COMPRESSED SENSING BASED IMAGE PROCESSING WITH APPLICATION TO 3D MRI

ABEERA TARIQ

RESEARCH CENTRE FOR MODELING & SIMULATION

NATIONAL UNIVERSITY OF SCIENCES AND TECHNOLOGY

2015

COMPRESSED SENSING BASED IMAGE PROCESSING WITH APPLICATION TO 3D MRI

ABEERA TARIQ

Research Centre for Modeling & Simulation

A thesis submitted to the National University of Sciences & Technology

Impartial fulfillment of the requirement for the degree of

Masters in Computational Science and Engineering

2015

STATEMENT OF ORIGINALITY

I hereby certify that the work embodied in this thesis is the result of original research and has not been submitted for a higher degree to any other University or Institution.

Date	Abeera Tariq

Dedication

This thesis is dedicated to:
The sake of Allah, my Creator and my Master,
My great teacher and messenger, Mohammed (May Allah bless
and grant him), who taught us the purpose of life,
My great parents, who never stop giving of themselves in countless ways,
My teachers at the RCMS Department, who gave me the background knowledge,
My beloved brothers and sister;
My friends who encourage and support me,
All the people in my life who touch my heart. ☺.

Acknowledgements

I've always believed that if you put in the work, the results will come.

Michael Jordan

In the Name of Allah, the Most Merciful, the Most Compassionate all praise is to Allah, the Lord of the worlds; and prayers and peace be upon Mohamed His servant and messenger.

First and foremost, I must acknowledge my limitless thanks to Allah, the Ever-Magnificent; the Ever-Thankful, for His helps and bless. I am totally sure that this work would have never become truth, without His guidance.

I owe a deep debt of gratitude to our university for giving us an opportunity to complete this work .

I am grateful to some people, who worked hard with me from the beginning till the completion of the present research particularly my supervisor Dr. Salman Shahid, who has been always generous during all phases of the research, and I highly appreciate the efforts expended by Engr. Sikander Hayat Mirza and Sir Fawad Khan.

I would like to take this opportunity to say warm thanks to all my beloved friends, who have been so supportive along the way of doing my thesis.

I also would like to express my wholehearted thanks to my family specially my parents and husband, Muhammad Gohar for their generous support they

provided me throughout my entire life and particularly through the process of pursuing the master degree. Because of their unconditional love and prayers, I have the chance to complete this thesis.

Last but not least, deepest thanks go to all people who took part in making this thesis real.

Table of Contents

List of Figures

List of Tables

Abstract

In the present age of technology, the buzzwords are low-power, energy-efficient and compact systems. This directly leads to the data processing and hardware techniques employed in the core of these devices. One of the most power-hungry and space-consuming schemes are that of image or video processing, due to its high quality requirements. In current design methodologies, a point has been reached in which physiological and physical impacts restrict the capacity to simply encode information faster. These limits have led to search for strategies to diminish the amount of obtained data without losing information and increase vitality and time effectiveness.

Compressive sensing (CS) is an emerging technology which is based on an efficient signal compression and reconstruction technique. It can be used to efficiently reduce the data acquisition and processing. It utilizes the sparsity of a signal in a different transform domain for sampling and stable reconstruction. It is another opportunity to conventional data processing and is robust in nature. Unlike the conventional methods, CS provides an information capturing paradigm with both sampling and compression. It allows signals to be sampled below the Nyquist rate, and still allowing optimal reconstruction of the signal. The required measurements for reconstruction are far less than those of conventional methods, and the process is non-adaptive, making the sampling process faster and universal.

In this thesis, CS method is applied to magnetic resonance imaging (MRI), which is popularly used imaging technique in clinical applications and image

compression. Over the years, MRI has enhanced significantly in both imaging quality and working speed. This has further advanced the field of diagnostic medication. In any case, imaging speed, which is fundamental to numerous MRI applications, still remains a major question. Moreover, the fast growing 3D MRI images and real-time MRI scanning has become more complex leading to a further increase in data acquisition times. On the other hand, to improve the speed, it becomes necessary that the processing algorithms are also computationally sped up. There have been various attempts where Graphic processing unit (GPUs), Field programmable gate array (FPGAs), clusters and central processing unit (CPUs) are used for this purpose, which have time durations ranging from minutes to hours. Again, though, there are solutions for reduction in computation time, the desire for the least computational time needs to be ascertained.

Considering the requirements discussed above, the work in this thesis is presented in two parts. In the first part, a scheme for 3D MRI reconstruction is proposed by taking the benefit from collaborative sparsity, which at the same time enforces nonlocal 3D sparsity and local 2D sparsity in a hybrid transform domain. An efficient and improved augmented Lagrangian technique is used for the solution of CS optimization problem which enhances the MRI performance when compared with the conventional sampling. Experimental results for all slices are combined to form a 3D view. This work reveals the efficacy of already proposed 2D recovery algorithm by reconstructing image in 3D.

As, 3D MRI experiences long scan time, and CS can significantly reduce the encoding time for 2D magnetic resonance images, therefore, for faster 3D MRI

reconstruction the algorithm is parallelized. So, in the second part this problem is tackled by using the benefits of multicore architecture and GPU. The sequential Compressed Sensing reconstruction algorithm is made parallel using different parallelizing techniques and compared on the basis of Peak signal to noise ratio (PSNR), correlation coefficients, Relative error (RE), Root mean square error (RMSE), histogram and topography measure. The effect of GPU based implementation is not significant because the application of complex collaborative sparsity on small blocks of images forms the algorithm repetitive in structure. This bottleneck lies because of a lot of time in data transfer from host to GPU and GPU to host. An optimization is introduced for sequential implementation by using combination of lower level and higher level languages. In this work, multicore architecture is proven to be most efficient and accelerated for 3D CS reconstruction because of repetitive structure of algorithm. When compared to sequential CS recovery via Collaborative sparsity this design is energy-efficient, fast and has lower complexity.

1

Introduction

1.1 Background

Information processing is a dominant part in any field that uses present technologies. We are in the century of digital revolution that is known for the development of new computational and analytical tools that are being developed for the extraction of information from data. These tools are being neglected continuously because of large problem sizes dealt by today's applications. Thus, the challenge is to devise new and computationally efficient set of data processing tools that can adequately adapt to this gigantic set of information. Kotelnikov, Nyquist, Shannon, and Whittaker laid the conceptual foundation of this revolution by working on sampling of continuous time band-limited signals. The bottom line of their research was that "A signal can be reconstructed from its samples, if the sampling frequency is twice the highest frequency contained in the signal. '' As a result of their theories, a big area of signal processing has been transferred from analog to the digital domain. Digitization has enabled the creation of systems that are more robust and cheaper. Unfortunately, in some emerging applications, the resulting Nyquist rate is so high that it results in too many samples. Alternatively, it may be too costly. Thus, despite extraordinary advances the acquisition and processing of signals continues to pose a tremendous challenge. However, the computational complexity of the reconstruction of sparse signals is

quite high. Meanwhile, the computation of CS based imaging technique becomes larger and larger along with the increasing demand on high resolution images. Recently, the graphics processing unit (GPU) and multi-core CPUs have shown great potential for accelerating computations in many application areas, which offer an alternative for fast reconstruction of sparse signals. This thesis realized the fast reconstruction of CS based MR images, taking advantage of the efficient parallel computing capabilities of GPU and CPUs.

1.2 Compressed Sensing

Claude Shannon[1] formulated his theorem concerning data acquisition and sampling. Shannon- Nyquist sampling theorem [2] is a fundamental theorem in signal processing. According to this theorem, if we have correctly sampled measurements, a band limited signal can be recovered perfectly. But the recovery of the original signal is hopeless when the measurements are not enough to reconstruct the signal. As shown by the following figures:

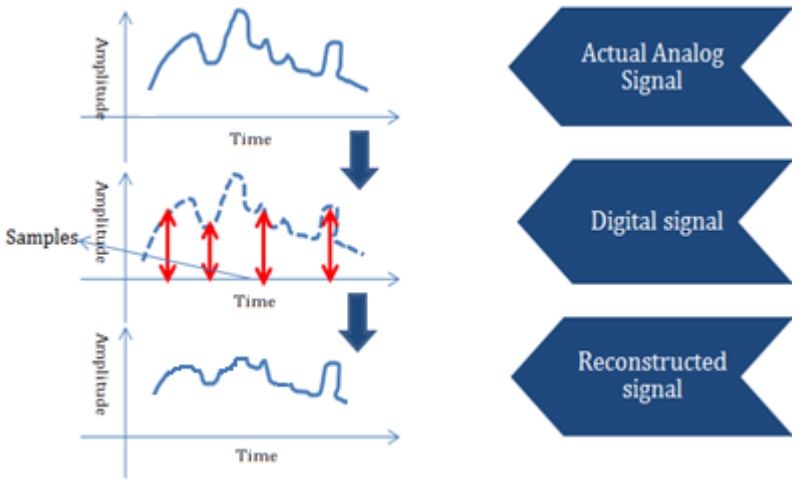


Figure 1.1 Unsuccessful Reconstruction

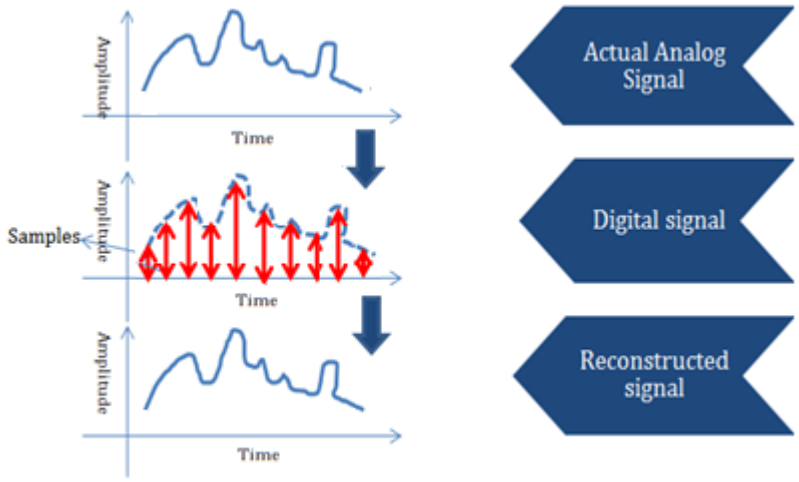


Figure 1.2 Successful Reconstruction

Since, this theory forces you to go the traditional way and to satisfy the Shannon-Nyquist theorem. Otherwise, we would not be able to reconstruct the original signal from samples of signal. Unfortunately this leads to very high amounts of data. The inefficiency of this technique is obvious. First one collects a massive amount of samples to discard nearly all of them in the compression step afterwards. So, one of the most intuitive thoughts would be to combine these both sampling and compression. Here, the new sensing paradigm, called Compressive Sensing comes into play. By simply combining two principles, the amount of samples needed for reconstruction can be reduced drastically. The process of CS with comparison of previous compression technique is illustrated in Figure 1.3 and 1.4.

