AKADEMIA GÓRNICZO-HUTNICZA IM. STANISŁAWA STASZICA W KRAKOWIE

AGH UNIVERSITY OF SCIENCE AND TECHNOLOGY
FACULTY OF ELECTRICAL ENGINEERING, AUTOMATICS, COMPUTER SCIENCE AND BIOMEDICAL ENGINEERING

DEPARTMENT OF AUTOMATICS AND BIOMEDICAL ENGINEERING

DADM project documentation - draft

Authors:	*SIMENS Healthkare*
Degree programme:	*Biomedical engineering*
Supervisor:	*PhD. Tomasz Pięciak*

Kraków, 2017

Contents

1. List of changes

Name	Date	Details
Sylwia Mól	19-Nov-2017	Document created
Sylwia Mól	20-Nov-2017	Structure changed
Sylwia Mól	21-Nov-2017	Chapter "Authors" added, in-out table added
Malwina Molendowska	26-Nov-2017	Description of 1st module added
Eliza Kowalczyk	27-Nov-2017	Description of 9th module added
Karolina Gajewska	27-Nov-2017	Description of 10th module added
Eliza Kowalczyk	28-Nov-2017	Description of 9th module changed

2. Assumptions

about the app - the aim, what you can do here etc.

3. Structure

dependences (tree), modules‘ descriptions

Module	Input	Output	Befo
1	**k**-space signals	**x**-space fully reconstructed data	
2			
3			
4			
5			
6			
8			
9	320x240 image	640x480 image	de
10	1)segmentated image 2)defined plane, image	1)3D model (e.g. vtkPolyData) 2)cross-section image	1)segmenta
11			

4. User guide

4.1. Requirements

what user need to use this app - e.g. windows version etc

4.2. Instruction

instructions for user - GUI screens etc

5. Detailed description

5.1. Module 1

The aim of this module is to formulate mathematical algorithm, which enables proper data reconstruction for images obtained with parallel MRI scans. The reconstruction is performed with use of Sensitivity Encoding (SENSE) algorithm in least squares (LS) solution context and Tikhonov regularization method.

Generally, parallel MRI acquisitions are targeted to diminish time needed for data sampling. The usage of multiple coils enabled simultaneous acquisition of signals. A further step, which is acquiring partial data from **k**-space, leads to craved time savings, meanwhile maintaining full spatial resolution as well as contrast at the same time. However, the approach of omitting lines in acquisition step results in data aliasing, i.e. folded images that need further data processing.

To clearly mark out how data is processed in this module, we list following reconstruction steps: i) the application of 2D Fourier Transform transform (2D FFT) to **k**-space data (acquired raw signals) from multiple coils. The result is a set of **x**-space images with folded pixels, ii) the sensitivity maps estimation of coils profiles (the information is needed to properly unfold subsampled data) and iii) the proper unfolding data process with usage of SENSE reconstruction algorithm and its alterations.

The most crucial step in processing is estimation of sensitivity coil profiles as a successful image reconstruction with use of pMRI algorithms highly depends on accurate sensitivity coil assessment. As sensitivity information varies from scan to scan it is impossible to obtain absolute maps. To obtain reliable knowledge, reference scans have to be conducted each time an examination is performed. These low-resolution information helps to estimate coil profiles with use of the many methods i.e. dividing each component coil image by a 'sum of squares' image.

It basic formulation, SENSE algorithm is applied to Cartesian MRI data undersampled uniformly by a factor r (i.e. $r = 2$ means that every other line in **k**-space is skipped). After Fourier transformation, each pixel in **x**-space image received in l-th coil can be seen as weighted sum of r pixels from full FOV, each multiplied by corresponding localized values of maps. The distance between those 'aliased' points in the full FOV is always equal to the desired FOVy value divided by subsampling rate. Obviously, depending on subsampling rate the number of folded pixels changes. Basically, the signal in one pixel at a certain location (x, y) received from l-th component coil image D_l^S with chosen subsampling rate r can be written as:

$$D_l^S(x,y) = S_l(x,y_1)D^R(x,y_1) + S_l(x,y_2)D^R(x,y_2) + ... + S_l(x,y_r)D^R(x,y_r), \tag{5.1}$$

where index l counts from 1 to L (number of coils) and index i counts from 1 to r. Eq.(5.1) can be rewritten as:

$$D_l^S(x,y) = \sum_{i=1}^{r} S_l(x,y_i)D^R(x,y_i) \quad \text{for} \quad l = 1,...,L. \tag{5.2}$$

