settings and how the composite images are displayed. One of the simplest features offered is the ability to change the magnification level of the slice views. By selecting *View / Zoom* you can set the zoom level, from one hundred percent to four hunderd percent. The color lookup table used by the fixed and moving images can also be changed using the GUI. There are two gradient based lookup tables and two linear tables. These are also changed using the *View* menu.

You also have the ability to turn on checkerboard composite mode. When this is activated the slice views no longer display the two images alpha blended onto each other, but instead displays a checkerboard composite of the two. An example of this can be seen in figure A.4. This mode gives you information on how well the similar shapes edges match up and if the curvature between the two is correctly aligned. Unless both images come from the same image modality it is unlikely the edges will match perfectly, simply because of the different spatial distortions present in each modality.



(a)



(b)

Figure A.3: This shows the result of an automatic registration alpha blended heavily towards the fixed image (a) and heavily towards the moving image (b)

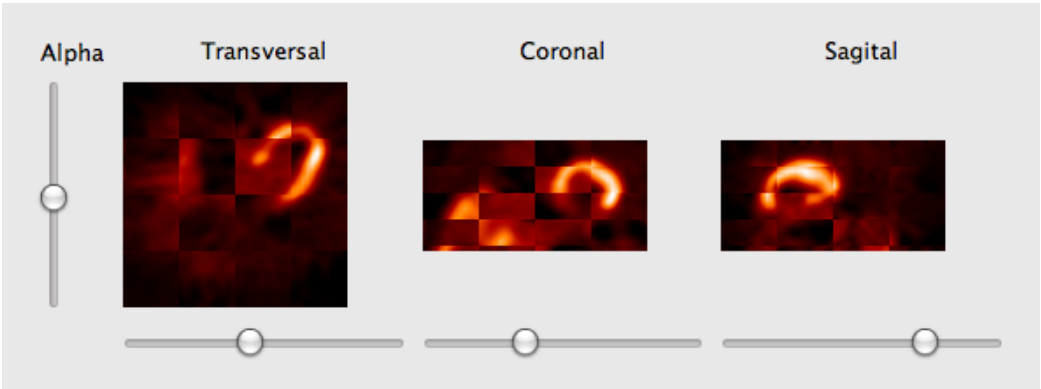


Figure A.4: PetSpectFusion slice views displaying a checkerboard composite of a SPECT and PET image

The registration parameters can be modified by selecting *Registration / Options*. Figure A.5 shows the options dialog, which allows you to modify the same parameters that can be modified from the command line. For more information about these parameters see section A.4.2.