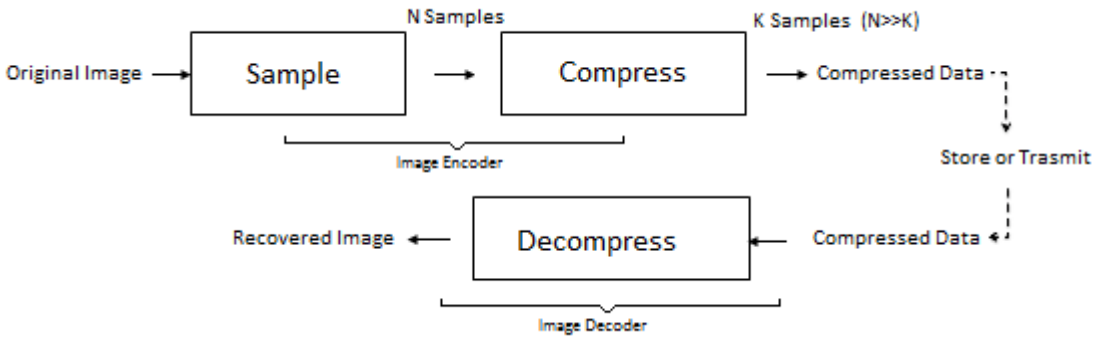


Figure 1.3 : Block Diagram of Previous Compression methods

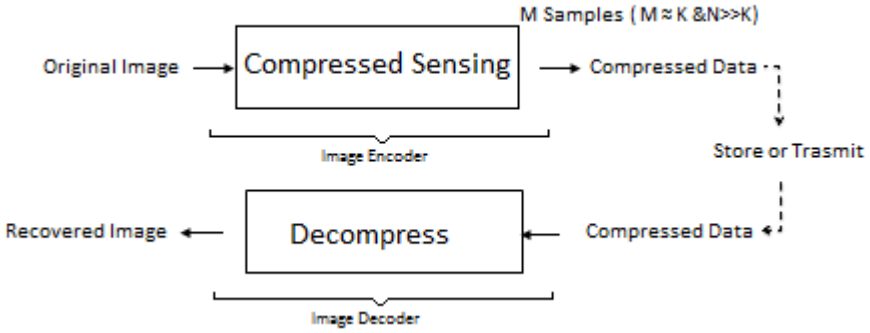


Figure 1.2 :Block Diagram of Compressed Sensing

Traditional paradigm is often extremely wasteful; huge data is acquired because of the Nyquist theorem, making compression a necessity prior to storage or transmission, and then discarded by a compression. In contrast, compressed sensing performs data collection and compression and reduces enormously the volume of data. Compressed sensing can recover accurate sparse signal from the linear measurements. An abstract concept of compressed sensing is shown in Figure 1.3 and theoretical detail is given below:

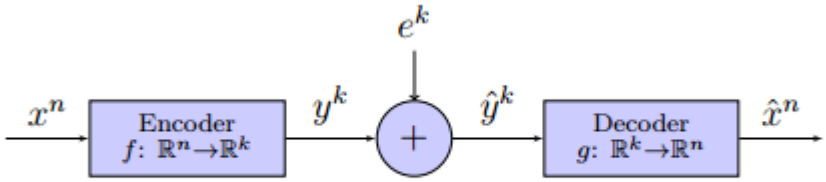


Figure 1.3 Illustration of Compressed Sensing Paradigm

Typically, image is reconstructed from the measurements that follow the Nyquist rate. By exploiting the sparsity of the image, CS can reconstruct the signal or image from the significantly reduced number of samples. Suppose the image or

signal is $x \in R^{Nx1}$ and exhibits sparsity in some transform domain Ψ. The signal x is K-sparse if

$$\|\boldsymbol{x}\|_{\boldsymbol{0}} = \#\{\boldsymbol{i}: \boldsymbol{x_i} \neq \boldsymbol{0}\} \leq \boldsymbol{K} \qquad \textbf{(1.1)}$$

More specifically, measurement vector $y \in R^{Nx1}$ is obtained as result of inner product of signal x and random projections $A \in R^{MxN}$. Mathematically; we represent these linear measurements as

$$\boldsymbol{y} = \boldsymbol{Ax} \qquad \textbf{(1.2)}$$

This process of obtaining linear measurements from signal x is called encoding. While the process of recovering sparse signal x from the linear measurements y is termed as decoding. The recovery of the signal from linear measurements is the ill posed problem given by Equation 1.2 but the sparsity makes recovery possible.

$$\textbf{min}_{\boldsymbol{u}} \|\boldsymbol{\Psi^T x}\|_{\boldsymbol{p}} \qquad \textbf{s.t.} \;\; \boldsymbol{y} = \boldsymbol{Ax} \qquad \textbf{(1.3)}$$

Where p characterizes the sparsity of the signal x and normally set to 1 for l_1 minimization and set to 0 for l_0 minimization. Recovery algorithm in [3] tries to solve this nonlinear optimization problem for 2D natural images. However, the algorithm considered here targets the 3D MRI.

1.3 Parallel Programming

Traditionally, software architectures are based on serial computations. A problem is broken into a discrete series of instructions. Instructions are executed sequentially one after another on a single processor. Only one instruction can be executed at any moment in time. With the evolution of simulation algorithms and extensively increasing data spaces and the limitations imposed at transistor level

on CPU processing speeds, parallel computing has emerged as a prevalent method to provide good performance metrics, especially for graphical processing applications.

1.4 Magnetic Resonance Imaging

Magnetic resonance imaging is a medical imaging technique that provides a way to quantify the water molecule diffusion in biological tissues. It is one of the most popular imaging modalities due to its excellent depiction of soft tissues, and inherent absence of emitted ionizing radiation. The traditional approach of MRI data acquisition is to sample at Nyquist rate followed by use of coding methods for compression. Recent trends have advanced to 3D-MRI and generally require faster acquisition techniques to achieve clinical practicality.

1.5 Research Objectives

The objectives expected to be achieved in this research endeavor are:

- Exploration of compression techniques.
- Implementation of sequential 3D compressed sensing algorithm.
- Parallelization of compressed sensing algorithm on different cores of processors.
- Implementation of compressed sensing algorithm on graphical processing unit (GPU).
- An optimized implementation of CS algorithm using mex files.
- Comparison of parallel, GPU based implementation and sequentially implemented CS algorithm.

1.6 Methodology

In this thesis, an algorithm for CS recovery via collaborative sparsity for 3D MRI compression is implemented. A conceptual methodology flow chart is shown in Figure 1.4. After comprehensive literature review, a 3D algorithm is implemented for CS. Then time for sequential algorithm is computed and algorithm is analyzed on the basis of various assessment parameters. Later, his sequential algorithm is parallelized using different processors and GPUs. Finally, in the end the performance of proposed 3D sequential algorithm is compared with the parallelized and GPU based algorithms.

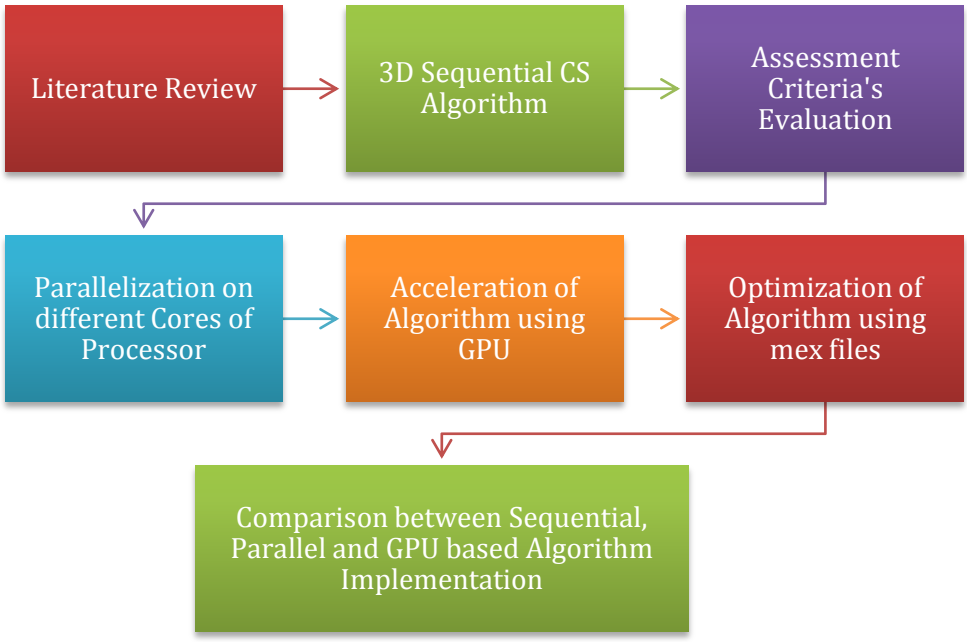


Figure 1.4: Methodology Flow Chart

1.7 Contributions

The primary essence of this work is to provide a low-complexity, low-cost, energy efficient encoding-decoding system for MRI, while overcoming the processing speed issues. In MRI processing, compressed sensing is used for data acquisition and then recovery of images. For CS-based imaging, the aim is to reduce the transform data in both domains and then decoding.

Specifically the contributions are summarized as follows:

- Local 2D and non-local 3D Sparsity: The Major challenge in compressed sensing theory is to find a domain in which signal exhibits maximum degree of sparsity. In this Algorithm signal sparsity is exploited in adaptive hybrid space transform domain. This method is discussed in Chapter 3.
- Efficient Augmented Lagrangian based technique: This transform domain result in severely undetermined inverse problem. Furthermore, when Augmented Lagrangian based technique used it ensures an enhanced reconstruction of image data in the form of 3D matrix. The idea behind the use of CS is to drastically reduce the transform coefficients. The simulation platform is built and performance of the system is shown in terms of signal-to-noise ratio.
- 3D Visualization: Above 3D matrix is visualized using orthogonal plane 2D texture mapping technique. It is a volume render for 3D data in openGL.

- Parallelization of Sequential Algorithm: Above sequential algorithm is computationally complex and time taking so it is parallelized on different cores of processors and GPU. These features aim at providing reduced complexity and increase the speed, which is one of the major issues in MRI reconstruction. This issue is dealt with in this chapter and results are compared with the existing sequential algorithm.
- The outcome of the research from this thesis has generated one journal manuscript.

1.8 Organization of the Thesis

This thesis comprises of seven chapters. The brief outline of each chapter is discussed as follows:

Chapter 1 --- Introduction

In this chapter, the background and the area of research are classified. The objectives, scope and methodology used in the research are documented. A systematic outline of the report is given at the end of the chapter.

Chapter 2 --- Literature Review

A comprehensive summary of the literature study is given in this chapter. Identification of challenges from the literature survey is also presented as the driver for the research.

Chapter 3 --- 3D Compressed Sensing Algorithm

In this chapter, sequential algorithm for compressed sensing with application to 3D MRI is discussed. The discussion encompasses parametric comparisons. Results are evaluated on the basis of these parameters for assessment.

Chapter 4 --- GPU based implementation of 3D Compressed Sensing algorithm

This algorithm is implemented on GPU but no speed up is gained. Results are shown at the end of chapter to check the efficiency of GPU based implementation.

Chapter 5 --- Optimization of Compressed Sensing Algorithm

This chapter is about optimization of compressed sensing algorithm. Optimization is introduced by using MATLAB's mex files.

Chapter 6 --- Parallel 3D Compressed Sensing Algorithm

In this chapter, the algorithm is parallelized on different cores of processors to make it computationally cost effective. This chapter also discusses 3D compressed sensing algorithm parallelization on different nodes of cluster so that minimum time and computational complexity is faced. Based on the computational and time complexity sequential and parallel algorithms are compared. After the analysis, one optimal algorithm, which is most accelerated, is recommended.

Chapter 7 --- Conclusions and Future Work

Conclusions from the current research presented in dissertation are derived and recommendations for future line of action are laid down in this chapter.

2

Literature Review

2.1 Background

A comprehensive summary of the efforts deployed round the globe for the implementation of fast and efficient compressed sensing algorithm is discussed. A review of existing literature was performed to support the methodology undertaken in this thesis. Next, a discussion of literature study associated with high performance computing is performed. Lastly, Identification of the missing links from the literature survey is also presented that gives the direction for the future avenues of the research.