Including all L coils the above equation can be rewritten in a matrix form:

$$\mathbf{D}^S(\mathbf{x}) = \mathbf{S}(\mathbf{x})\mathbf{D}^R(\mathbf{x}), \tag{5.3}$$

The vector $\mathbf{D}^S(\mathbf{x})$ denotes the aliased coil image values at a specific location $\mathbf{x}$ = (x, y_i) and has a length of L, $\mathbf{S}(\mathbf{x})$ is a LxR matrix and represents the sensitivities values for each coil at the r superimposed positions and $\mathbf{D}^R(\mathbf{x})$ lists the r pixels from full FOV image to be reconstructed. The closed-form solution of the problem is as follows:

$$\widehat{\mathbf{D}^R(\mathbf{x})} = (\mathbf{S}^H(\mathbf{x})\mathbf{S}(\mathbf{x}))^{-1}\mathbf{S}^H(\mathbf{x})\mathbf{D}^S(\mathbf{x}), \tag{5.4}$$

where $\widehat{\mathbf{D}^R(\mathbf{x})} = [\widehat{D^R(x, y_1)}, ..., \widehat{D^R(x, y_r)}]^T$ and $\mathbf{S}^H(\mathbf{x})$ is the conjugate transpose of the $\mathbf{S}(\mathbf{x})$ matrix. The final reconstruction image is defined as:

$$M(\mathbf{x}) = \left|\widehat{\mathbf{D}^R(\mathbf{x})}\right|. \tag{5.5}$$

The 'unfolding' process can be performed as long as inversion of $\mathbf{S}(\mathbf{x})$ matrix is possible. Therefore, we cannot set the value of subsampling rate exceeding the number of coils L. To restore full FOV data, SENSE algorithm has to be recalled for each pixel in aliased $\mathbf{x}$–space image.

A regularization approach is defined as an inversion method that introduces additional information in order to stabilize the solution. This method is beneficial as it roughly matches the desired solution and is less sensitive to perturbations of the data. Tikhonov regularization is a common approach to obtain an inexact solution to a linear system of equations. In particular, the Tikhonov regularized estimate reads as follows:

$$\widehat{\mathbf{D}^R_{reg}} = \underset{\mathbf{D}^R}{\text{argmin}} \left\{ \left\| \mathbf{D}^S - \mathbf{S}\mathbf{D}^R \right\|^2 + \lambda^2 \left\| \mathbf{A}(\mathbf{D}^R - \mathbf{D}) \right\|^2 \right\}, \tag{5.6}$$

where λ is a regularization parameter ($\lambda > 0$) and $\mathbf{D}$ is a prior image known as a regularization image. Selection of the parameter λ and $\mathbf{D}$ can be performed using different procedures. In this module $\mathbf{A}$ is assumed to be an identity matrix. The first term provides fidelity to the data and the second introduces prior knowledge (e.x. median filtered initial guess of LS SENSE) about the expected behaviour of $\mathbf{D}^R$. The Tikhonov regularization problem is given by:

$$\widehat{\mathbf{D}^R_{reg}} = \mathbf{D} + (\mathbf{S}^H\mathbf{S} + \lambda\mathbf{A}^H\mathbf{A})^{-1}\mathbf{S}^H(\mathbf{D}^S - \mathbf{S}\mathbf{D}). \tag{5.7}$$

A reasonable value for λ can be picked using many technique, i.e. the L-curve criterion or generalized cross-validation.