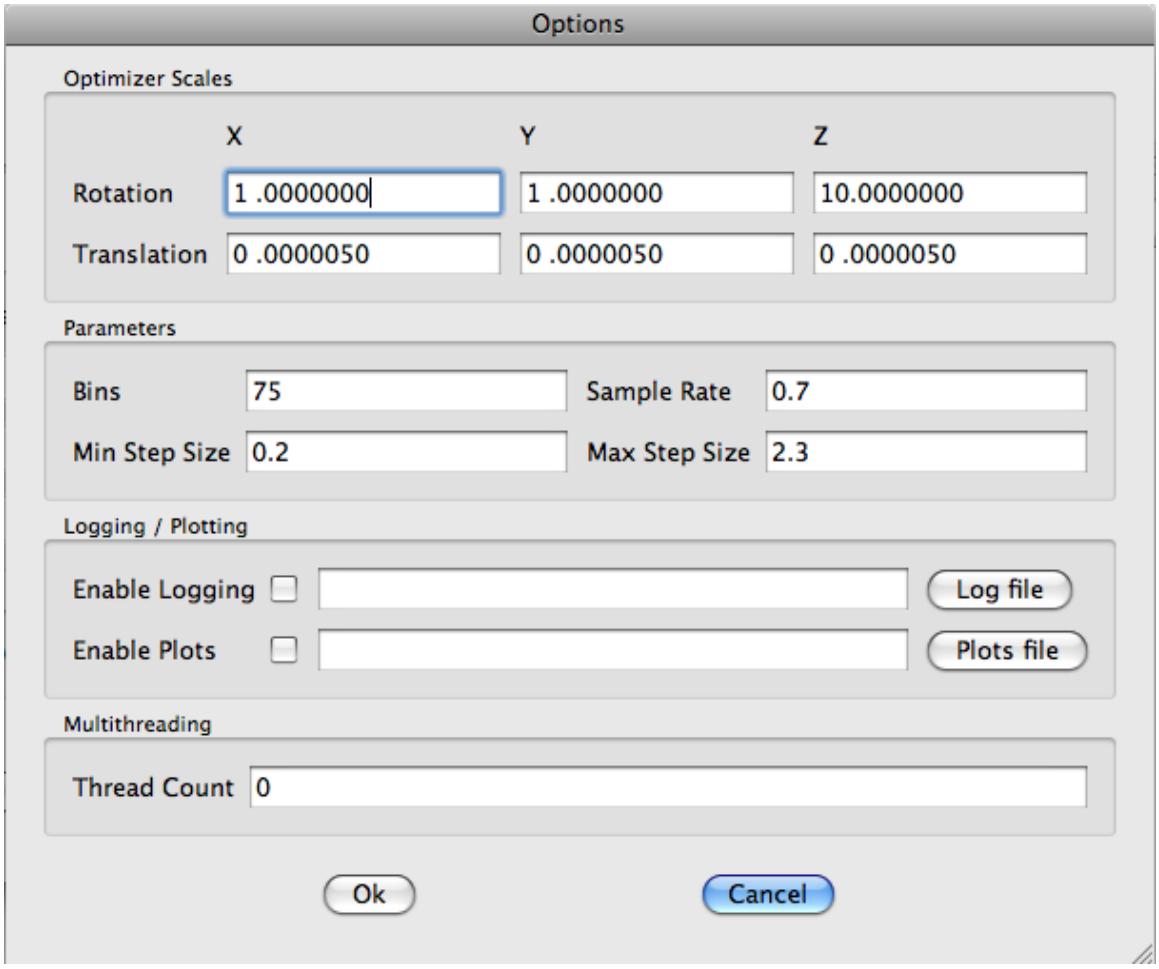


Figure A.5: PetSpectFusion registration options dialog

In the GUI you can also activate multi-resolution registration mode (which was discussed earlier in this report). This is accomplished by selecting *Registration / Use Multi Resolution*. The next time you perform a registration it will use multi-resoltution mode with the number of levels specified in options dialog.

A.6 Building PetSpectFusion from source

Compiling the source code is a somewhat complicated procedure. Before you can build the program, you need to have the following software installed:

- CMake (http://www.cmake.org the cross platform make file generator (Version 2.6 or higher)

- Qt (http://trolltech.com/downloads/opensource) a cross platform C++ graphics framework (Version 4.4 or higher, 4.5 for 64 bit support on OS X)
- ITK (http://www.itk.org) the Insight Registration and Segmentation Toolkit (Version 3.10 or higher)

Once you have this software installed you need to get a copy of the source code, either directly from the CAMP SVN server, or from http://home.in.tum.de/ jensen/downloads. The first step of building PetSpectFusion involves running CMake in the top level directory of the source code. In CMake you will have to specify the installation directory of ITK and Qt on your system. If everything was successful CMake will generate a build script or project file appropriate for your system, on OS X it generates Xcode projects, on Windows it generates visual studio projects. It is recommended that the build directory be separate from the source code directory, so that the source code can be cleanly packaged when the program is ready for a release.

You can then use the development environment generated by CMake for modifying the source code files. Do not try to modify internal settings in the generated project, these changes will just be overwritten the next time you build the project, because CMake recreates the project every time it detects something has changed. If you need to add or remove any source files, you will have to modify the file *CMakeLists.txt*. This file contains the internal setting of the project, including which source files belong to the project and what libraries are used by the project. This file is reread everytime you build the project, which in turn recreates the project file.

For more information on this behavior view the CMake documentation.

Bibliography

[1] P.J. Besl and N.D. McKay. A method for the registration of 3-D shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14:239 – 256, 1992.

[2] Manuel D. Cerqueira and Arnold F. Jacobson. Assessment of Myocardial Viability with SPECT and PET Imaging. *American Roentgen Ray Society*, pages 477–483, September 1989.

[3] R. Duda and P. Hart. *Pattern Classification and Scene Analysis*. John Wiley and Sons, 1973.