2.2 Compressed Sensing Related efforts

The initial work for the understanding of reconstruction of an object from incomplete frequency samples can be attributed to Emmanuel Candès, Terence Tao and David Donoho. They introduced the compressed sensing by conducting a thorough research about the discovery of the signal using data less than Shannon Nyquist's criteria [4-6]. In compressed sensing, one adds the constraint of sparsity, allowing only solution which has a small number of nonzero coefficients. However, if there is a unique sparse solution to the underdetermined system, then the compressed sensing framework allows the recovery of that solution. So here the challenge is to seek a domain in which

signal has high degree of sparsity. A sparse signal is encoded in the form of some linear measurements and decoded from the incoherent measurement by solving the underdetermined linear equations. Direct solution of this linear system is non-deterministic polynomial-time(NP) hard problem because of exponential complexity of computations[7].Many solutions were proposed for the approximation of this NP-hard problem. Most common method applied for compressed sensing is using l_0 and l_1 equivalence [8]. Donoho proved that in CS theory reconstruction is possible by minimizing the number of non-zero elements (l_0 norm) assuming MRI is compressible in some transform domain. But the issue is that it is discontinuous and an NP-hard (Non-deterministic Polynomial-time hard) problem. Candès et al., showed that l_1 norm is the optimal convex approximation of l_0 norm. This equivalence result allows one to solve the l_1 norm problem, which is easier than the l_0 norm problem and has been proved that for many problems it is probable that the l_1 norm is equivalent to the l_0 norm in a technical sense. Researchers also introduced some greedy methods that could handle CS problem [9-13]. But l_1 and l_0 based methods provide better stability and require comparatively less measurements than greedy algorithms for computation of signal support.

Over the recent years, various CS recovery methods have been proposed. These methods can be categorized in three ways. In first category, algorithms are based on a range of sparsity bases using a subset of the largest transform coefficients for exact signal reconstruction. For example, fourier, Wavelet, Curvelet, Walsh and Discrete Cosine etc. All conventional MRI based processing rely on the Fourier transform for data acquisition, including 3D and dynamic MRI

[14-16]. In many instances, it is noticed that the Fourier encoding is not well suited for compressed sensing recovery, the incoherent conditions is only weakly satisfied with respect to sparse transforms [17]. This shows that the use of matrices other than the Fourier transform could possibly lead to better results. For example, wavelet transform in a coarse scale has its energy concentrated rather than spread out in the Fourier domain, which suggests the incoherence condition is barely satisfied [18]. Majumdar and Ward [19] used the wavelet domain for reconstruction of T1- or T2- weighted images. He used the images having the same cross section for high correlation and exploited the spatial correlation through wavelet transform. But to reconstruct the perfect signal one has to compute all the coefficients. As a result the algorithm requires more computations. Zhen [20] proposed a novel 3D Walsh transform and made an effort to experimentally compare 2D wavelet transform with 3D Walsh transform. 3D Walsh transform was preferred because it reconstructed image with better quality and it was computationally efficient too. In [21] optimal orthogonal wavelet approximation was used for adaptive compressed sensing based recovery. This algorithm was based on greedy approach for fast computation but the quality of the reconstructed image was affected.

Xiaobo et al.,[22] compared the traditional wavelet with Contourlet transform domain for compressed sensing recovery. He showed that wavelet does not sparsely represent MRI curves but Contourlet can overcome this shortage and reduce pseudo-Gibbs phenomenon. He concluded that medical images like MR images are not sparse in one domain only. We can combine more domains to sparsely represent the different structure in images. For example wavelet

transforms can provide suitable representation of point like feature in an image and line like features can be more sparsely represented by curvelet transform. In [23] combined wavelet and the curvelet based scheme was developed for two different frames. An algorithm for fast reconstruction using split Bregman technique was proposed. But the algorithm was dependent on the values of regularization parameter. So choice of better regularization parameters can lead us to better results. Some researchers proposed data adaptive sparsity transform based reconstruction algorithms also. M Hong et al., [24] experimentally explained the singular value decomposition (SVD) based transform as data adaptive sparsity basis and evaluated it for reconstruction of multiple MR images types. It was found useful for speed sensitive and computationally heavy application, such as cardiac MR images.

In second category, algorithms are based on geometric similarities and exploit the sparsity in spatial domain. Total variation based sparsifying transforms are well known for their property of preserving edges and boundaries [25]. But most of the time total variation based algorithms recover over-smooth image. To deal with this undesirable artifact many improved versions of total variation are designed [26-28] . Gaussian mixture models based algorithms hold an important role in compressed sensing. Guoshen and Sapiro proposed a novel statistical compressed sensing algorithm. Their algorithm could efficiently deal with a collection of images unlike the conventional compressed sensing algorithms which deal with one image only at a time. Moreover, this model revealed the broad range of possible algorithms which can be generated for improved compressed sensing reconstruction [29]. In [30] a

GMM-based compressed sensing algorithm reconstructed the images by estimating the Gaussian components but the computations grow with the image dimensionality. Moreover, it lacked the performance guarantees. Jianbo et al., addressed these drawbacks in [31] and developed a method to estimate the unknown GMM, unlike the traditional GMM based algorithms, for recovery of images using linear measurements with incomplete information. Many adaptive dictionary based algorithms are developed. These methods exploit the structural similarity using machine learning techniques. This approach produced optimal effective dictionaries at the cost of complexity. The sparse dictionary learning from corrupted data was first introduced by Ron et al., in [32]. He presented an efficient trainable dictionary which outperformed for denoising of corrupted data of CT scan. The proposed dictionary had the potential to be used for compressed sensing as well. In [33] a dictionary learning based compressed sensing technique was developed. This algorithm recovered image with enhanced PSNR values in some cases and it still had the improvement capability using the mini batch modification [34]. Yue Huang et al., presented a method for MRI recovery using the Bayesian approach that uses nonparametric dictionary to learn a sparse basis and total variation penalty term for smoothness. In [35] Ni et al., designed a two stage algorithm to get the sparser representation of signal and enhanced accuracy of reconstruction with the approximate single value decomposition for the dictionary updating. But still it had 'block' effect as a drawback, which is very visible at low sampling rate. The research of Yaghoobi in [36] approximated an audio signal using a novel compressible dictionary based model. But dictionary and recovery related many investigations were left to be

revealed. First category faced features dependent sparsity as a drawback. Neither of basis transform could present whole image features sparsely. The second category algorithms are computationally complex and thus limiting the reconstruction quality.

In third category, algorithms are designed by combining sparsifying transforms for assistance of one another. Recently, these types of algorithms are gaining interest. In [37] Qu, Xiaobo, et al. compared the performance of algorithm using combined transforms and single transform. His research showed that algorithm with combined transforms outperformed because plentiful features were sparsely represented for signal reconstruction guidance. The research in [38] proposed an algorithm that exploits local sparsity by total variation and two non-local sparsities by wavelet transform and patch correlation. This combined sparsity measure significantly reduced the sampling rate required for image compressive sensing. [39] is one of the state of the art algorithms, it uses the combined benefits of wavelet and total variation for sparsity of MRI. There are many solutions for convex optimization problem generated as a result of compressed sensing encoding phase. In [40] almost an optimal CS recovery is achieved through linear programming. Daubechies et al. in [41] applied iterative thresholding based iterative technique for solution of linear inverse problems.

MRI has been accepted as a versatile and powerful medical diagnostic tool over the past few years. However, MRI suffers from lengthy image processing time. For instance, encoding and decoding may take several hours for MR images acquisition. Compressed sensing approach has claimed to reconstruct a good quality image from very few measurements. Thus, fast reconstruction is the key

for adoption of compressed sensing approaches to MRI. But imaging speed remains a major challenge for volumetric MRI data collection and encoding through CS [42, 43] . Three dimensional image processing is a new and emerging field so has enormous applications. However tremendous data of 3D images slower down the data processing speed. So, 3D-MRI processing demands the development of new methodologies for fast clinical imaging practice. Compressed sensing brought new algorithmic and computational challenges. What makes the problem of fast reconstruction so challenging is that a compressed sensing reconstruction solves a large-scale non-linear optimization. This requires the use of iterative schemes. Every iteration involves operations which are computationally intensive. In addition, the number of iterations required is a polynomial function of the size of the problems with larger sizes taking many more iterations to converge. Though compressed sensing algorithms are very advantageous but with computationally intensive and complex reconstruction [44]. Therefore, it is critical to develop some methods for fast processing. Recently, several algorithms have been proposed which show trade-off between complexity and accuracy through light weight operations [45], [46] and [47]. In recent years, many reduced sampling methods are emerging based on multiple receiver channels [48], spatial and temporal correlations [49] and data redundancy [50]. Most of these algorithms are not clinically viable. Various studies revealed that the performance of the medical image processing algorithms can be enhanced by implementing them on graphical processing units (GPUs) [51, 52], multicore central processing units (CPUs) [53, 54] and field-programmable gate array (FPGA)[55, 56] which showed a remarkable advance

towards clinical feasibility. Qiyue accelerated already proposed time consuming compressed sensing algorithm. This approach improved the quality of the image and reduces the computation time by parallelizing it on multicore processors. It is hardly possible to accelerate an algorithm by the factor larger than the number of cores. He addressed this limitation through a preprocessing mechanism which avoid the smooth patches and sort the wavelet coefficients for to reduce the total computation time in [57]. In [58] a parallel compressed sensing algorithm is compared with sequential compressed sensing algorithm to demonstrate the significant speed up on parallel architecture using multi core CPUs and GPU. CPU cores speed up computations by the factor of 2.2 and GPU speed up of 35. The parallel algorithm implemented in [59] discusses the pros and cons of three parallel architectures. This implementation was also an effective approach for speed up of compressed sensing reconstruction. Daehyun et al. also showed in [42] that optimized use of parallel architecture can even efficiently reduce the computational overhead for reconstruction of single 3D MRI volume. In [60] CS-based 3D MRI reconstruction using multicore CPU and GPU architectures is proposed that can achieve fast data acquisition. A novel implementation of CS on high performance architectures (FPGA and Multicore CPU) is presented in [61]. Many core CPUs can offer performances comparable to GPU [62]. However [63] showed that GPU for the applications that require irregular memory accesses may not be suitable. Conversely, FPGA may not be suitable for applications which have large and complex computational kernels that require double-precision floating point calculations due to limitations in silicon area. As a result, developers have to decide which architecture is suitable for their application

such that they can achieve the most performance enhancement. Again, though there are solutions for reduction in computation time, the desire for the least computational time needs to be ascertained.

2.3 Missing Links in Literature

As has been noted above, compressed sensing is an emerging and broad field of research. Compressed sensing has recently gained a very significant and authentic effect in MRI and other related fields. If we consider the current scenario then a point has nearly been reached in which physical and physiological effects limit the ability to just encode data faster. The traditional approach of MRI data acquisition is to sample at Nyquist rate followed by use of coding methods for compression. These limits have led to look for methods to reduce the amount of acquired data without degrading image quality and increase the energy efficiency. This research builds on the foundation of designing an algorithm that can encode 3D-MRI data faster to achieve clinical practicality.

3

Compressed Sensing for 3D MRI Recovery

3.1 Background

According to CS theory, main reason behind the successful recovery is the sparsity. So, sparsity of a signal play key role for lossless and successful signal reconstruction. The sparsity and recovery quality are directly related with each other. Therefore, seeking a highly sparse domain is most important question for CS recovery. Since, image's sparsity is feature dependent. So, there is no universal domain in which whole image is sparse. Most compressed sensing algorithms explore a set of fixed transform domains (i.e. wavelet) and that is why they are signal-independent, resulting in poor rate-distortion performance compared to the conventional coding techniques. Compressed sensing algorithm used in this thesis uses combined sparsity in a hybrid domain. In Section 3.2 a brief description on compressive sensing theory is provided. In Section 3.3 we discuss magnetic resonance imaging with respect to single 2-D slice and 3D MRI.