Module input: Synthetic MR images are brain MRI slices coming from BrainWeb are normalized to [0-255] (all with intensity non-uniformity INU=0). Only T1- and T2-weighted data is used. The dataset is free of noise and the background areas are set to zero. The slice thickness equals 1 mm. These images are used then to simulate synthetic noisy accelerated parallel Cartesian SENSE MRI data according to following steps (the data simulation is performed with use of eight receiver coils ($L = 8$)): i) simulated sensitivity maps (divided into the ratio 3:1 for real and imaginary parts, respectively) are added to fully-sampled $\mathbf{x}$-space data, ii) correlated complex Gaussian noise with different values of standard deviations is added to each coil image, iii) 2D FFT and data subsampling with chosen reduction factor r is performed and iv) 2D iFFT is applied to recover data in $\mathbf{x}$-space. Then, data reconstruction process is conducted.

Module output: The output is full resolution reconstructed data performed with two different algorithms: SENSE (LSE) and Tikhonov regularization.

5.2. Module 2

-detailed description of module, algorithm etc, NO CODES, only theoretical!

5.3. Module 3

-detailed description of module, algorithm etc, NO CODES, only theoretical!

5.4. Module 4

-detailed description of module, algorithm etc, NO CODES, only theoretical!

5.5. Module 5

-detailed description of module, algorithm etc, NO CODES, only theoretical!

5.6. Module 6

-detailed description of module, algorithm etc, NO CODES, only theoretical!

5.7. Module 8

-detailed description of module, algorithm etc, NO CODES, only theoretical!

5.8. Module 9

Interpolation is a method of constructing new points based on the existing ones. In other words finding a value of a new point in High Resolution (HR) image based on points in Low Resolution (LR) pictures.

In classical interpolation techniques pixels in LR data y can be related to the corresponding x values of HR data: $y_p = \frac{1}{N}\sum_{i=1}^{N} x_i + n$, where y_p is the pixel of LR image at location p, x_i is each one of the N High Resolution pixels contained within this LR pixel and n is some additive noise.

The biggest problem is that to find High Resolution data from the Low Resolution values. Unfortunately, there is an infinite number of values that meet that condition. Interpolation methods can be divided into three basic techniques.

The first group are the most common ones, like linear or spline-based interpolation. These techniques assume that it is possible to count the value of a new point by determination some kind of generic function. The main disadvantage is that they are correct only for images of homogeneous regions. As is well known, brain consists of grey substance, white substance and cerebral spinal fluid, so above-mentioned methods are not appropriate for MRI images interpolation. The second one is Super Resolution technique. It is commonly used to increase image resolution on functional MRI (fMRI) and Diffusion Tensor Imaging (DTI). The method is based on acquisition of multiple Low Resolution (LR) images of the same object. It is time consuming and not adequate for clinical applications.

The last but not least method is non-local patch-based technique which is based on self-similarity of a single image. It is possible to improve resolution by extracting information from a single image instead of acquiring several pictures.

The aim of this module is to increase MRI image resolution by upsampling. The input data is a single 320x240 image. After the processing it is going to be twice as big. To be more specific 640x480 pixels. The process can be named as double upsampling.

The first step of the algorithm is to divide every pixel of LR image into more pixels. Then the patch is designed. It is a rectangle which will be the area of interest. The pixels that are inside the area are taken into account during calculation the values of new points. After that an appropriate estimator is used to correctly calculate the values of new pixels. Every pixel has to be classified into one of three groups (grey substance, white substance or cerebral spinal fluid).

5.9. Module 10

To prepare tree dimension visualization of the cerebral cortex is used algorithm of marching cubes.

The input data is multiple 2D slices of MR image. The marching cubes algorithm create a polygonal representation of constant density surfaces from a 3D array of data. To select the cerebral cortex is used output data from segmentation made in module 8. The space of the image is divided into a regular grid of cubes. In each iteration one cube is considered. At each vertex of cube is determined how the surface intersects this cube. The density value and compared with the limit value - surface constant. If the data value is bigger than suface constant, one is assinged to a cube's vertex. There are 256 combinations of cube orientation relative to the surface, but we can distinguish 15 basic patterns, that repeat as symmetrical reflections, produces all possibilities (Fig. 5.1). If all values are less than the constant value, then the cube does not form any polygon. Otherwise, the edges of the polygon are defined (by linear interpolation) at the edges that intersect the surface. Using central differences, a unit normal at each cube vertex is calculated and then normal to each trangle vertex is interpolated. The output of the algorithm is the triangle vertices and vertex normals.