[4] Stefan Eberl, Iwao Kanno, Roger R. Fulton, Anneke Ryan, Brian F. Hutton, and Michael J. Fulham. Automated Interstudy Image Registration Technique for SPECT and PET. *Journal of Nuclear Medicine*, pages 137 – 145, 1996.

[5] Guido Germano. Technical Aspects of Myocardial SPECT Imaging. *Journal of Nuclear Medicine*, 42:1499 – 1507, 2001.

[6] Maria Carla Gilardi, Giovanna Rizzo, Annarita Savi, Claudio Landoni, Valentino Bettinardi, Claudio Rossetti, Giuseppe Striano, and Ferruccio Fazio. Correlation of SPECT and PET cardiac images by a surface matching registration technique. *Computerized Medical Imaging and Graphics*, pages 391 – 398, July 1998.

[7] B. F. Green. The orthogonal approximation of an oblique structure in factor analysis. *Phychometrika*, 17:429 – 440, 1952.

[8] Mark W. Groch and William D. Erwin. SPECT in the Year 200: Basic Principles. *Journal of Nuclear Medicine Technology*, 28:233–244, 2000.

[9] Derek L G Hill, Philipp G Batchelo, Mark Holden, and Dave J Hawkes. Medical Image Registration. *Physics in Medicine and Biology*, 46:R1 – R45, 2001.

[10] B. K. P. Horn. *Robot Vision*. MIT Press, 1986.

[11] B. K. P. Horn. Closed-form solution of absolute orientation using unit quaternions. *J. Opt. soc. America A.*, 4:629 – 642, 1987.

[12] Luis Ibanez, Will, Schroeder, Lydia Ng, Josh Cates, and Insight Software Consortium. *The ITK Software Guide Second Edition*. Kitware Inc, November 2005.

[13] Philipp A. Kaufmann, Paolo G. Camici, and S. Richard Underwood. *The ESC Textbook of Cardiovascular Medicine*, chapter 5, pages 141 – 158. Wiley, John & Sons, Incorporated, 2006.

[14] Mark T. Madsen. Recent Advances in SPECT Imaging. *Journal of Nuclear Medicine*, 48:661 – 673, 2007.

[15] D. Mattes, D. R. Haynor, H. Vesselle, T.K. Levellen, and W. Eubank. PET-CT Image Registration in the Chest Using Free-form Deformations. *IEEE Transactions on Medical Imaging*, 22:120 – 128, 2003.

[16] R. A. Peace, R. T. Staff, H. G. Genmell, F. I. Mckiddie, and M. J. Metcalfe. Automatic detection of coronary artery diesease in myocardial perfusion SPECT using image registration and voxel to voxel statistical comparisons. *Nuclear Medicine Communications*, 23:785 – 794, 2002.

[17] C. A. Pelizzari, G. T. Y. Chen, D. R. Spelbring, R. R. Weichselbaum, and C. T. Chen. Accurate three-dimensional registration of CT, PET, and/or MR images of the brain. *Journal of Computer assissted tomography*, 13:20–26, 1989.

[18] J. P. Pluim, J. B. A. Maintz, and M. A. Viergever. Mutual-Information-Based Registration of Medical Images. In *IEEE Transactions on Medical Imaging*, pages 986 – 1004, August 2003.

[19] Loren Schwarz. MR-based Attenuation Correction for PET, August 2006.

[20] Piotr J. Slomka, Hidetaka Nishina, Daniel S. Berman, Xingping Kang, John D. Friedman, Sean W. Hayes, Usaf E. Aladl, and Guido Germano. Automatic qauntification of myocardial perfusion stress-rest change: a new measure of ischaemia. *The Journal of Nuclear Medicine*, 45:183–191, 2004.

[21] Timothy G. Turkington. Introduction to PET Instrumentation. *Journal of Nuclear Medicine Technology*, pages 1 – 8, 2001.

[22] P. Viola and W.M. Wells III. Alignment by Maximalization of Mutual Information. *International Journal of Computer Vision*, 24:137 – 154, 1997.

[23] Wikipedia. Reproduced from http://en.wikipedia.org/wiki/File:Gradient_descent.png, 2004.

[24] Wikipedia. Reproduced from http://de.wikipedia.org/w/index.php?title=Datei:Indexed_palette-H.png&filetimestamp=20050617112550, 2005.

[25] Wikipedia. Derived from http://en.wikipedia.org/wiki/File:Enclosing_points2.svg, 2008.