3.2 Compressed Sensing

3.2.1 Overview

Let a signal $\mathbf{s} \in R^N$ be our signal of interest which is real valued and sparse in some domain $\mathbf{\Psi}$. We consider a dictionary (measurement matrix) $\mathbf{\Phi} \in R^{MxN}$ (M<<N) with the vectors $\boldsymbol{\varphi_i} \in R^N$ as columns. We collect m linear measurements which can be viewed as inner product of signal of interest **s** and column vectors $\boldsymbol{\varphi_i}$ of measurement matrix $\mathbf{\Phi}$. The measurements $\mathbf{y} \in R^M$ of signal of interest **s** are given by the following equation.

$$\mathbf{y} = \mathbf{\Phi s} \tag{3.1}$$

Our basic goal is the recovery of the K-sparse signal $\mathbf{s} \in R^N$ from its measurement vector $\mathbf{y} \in R^M$. If M < N then we cannot solve the linear system for **s** by inverting the measurement matrix $\mathbf{\Phi}$. Equation 3.1 is an ill posed problem. But this system might still determine x uniquely under the additional condition that x is K-sparse.

3.2.2 Sparsity

The sparsity of the signal is the number of at most non-zero components in the coefficient vector $\mathbf{\Psi s}$ of some transform domain.

$$\|\boldsymbol{s}\|_{\mathbf{0}} = \#\{\mathbf{i} : \boldsymbol{s_i} \neq \mathbf{0}\} \tag{3.2}$$

If a signal is transformed in some other domain and the number of non-zero elements in Equation 3.2 is K for the signal, it is called K-sparse. Thus, if $\|s\|_0$ << m, **s** is a sparse signal.

Thus, CS recovery problem of $\mathbf{s}$ can be formulated as:

$$\mathbf{min}_s \|\mathbf{\Psi s}\|_p \qquad \mathbf{s.t.}\ \ \mathbf{y} = \mathbf{\Phi s} \tag{3.3}$$

where $\mathbf{\Phi}$ denotes the measurement matrix and p represents the sparsity of vector $\mathbf{\Psi s}$. The value of p is normally set to 0 or 1. If the value of p is set to 1, it is called l_1 norm and denoted by $\|*\|_1$. It adds all the absolute entries in coefficient vector $\mathbf{\Psi s}$. While l_0 norm is when value of p is set to 0 and it is denoted by $\|*\|_0$. It counts the nonzero values of the elements in coefficient vector $\mathbf{\Psi s}$.

3.3 CS recovery via Collaborative sparsity for a single 2D slice

A collaborative sparsity based CS algorithm for 2D natural images based uses two image priors, smoothness in local region and repetitiveness in nonlocal region. This algorithm designs a sparsity which is consistent with image priors. This thesis follows the same pattern for 3D MRI images. The advantage of using this 2D algorithm over other algorithms is that it uses combination of two sparsities. To obtain the concept of collaborative sparsity for images we adopt and briefly review theory from [64].

3.3.1 Collaborative Sparsity Measure

In this section an overview of CS algorithm for a single slice using collaborative sparsity is provided. By merging local 2D sparsity in space domain and nonlocal 3D sparsity in transform domain a new collaborative sparsity measure for high fidelity image CS recovery is established. This collaborative sparsity can be expressed as

$$\mathbf{CoSM}\ (\mathbf{s}) = \|\Psi_{L2D}\ \mathbf{s}\|_p + \alpha\ \|\Psi_{N3D}\ \mathbf{s}\|_q \tag{3.4}$$

with α as regularization parameter and p and q represents the sparsities with values from [0,1]. Ψ_{L2D} denotes to local smoothness prior and Ψ_{N3D} corresponds to nonlocal self-similarity prior. By using the horizontal and vertical finite difference operator based filter and putting the value of p to 1, local 2D sparsity Ψ_{L2D} in Equation 3.4 is given by

$$\|\Psi_{L2D}\,\mathbf{s}\|_1 = \|D\,\boldsymbol{s}\|_1 = \|D_V\,\boldsymbol{s}\|_1 + \|D_H\,\boldsymbol{s}\|_1 \qquad \textbf{(3.5)}$$

while nonlocal 3D sparsity Ψ_{N3D} in Equation 3.4 is achieved by the following four steps:

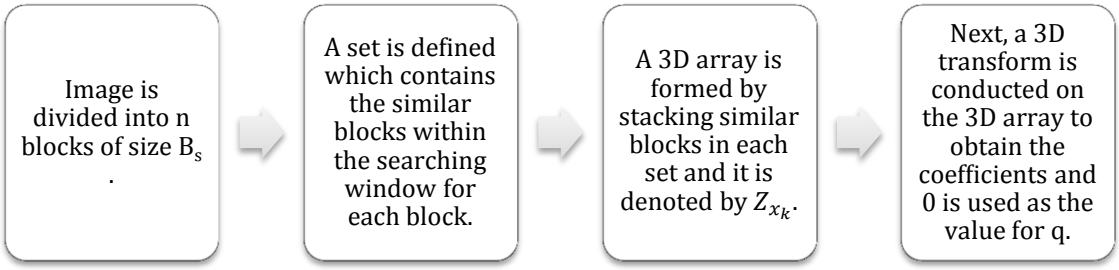


Figure 3.1 Illustration of nonlocal 3D sparsity

The following mathematical formulation of the nonlocal 3D sparsity Ψ_{N3D} is generated from all above illustrated steps in Figure 3.1.

$$\|\Psi_{N3D}\,\mathbf{s}\|_0 = \|\Theta_s\|_0 = \sum_{k=1}^{n}\left\|T^{3D}(Z_{s_k})\right\|_0 \qquad \textbf{(3.6)}$$

where $T^{3D}(Z_{s_k})$ represents the transform coefficients in nonlocal 3D transform domain and Θ_s is the column vector with the transform coefficients in lexicographic order. After dividing transform coefficients from vector Θ_s into n groups, groups are inverted for the estimation of each block. Estimated block are returned to their positions and by averaging these entire estimated blocks final image estimate is achieved. Final image estimate is given by

$$\hat{\mathbf{s}} = \Omega_{N3D}\,\Theta_s \qquad \textbf{(3.7)}$$

With the help of the above expressions, we can further rewrite Equation 3.4 as

$$\boldsymbol{CoSM}\ (\boldsymbol{s}) = \|\Psi_{L2D}\ \mathbf{s}\|_1 + \alpha\ \|\Psi_{N3D}\ \mathbf{s}\|_0\ =\ \|_D\ s\|_1 + \alpha\|\Theta_s\|_0 \qquad \textbf{(3.8)}$$

3.4 Recovery via Collaborative Sparsity Measure

The optimization problem for CS recovery can be described by

$$\min_s\|D\ \boldsymbol{s}\|_1\ +\ \alpha\|\Theta_s\|_0 \qquad s.t.\quad \mathbf{\Phi s} = \mathbf{y} \qquad \textbf{(3.9)}$$

Note that the solution of Equation 3.4 is quite difficult to solve directly due to the combinatorial computational complexity of this algorithm. Therefore, for computationally feasible algorithm auxiliary variables w and x are introduced and corresponding Augmented Lagrangian function led to our final step which is written as

$$\mathcal{L}_A\ (w, s, x) =\ \|w\|_1 -\ \nu^{\,T}(Ds - w)\ +\ \frac{\beta}{2}\|\ Ds - w\ \|_2^2 + \alpha\|\Theta_x\|_0\ -\gamma^T(s - x) +$$

$$\frac{\theta}{2}\|\ s - x\ \|_2^2 - \lambda^T(\Phi s - y) + \frac{\mu}{2}\|\ \Phi s - y\ \|_2^2 \qquad \textbf{(3.10)}$$

where β, θ, μ are regularization parameters and ν, γ, λ are Lagrangian multipliers. The augmented Lagrangian method seeks a saddles point by iteratively solving the Equations 3.11 and 3.12.

$$w_{k+1}, s_{k+1}, x_{k+1} =\ \min_{w,s,x}\mathcal{L}_A\ (w, s, x) \qquad \textbf{(3.11)}$$

$$\begin{cases} \nu_{k+1} =\ \nu_k -\ \beta\ (Ds_{k+1} - w_{k+1}) \\ \gamma_{k+1} = \gamma_k -\ \theta\ (s_{k+1} - x_{k+1}) \\ \lambda_{k+1} =\ \lambda_k -\ \mu\ (\Phi s_{k+1} - y) \end{cases} \qquad \textbf{(3.12)}$$

Equation 3.11 is still hard to solve. Therefore, for efficient solution this is further divided into three sub problems. For brevity, derivations for the solution of sub problems are excluded. Details of derivations of these sub problems and augmented Lagrangian method is presented in [64].

3.4.1 *'w'* sub problem for CS reconstruction

After simplifications *w* sub problem has the following closed form

$$\widetilde{w} = \max\left\{\left|Ds - \frac{\nu}{\beta}\right| - \frac{1}{\beta}, 0\right\}.\operatorname{sgn}\left(Ds - \frac{\nu}{\beta}\right) \quad \textbf{(3.13)}$$

3.4.2 *'s'* sub problem for CS reconstruction

The solution of s sub problem is obtained as

$$\tilde{s} = s - \eta\, d \quad \textbf{(3.14)}$$

where *d* is the gradient direction, η is the optimal step.

3.4.3 *'x'* sub problem for CS reconstruction

Thus, the efficient solution for *x* sub problem is derived as

$$\tilde{x} = \Omega_{N3D}\,\widetilde{\Theta}_x = \Omega_{N3D}\,(\text{hard}\,(\Theta_r, \sqrt{2\tau})) \quad \textbf{(3.15)}$$

Where $\widetilde{\Theta}_x = \text{hard}\,(\Theta_r, \sqrt{2\tau}) = \Theta_r.1\,(\,abs(\Theta_r) - \sqrt{2\tau}\,)$

3.5 CS recovery via Collaborative sparsity for 3D MRI

Recent trends have advanced to 3D-MRI. For that reason, realization of 3D-MR images is necessary in compressed sensing. We propose a method for slice by slice 3D reconstruction. For the reconstruction of complete 3D MRI algorithm outlined in Algorithm 1 is used. General block diagram of proposed slice by slice algorithm is shown below.

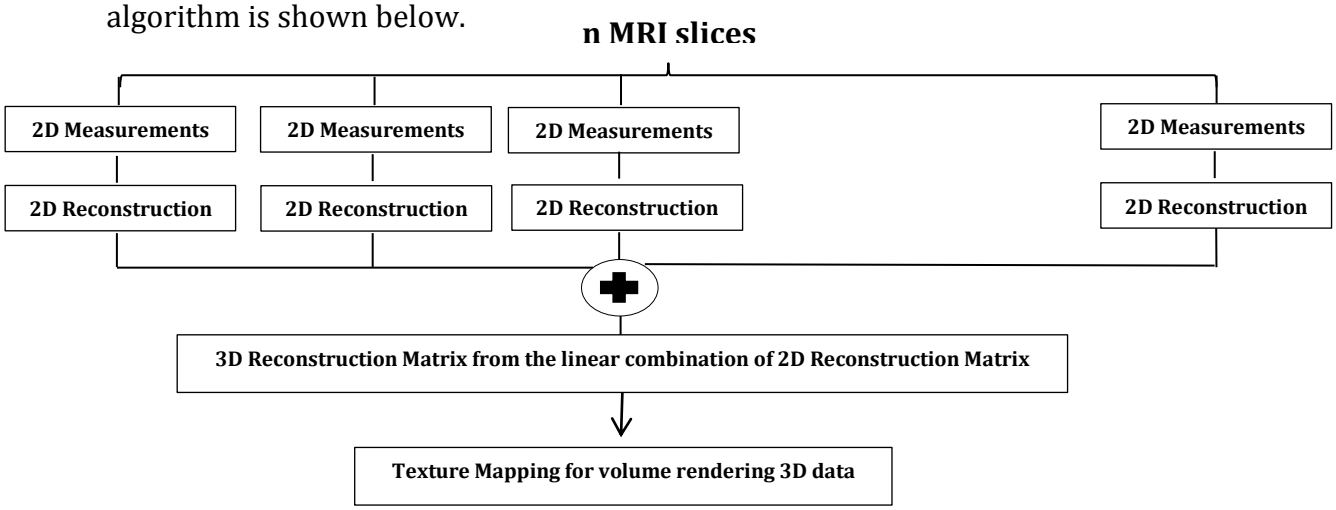


Figure 3.2: General system setup of Proposed 3D algorithm

The inputs are observed measurements $\mathbf{b_n}$ for all n slices, the measurement matrix **A**, regularization parameters and output is an approximation of 3D MR image signal. One of the main reasons of using collaborative sparsity based algorithm for each slice is that, it utilizes the image priors in local and nonlocal regions both. For high quality recovery results critical role is played by image priors. So, for more advantageous and effective solution a sparsity measure based on image priors is highly efficient.