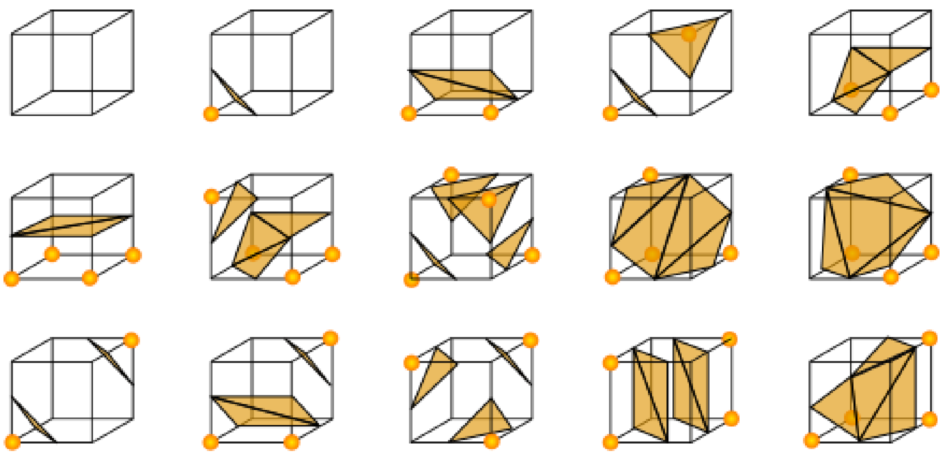

Figure 5.1. Triangulation for the 15 patterns.

visualization the model, obtained by marching cubes, the VTK library is used, which enables building the three-dimension model.

The second part of this module includes visualization of the brain's cross-section on arbitrarily defined plane.

There are three primary imaging planes that are performed in medical imaging:

– axial plane, which is any plane that divides the body into superior and inferior parts, roughly perpendicular to spine.

– sagittal plane, which is any imaginary plane parallel to median plane.

– coronal plane, which is any vertical plane that divides the body into anterior and posterior sections.

The MRI produces two-dimensional images that consist of slices of brain's and is usually performed in axial plane. To receive images in the remaining planes, linear interpolation is used

5.10. Module 11

-detailed description of module, algorithm etc, NO CODES, only theoretical!

6. Implementation

6.1. Tools

-python version, libraries

6.2. Module 1

-code

6.3. Module 2

-code

6.4. Module 3

-code

6.5. Module 4

-code

6.6. Module 5

-code

6.7. Module 6

-code

6.8. Module 8

-code

6.9. Module 9

-code

6.10. Module 10

-code

6.11. Module 11

-code

7. Tests

7.1. Module 1

-module 1 tests

7.2. Module 2

7.3. Module 3

7.4. Module 4

7.5. Module 5

7.6. Module 6

7.7. Module 8

7.8. Module 9

7.9. Module 10

7.10. Module 11

7.11. Application

-whole app tests

8. Authors

Authors of this project are students of Biomedical Engineering, AGH UST, Krakow, Poland.

Name	Role
Sylwia Mól	Project Manager
Jacek Fidos	Software architect
Maciej Gryczan	GUI engineer
Adrian Stopiak	Vizualization engineer
Malwina Molendowska	1st module developer
Klaudia Gugulska	2nd module developer
Kacper Turek	3rd module developer
Magdalena Rychlik	4th module developer
Alicja Martinek	5th module developer
Mateusz Pabian	6th module developer
Anna Grzywa	8th module developer
Magdalena Kucharska	9th module developer
Eliza Kowalczyk	9th module developer
Karolina Gajewska	10th module developer
Michał Kotarba	11th module developer

List of Figures