Algorithm 1: Compressed Sensing Recovery for 3D MRI

Outer loop n **times**

$s_{no} = A^T b_n$, $v_{no} = \lambda_{no} = \gamma_{no} = 0$, $w_{no} = x_{no} = 0$ {Initialization for each slice}

Mid loop x **times**

Inner loop y **times**

Solve **w** sub problem by using the derived expression in Eq. 3.13

Solve **s** sub problem by using the derived expression in Eq. 3.14

Solve **x** sub problem by using the derived expression in Eq. 3.15

end

Update Lagrangian multipliers by using the derived expressions in Eq. 3.12

end

end

3D matrix is formed from restored n slices $\hat{\boldsymbol{s}}_{\boldsymbol{n}}$

3D MRI is formed by mapping the pixel values

3D reconstruction is achieved by obtaining random projections of the corresponding slice and reconstructing each image slice independently. These slices are dealt as bunch of frames. Image recovery is posed as a constraint convex optimization problem that could be transformed into unconstrained convex optimization problem with the collaborative sparsity based recursive algorithm. All independent frames are combined and 3D mapping of pixels is done by emerging the entire data ensemble from all the frames. As shown by block diagram of CS algorithm for 3D MRI. Next section deals with the simulation aspects of CS based image processing.

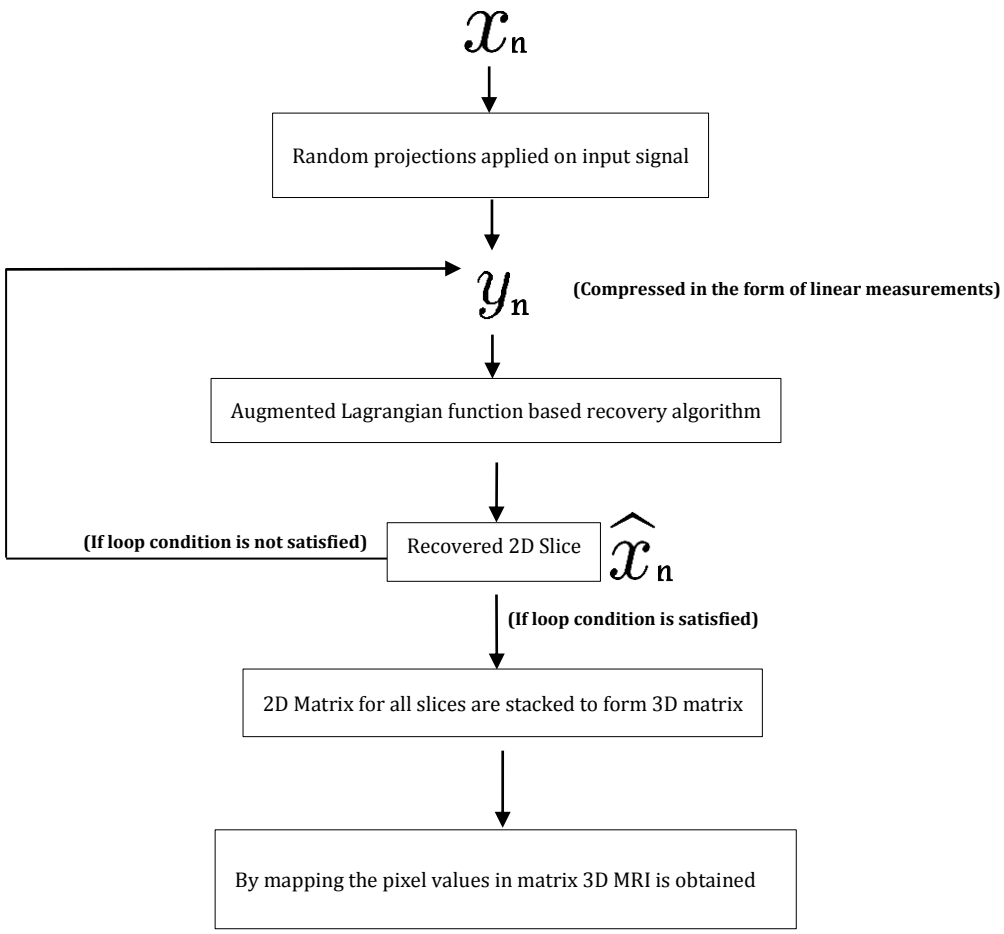


Figure 3.3 : Control Flow for 3D Recovery Algorithm

3.6 Result for 3D MRI

For the purpose simulations, dataset of 3D-MRI image used is of size 256x256x19. Dataset have been taken from [65]. For a fair comparison of our method with the collaborative sparsity based algorithm we compare the individual slices of 3D MRI reconstructed from our method and original 2D images. Different views of reconstructed 3D MRI are shown in Fig. 3.6. Though, the performance of slice wise PSNR of recovered 3D MRI is same as 2D image recovered from the collaborative sparsity based 2D algorithm. Experiments conducted using MRI data of knee demonstrate the performance of proposed 3D algorithm. The operating system for numerical experiments is 64bit Windows 7 with MATLAB version 2013B. The average processing time was computed over 285 repeated iterations.

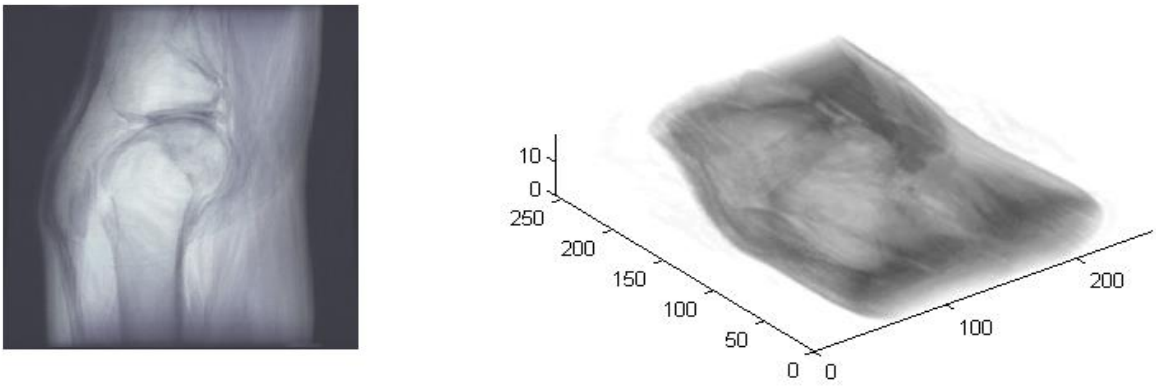


Figure 3.4 : Recovered 3D MRI in Grayscale

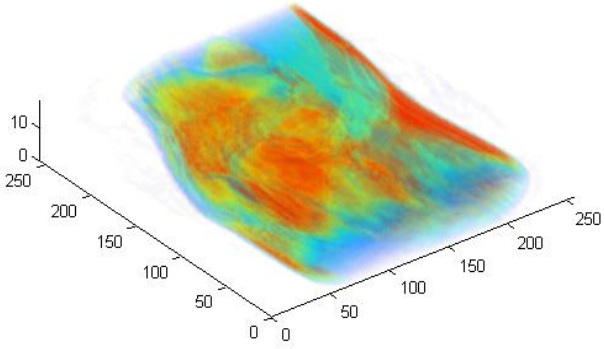


Figure 3.5 (a) : Recovered 3D MRI with opacity

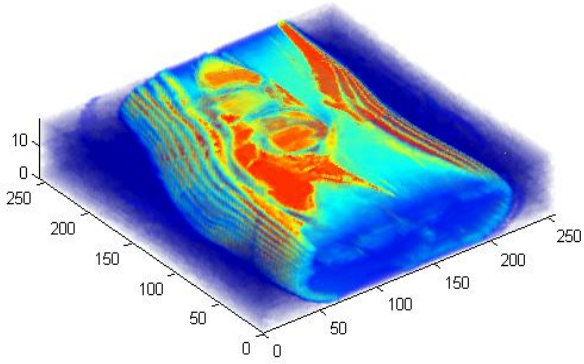


Figure 3.5 (b): Recovered 3D MRI with colors

3.7 Performance Analysis of 3D MRI

In this section, 3D-MRI processing using the 2D compressed sensing Recovery via Collaborative Sparsity Algorithm is evaluated on the basis of different assessment. The aim is to show the enhanced performance of 3D-MRI using 2D compressed sensing Recovery via Collaborative Sparsity Algorithm. A few slices are picked from reconstructed 3D MRI for evaluation. Different evaluation criteria are:

- Root Mean Square Error (RMSE)
- Histogram
- Peak Signal to Noise Ratio (PSNR)
- Topography Measure
- Relative Error (RE)
- Correlation Coefficients

The original and reconstructed slices chosen from 3D MRI for evaluation are shown below.

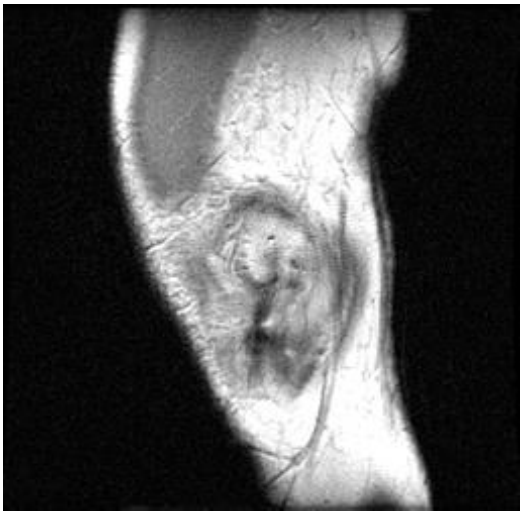


Figure 3.6 (a): Origional Slice Number 1

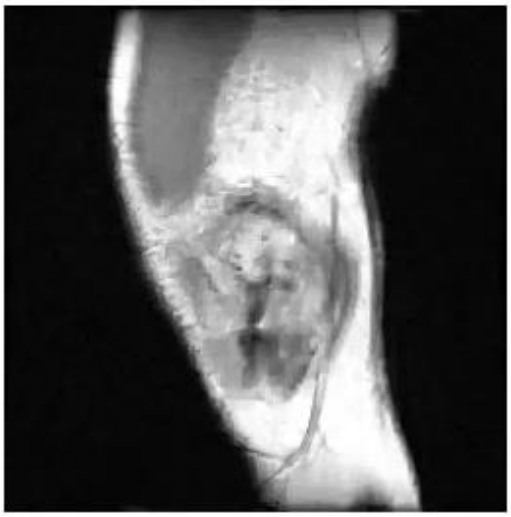


Figure 3.6 (b): Reconstructed Slice Number 1

Figure 3.7 (a): Origional Slice Number 6

Figure 3.7 (b): Reconstructed Slice Number 6

Figure 3.8 (a): Origional Slice Number 12

Figure 3.8 (b): Reconstructed Slice Number 12

Figure 3.9 (a): Origional Slice Number 18

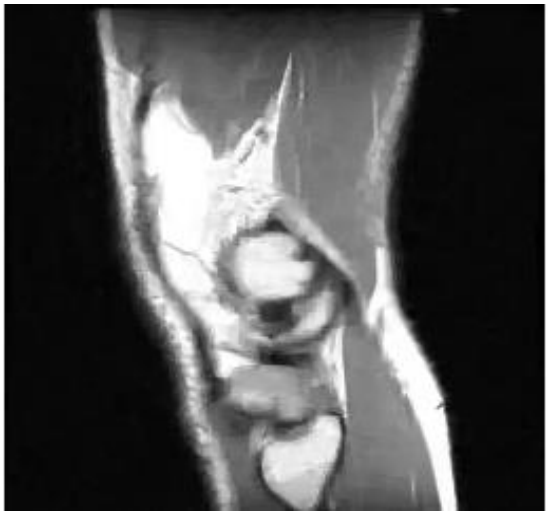


Figure 3.9 (b): Reconstructed Slice Number 18

3.7.1 Root Mean Square Error

The following table shows the root mean square error of 4 random slices in reconstructed 3D MRI for knee data. From Table 3-1 it is very clear that reconstructed 3D MRI have very less RMSE.

Table 3-1: Root Mean Square Error Values of Reconstructed Images

No.	Slice Number	Root Mean Square Error (RMSE)
1	1	7.9%
2	6	9.9%
3	12	11%
4	18	8%

3.7.2 Peak Signal to Noise Ratio

Table 3-2 contains the peak signal to noise ratio of 4 random slices in reconstructed 3D MRI from knee data. The PSNR of the reconstructed 3D MRI is relatively high.

Table 3-2: Peak Signal to Noise Ratio of Reconstructed Images

No.	Slice Number	Peak Signal to Noise Ratio(PSNR)
1	1	30.1
2	6	30
3	12	28.1
4	18	27

3.7.3 Topography Measure

Using topography measure, it is shown in Table 3-3 that reconstructed 3D MRI very resembles with original MRI in shape and features. Difference in topography measure of reconstructed and original MRI is very less.

Table 3-3: Topography Measure of Reconstructed Images

No.	Slice Number	Topography Measure
1	1	6%
2	6	7.5%
3	12	7.9%
4	18	7%

3.7.4 Relative Error

In Table 3-4, we observe that relative error in reconstructed and original MRI is negligible.

Table 3-4: Relative Error of Reconstructed Images

No.	Slice Number	Relative Error (RE)
1	1	6.6%
2	6	7.6%
3	12	7.9%
4	18	7%

3.7.5 Correlation Coefficients

Correlation coefficients in Table 3-5 show that there is close relation in reconstructed and original 3D MRI.

Table 3-5: Correlation Coefficients of Reconstructed Images

No.	Slice Number	Correlation Coefficients
1	1	7.9%
2	6	9.9%
3	12	11%
4	18	8%

3.7.6 Histogram

In histograms for original and reconstructed MRI show the similarity of images.

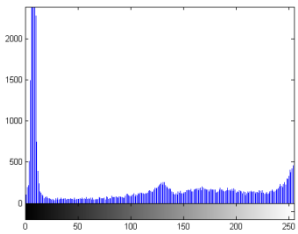


Figure 3.10 (a): Histogram of Original Slice 1

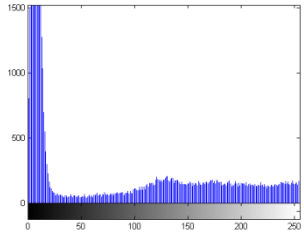


Figure 3.10 (b): Histogram of Recovered Slice 1

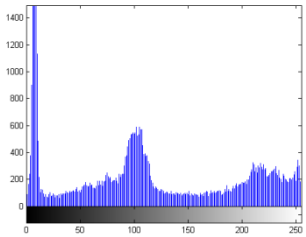


Figure 3.11 (a): Histogram of Original Slice 6

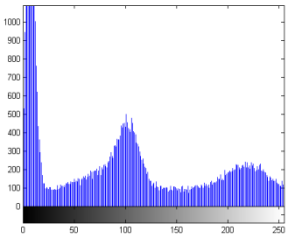


Figure 3.11 (b): Histogram of Reconstructed Slice 6

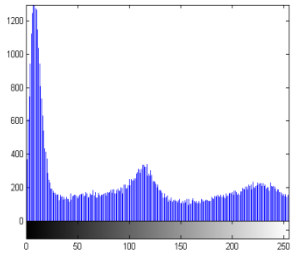


Figure 3.12 (a): Histogram of Original Slice 12

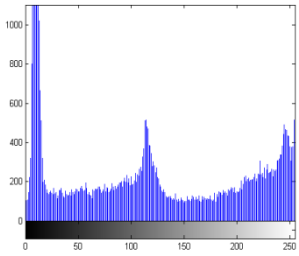


Figure 3.12 (b): Histogram of Reconstructed Slice 12

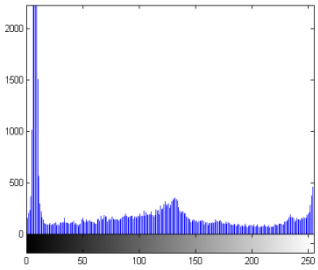


Figure 3.13 (a): Histogram of Original Slice 18

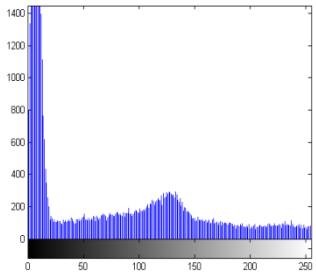


Figure 3.13 (b): Histogram of Reconstructed Slice 18

4

3D Compressed Sensing Algorithm on GPU

4.1 Background

High performance computing includes computers, networks, algorithms and environments to make the systems ranging from multicore PCs to fastest supercomputers usable. Other than reduction of data acquisition time for 3D-MRI, we also consider that our proposed methods should be physically implementable with low complexity and minimum hardware. In conventional methods, processing required for data acquisition is enormous due to its repetitive nature even for a single slice and leads to a slower system. The multi core based algorithm implementation is demonstrated in Chapter 5. Recently, our focus is on combining computing architectures along with GPUs to achieve maximum performance. In this chapter we will analyze GPU based implementation of this algorithm and the analysis procedures including speedup and efficiency (if any).

4.2 Algorithm overview w.r.t Computational complexity

In this section, the core framework of compression algorithm and execution flow is discussed. Moreover, the execution flow of algorithm is described. The generic framework of CS compression algorithm can be divided into two main blocks.

- Compressive Sampling (Encoder)
- Reconstruction (Decoder)

Attraction of CS based algorithms is due to the computationally inexpensive encoder at the cost of complex decoder. The main component in CS system that is computationally intensive is the decoding. Encoder conducts same random projections on all input signals and decoder has to recover signal in sparse domain which is exhaustive optimization problem. Flow chart of the most important steps in algorithm for recovery of 3D MRI is shown in the Figure 4.1.

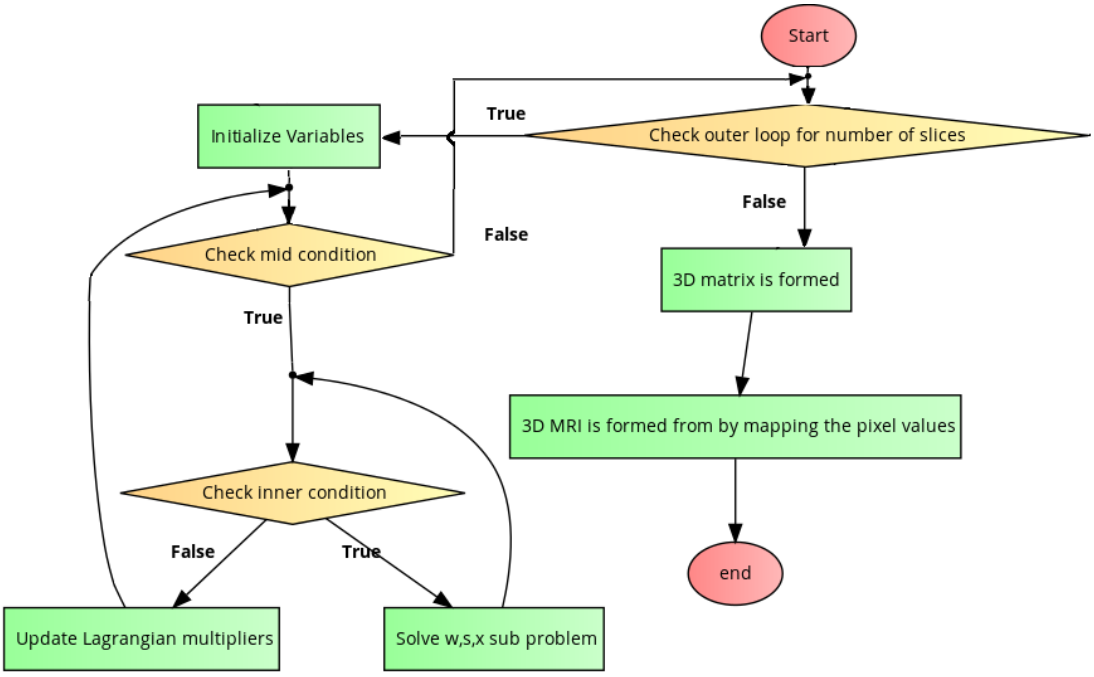


Figure 4.1 Flow chart of Collaborative Sparsity based recovery algorithm

The algorithm consists of three for loops, outer, mid and inner loop. Based on these loop structures we divide reconstruction algorithm into three stages. Initially, computing starts with stage1 at outer loop, this loop makes sure that all the slices of MRI are dealt for recovery purpose. The next step is stage2, mid loop iterates for solution of optimization problem by utilizing augmented Lagrangian function. Due to no differentiability, stage2 at stage3 is further divided into sub problems in inner loop. This loop solves these sub problems iteratively for efficient solution. The execution time for complete algorithm is shown in Figure 2.

Figure 4.2 Execution Time Division of Algorithm

It is clear from the chart that main component which is computationally intensive is the stage3 at calculation of sub problem **x**. Sub problem x is computationally complex because CPU consumes a lot of time in similar patch search, 3D matrix operations and 3D transforms. The main complexity comes from the number of 3D matrix operations required to obtain a single 2D slice of a

3D image. And, when 3D processing has to be undertaken, the processing increases N-fold. This also makes the MRI system bulkier. Sub problem **x** accounts for 99% of whole algorithm execution time. Thus, optimization requires reducing the acquisition time by providing parallel techniques on stage3. One possible way of reducing the time required for processing is by implementing stage3 of algorithm on GPU.

4.3 General Purpose Graphics Processing Units (GPGPUs)

Last few years demonstrate that GPUs are hardware architectures which are used for maximum performance in graphics applications. Now a day, GPUs are interpreted as massively parallel many core processors and fully programmable. GPU is used to process the computationally intensive in a massive-parallel manner; hence the speed of processing is increased. While the CPU works on sequential algorithms. The performance advantage provided by the use of graphics processing units makes this technology particularly fascinating for scientific applications.

The basis for using GPUs in parallel computing is:

- Distinguished throughput computation
- Maximum bandwidth memory
- Availability to all

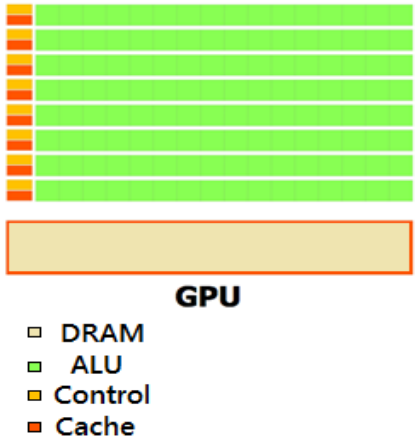


Figure 4.3 : General GPU Architecture

GPU systems use a combination of CPU and GPU to produce a co-processing model. Computationally intensive parts which need to be processed in a massively parallel manner are accelerated by the GPU in order to benefit from their high computing. The advantage of using GPU is in terms of hundreds of cores for massively parallel computations but does have high communication time for CPU to GPU and GPU to CPU and memory allocation. The computational time for GPU computation is utilized in four steps as shown in Figure 4.4. In the first step, memory is allocated on GPU for data used by computation on GPU. In the next step, data is transferred from host memory to GPU memory. Then computation is performed on GPU and data produced is transferred back to host memory. In the last step GPU memory is freed.

Figure 4.4 : Steps for GPU computation

4.4 GPU-based CS recovery for MRI

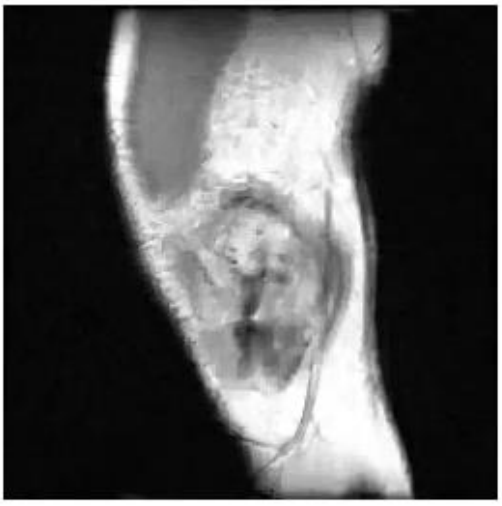
For GPU based CS recovery, open source CS recovery via collaborative sparsity code is chosen as solver. Jian Zhang's algorithm was originally implemented in MATLAB. This implementation is modified to work with GPU computing in MATLAB. For loops involved in stage3 are vectorized to use optimized libraries for matrix operations.

We use MATLAB® Parallel Computing Toolbox [66] to perform automatic optimization for GPU computing. From the beginning all the arithmetic

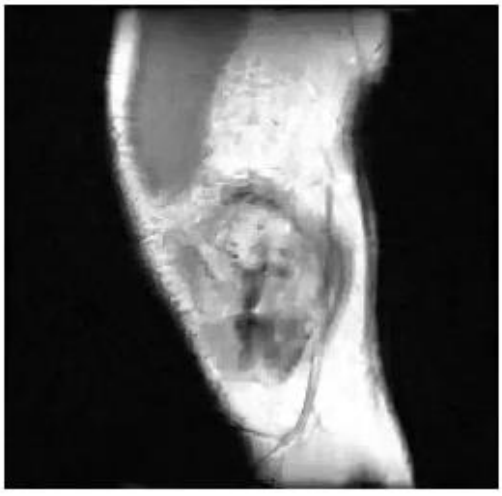
operations are performed on CPU because initially the operations are computationally simple. When control is moved to stage3, data is transferred on GPU through temporary memory allocation using MATLAB command *gpuArray.* MATLAB automatically transfers control on GPU, if an array lies in GPU memory, for all subsequent complex operations. After loop solution is transferred to CPU using MATLAB command *gather.* All temporary memory allocation on GPU is freed.

4.5 GPU-based Simulation for 3D MRI

In this section, GPU-based compressed sensing implementation for 3D MRI is compared with the sequential compressed sensing algorithm with application to 3D MRI illustrated in Chapter 3. The GPU used in our setup is TESLA T10 448 CUDA cores running at 1.2 GHz and has 1280 MB of off-chip global memory. The aim is to show the comparison of computational time and hence, the suitability of architecture for this collaborative sparsity based CS algorithm.



(a) Reconstructed Slice from Sequential implementation



(b) Reconstructed Slice from GPU based implementation

Figure 4.5: Reconstructed Images for a 256x256 single 2D slice of a first 3D dataset.

A single slice reconstructed from sequential implementation and slice reconstructed from GPU based implementation is shown above. The PSNR values for some slices are tabulated in Table 4.1. As depicted in the table, for this GPU based implementation obtained PSNR is same as that of sequential implementation.

Slice Number	GPU based CS Implementation (1x) PSNR(db)	Sequential CS Implementation (4x) PSNR(db)
1	30.1	30.1
6	30	30
12	28.1	28.1
18	27	27

Table 4-1: Comparison of PSNRS for GPU based and sequential implementation

Furthermore, for better analysis and comparison of GPU based CS implementation with sequential CS implementation computational time for reconstruction of whole 3D MRI is calculated. It shows that the computational time required for GPU based implementation is 4x slower when compared with sequential implementation because of repetitive calls to function used for solution of *sub problem u.*

While most work about GPU based CS implementation has emphasized the use of GPU but some algorithms are not appropriate for GPU based implementation due to irregular memory access [63]. In fact, very often communication from host to GPU and GPU to host can increase the communication overhead. This communication overhead limits the use of GPU in algorithms that have repetitive structure. By performing these simulations, we prove that GPU based CS implementation is computationally exhaustive as compared to sequential algorithm because of loops in algorithm. These repetitive calls to GPU result in computation overhead because of communication and memory allocation again and again. From all the above illustrations we can conclude that, the sequential CS implementation is superior to that of GPU based CS implementation for this algorithm.

5

An optimized 3D Compressed Sensing Algorithm

5.1 Background

In the previous chapter, a 3D compressed sensing algorithm was derived by using already proposed collaborative sparsity based recovery algorithm. But this recovery algorithm typically requires a significant computational effort for 3D MRI reconstruction. For fast recovery, it is necessary that an optimization should be defined for computationally complex areas of algorithm. This is widely known that low level languages are fast with respect to computation time. From an implementation point of view, it is time consuming to transfer whole MATLAB implementation which is a high level language into a low level C language. Therefore, using a combination of low level and high level language is highly desirable. This is possible by using mex files in MATLAB. The peculiar nature of mex files implies that some part of whole implementation can be rewritten in a low level C language and called in MATLAB. In this chapter, we deal with this challenge. We separate some complex part of existing implementation. These parts are rewritten in C language and converted into mex file. An optimized Implementation with its suitability and bounds is defined. However, this method lacks for quantifying and analyzing its performance.

5.2 Problem Formulation

In the previous chapter, quadratic penalty function and Lagrangian function were used for MRI data acquisition. This combination of quadratic penalty function and Lagrangian function is called augmented Lagrangian function. Currently, aim is to formulate an optimized CS recovery algorithm so that a fast MRI reconstruction can be performed. This also includes CS reconstruction without degrading the image quality. The collaborative sparsity based CS algorithm is same as outlined in section 3.5.

5.3 Mex files in MATLAB

In order to use mex files, we first present a brief concept of mex files. Mex is a short description of MATLAB executable. It is used for interfacing between MATLAB and codes written in low level languages (FORTRAN, C etc.) These codes use MATLAB data structures as input and produce data structures as output. Any MATLAB supported C compiler is installed and mex is configured to use that compiler. When these mex files compiled, are called within MATLAB environment just like MATLAB M-files.

5.4 Optimization by introducing mex files

Mex are used for interfacing of C codes into MATLAB codes. These executable are used for optimization of MATLAB codes. Complex MATLAB M-files are rewritten into C language so that we can obtain an optimized implementation. We use same sequential implementation and try to provide reduction in computation time.

5.4.1 Selection of complex functions

Since, sub problem ***x*** utilizes maximum of whole algorithm execution time. Thus, to reduce the acquisition time for sub problem ***x*** optimization is introduced in stage3. Nonlocal sparsity is explained in details in section 3.3.1.

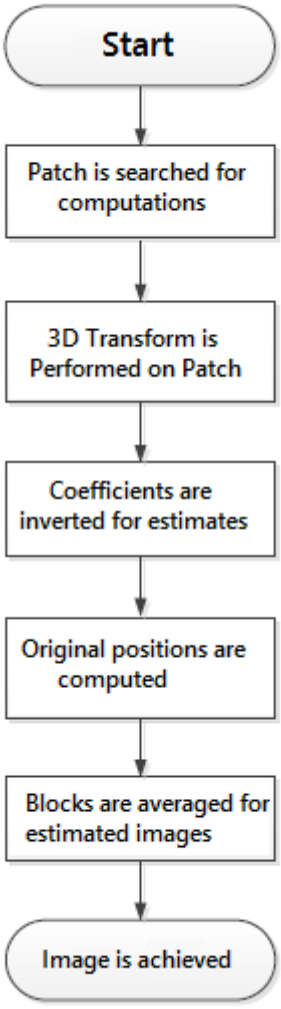


Figure 5.1 : Flow chart for solution of sub problem *u*

Sub problem ***x*** uses nonlocal sparsity for solution. The solution of sub problem ***x*** is calculated by using Equation 3.15 and mathematical expression is as follow

$$\tilde{x} = \Omega_{N3D}\, \widetilde{\Theta}_x = \Omega_{N3D}\, (\text{hard}\, (\Theta_r, \sqrt{2\tau}))$$

Some steps followed through the implementation for sub problem ***x*** are depicted in Figure 5.1. As observed the computational intense part is sub problem ***u***, which is a repetitive process through double loops. It is composed of three major components, two for core calculations, namely DWT2DCT for 3D transform and IDWT2DCT for inverse transform, and one for data formulation, namely PatchSearch.

5.4.2 M file to mex file conversion

These are very important processing steps in recovery algorithm. Hence, optimizing these components can benefit in terms of computational time. Above mentioned complex functions are rewritten into C language. Gate functions define inputs and outputs for these mex files. By adopting this implementation, we attempt to reduce the computational complexity and also speed up the 3D-MRI process.

5.5 Simulation Results

This section compares mex based implementation with the sequential compressed sensing algorithm illustrated in Chapter 3. In this chapter, our aim is to show the suitability of this optimized implementation for CS algorithm with application to 3D MRI. This implementation using combination of low level and high level language is competent to deal loop complexity. In start, this implementation works perfectly fine. Later, sorting algorithm used in C language could not compete with optimized sorting algorithm used in MATLAB. This

sorting algorithm is not able to sort arrays of each size efficiently. This mex based implementation obtained PSNR same as that of sequential implementation. This algorithm took 419ks for recovery of all slices using collaborative sparsity based compressed sensing algorithm. This computational time is 2.8x slower when compared with sequential implementation because of restriction of sorting algorithm in C language. Our simulation and synthesis results demonstrate that this implementation may be improved to process loops efficiently but complex sorting algorithm used lacks in performance.

6

3D Compressed Sensing Algorithm on Multi-cores

6.1 Background

Compressed Sensing recovery algorithms typically require a significant computational effort for the problem sizes occurring in most practical applications. While the computational complexity is a major issue for applications where 3D images are processed (e.g., in MRI), it becomes extremely challenging when high SNR is required. Hence, to meet the stringent reconstruction, parallelizing algorithms is of paramount importance.

6.2 Types of High Performance Computing Architectures

The widely used high performance computer architecture types are:

1. Shared Memory Architectures. (SMAs)
2. Distributed Memory Architectures. (DMAs)
3. Hybrid Distributed-Shared Memory Architectures. (HDSMAs)

6.2.1 Shared Memory Architectures

Common ability of shared memory architectures (SMA) is that all the memory is accessible by all processors. At the same time more than one processor can share the memory resources while operating independently.

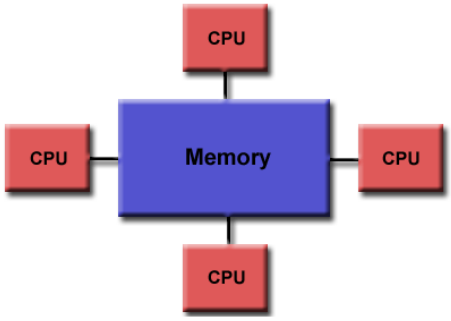


Figure 6.1 : Shared Memory Architecture

On the basis of memory access time it is further divided into following types:

- Uniform Memory Access (UMA)
- Non Uniform Memory Access (NUMA)

In uniform memory access, all processors share the main physical memory. Updates in shared memory locations by any of the processors are visible to every processor. When two or more symmetric multiprocessors are physically linked, it is non-uniform memory access. In this case, updates in shared memory locations by any of the processors are not visible to every processor. The advantage of SMAs is that the sharing of data is very fast and programming approach is very user friendly. The disadvantage of shared memory architectures is that if we add more CPUs then the synchronization constructs make programmer's job tough.

6.2.2 Distributed Memory Architectures (DMA)

In Distributed Memory Architectures (DMA) is about the local memory of every processor which is not shared by any other processor. Inter processor memory is used to connect through a communication network in

DMA. It is programmer's task to explicitly define how and when data is communicated.

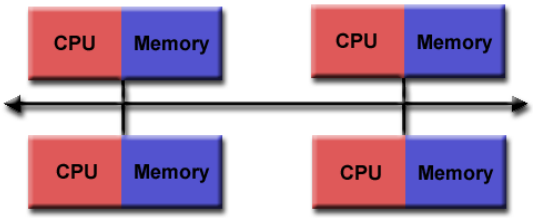


Figure 6.2: Distributed Memory Architecture

The major advantage of using DMAs is memory scalability with number of processors. The main problem is that it is programmer's responsibility to define all the data communication between processors.

6.2.3 Hybrid Distributed-Shared Memory Architectures (HDSM)

Hybrid distributed-shared memory architectures secure the services of both SMA and DMA. In HDSMA, processors on that machine can address that machine's memory as global. The distributed memory component is the network of multiple SMPs. Therefore, a communication through network is required to transfer data from one SMP to another SMP machine on the network.

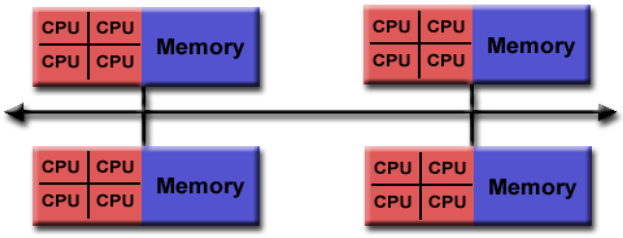


Figure 6.3 : Hybrid Distributed Shared Memory Architecture

6.3 Parallel Programming Method in MATLAB

MATLAB® Parallel Computing Toolbox [66] solves computationally complex and huge data problems by parallelizing them on multicore processors. High-level constructs – parallel for-loops and well-developed MPI integrated libraries are available that can be used in parallelizing MATLAB applications. In this section we will look at how this parallelizing method executed in MATLAB influence the speedup and efficiency of our compression code.

6.3.1 Parallel for loop

A parallel for-loop [67] is an easy way to divide independent loop iterations of intensive computation among different workers. The algorithm using "parfor" loop is parallelized by running it on client that divide up the task among worker and gather the results. Parallel implementation using "parfor" is similar to OpenMP. The figure below illustrates the working of "parfor" loop for possible number of cores/workers.

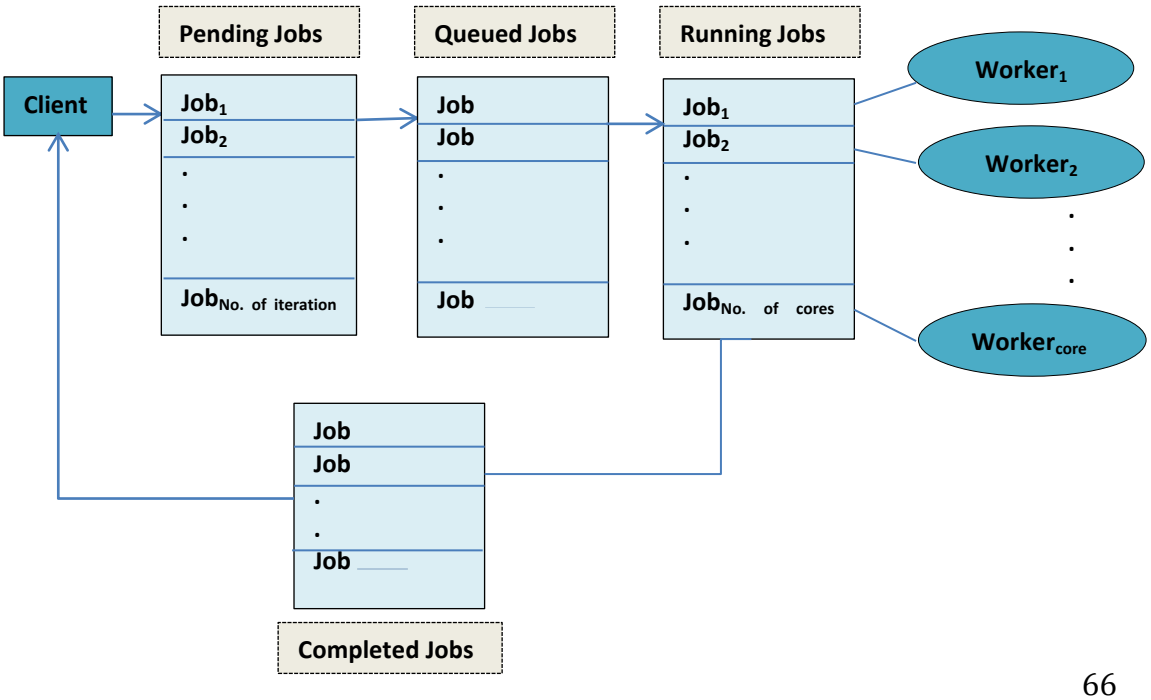


Figure 6.4 : Working of parallel for loop

The “parfor” loop automatically detects workers and exchange code and data between client and workers. It divides the task by allocating iterations among multiple workers. The only requirement for distributing execution using “parfor” is that iterations must be independent of each other, and no communication can occur between workers during the execution of the loop. Compressed sensing algorithm using 2D multiple slices is computationally intense because it iterate the independent 2D algorithm for a 3D reconstruction. So, to parallelize this algorithm it is desirable to distribute independent loop among workers for computation. This CS algorithm is an independent implementation of 2D multiple slices to form complete 3D MRI.

6.4 Parallelizing Compressed Sensing Algorithm

Compressed sensing is accepted as a powerful technique in computer vision and image compression for compression of images having huge information and extraneous data but it has not been adopted as much due to its high computational cost and storage complexity.

After considering the performance blockages of the 3D compressed sensing algorithm on a single-core processor, a multi-core and multi-node based 3D CS implementation was done and results were analyzed. The performance of 3D CS improved significantly because of the introduction of parallel computing technique.

6.5 Speed-Up Results

The easiest code to parallelize in MATLAB using all cores of a CPU is through the 'par-for' loops. Within the par-for loop each iteration is independent of all others, and the MATLAB built in scheduler, portions-out each iteration to a worker for computation. The results are then collected and returned appropriately by the scheduler. A comparison and analysis of the potential speedups achieved between one core and multi core implementation using above mentioned techniques is given below. Table 1 shows the algorithm execution time for the outer main loop through sequential and parallel versions of 3D compressed sensing:

Table 6-1 : Serial and parallel execution times for 3D MRI data on a multicore machine

	Sequential Implementation	**Number of MATLAB workers**			
		2	**4**	**6**	**8**
CPU Runtime(ks)	152	39	27	24	20
CPU Speedup (x)	1	3.8	5.6	6.3	7.6

The reduced computational runtime and speedup achieved by distributing the voting calculation on 2, 4, 6 and 8 processors of a single CPU is presented below:

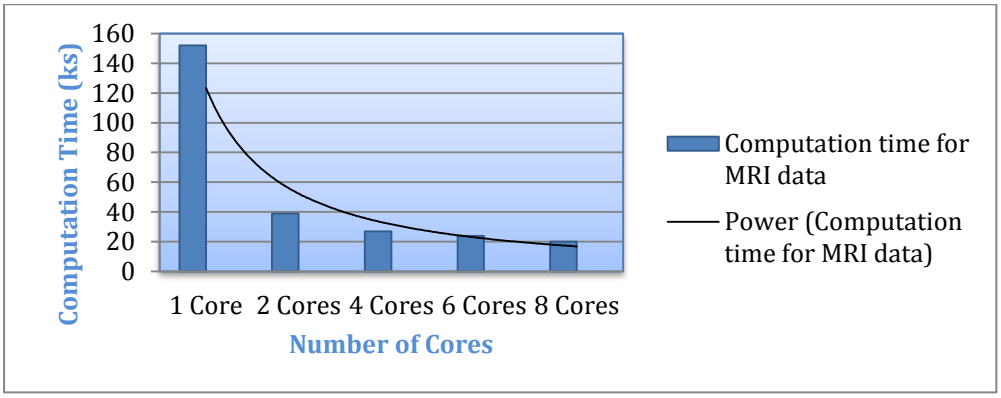


Figure 6.5 : Parallel Runtime of 3D MRI data recovery algorithm on an 8-core Machine

Figure 6.5 shows parallel runtime of 3D MRI reconstruction on an 8-core machine. It is clear that the runtime is reduced by increasing the number of processors.

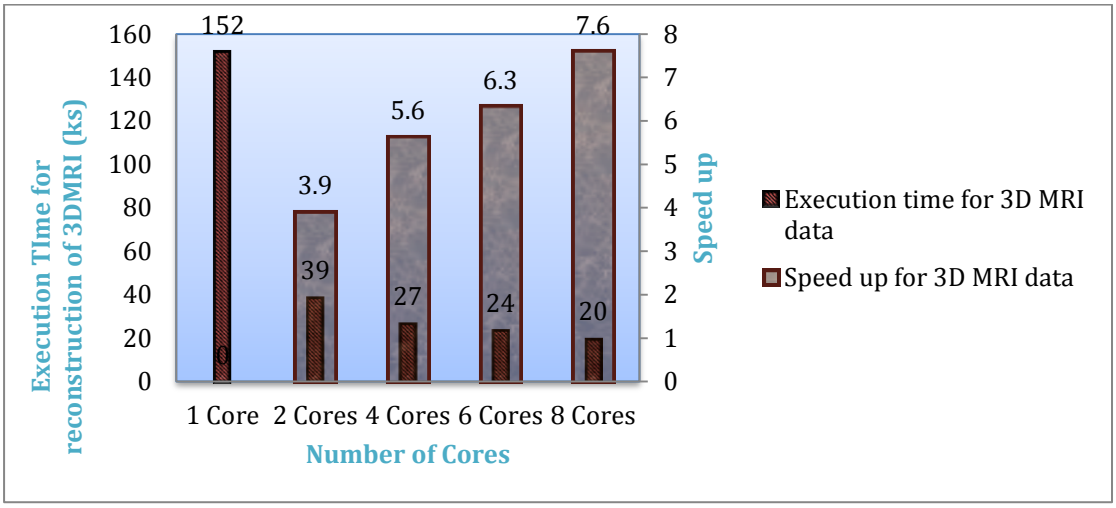


Figure 6.6 : Parallel Speed up of 3D MRI reconstruction on 8-core Machine

Our parallel implementation of compressed sensing algorithm reduces the runtime of the program to 7.6 times the sequential one on an 8-core machine. However, the speedup of parallel implementation of compressed sensing algorithm as presented in Figure 6.6 is 760% higher than that of sequential implementation of compressed sensing algorithm, a factor of 7.6. Thus, our program enables data distribution on 8 cores with no performance loss.

6.6 Comparison with other optimizations

In this section, we show some of results obtained for 3D MRI reconstruction to demonstrate the efficiency and effectiveness of this parallel algorithm. We compare all the results obtained as a result of optimizations applied to sequential algorithm. These results are obtained after 285 iterations and for same dataset.

Table 6-2 : Execution times and Speedup for all implementations

Type of Implementation	Runtime (ks)	Speedup(x)
CPU - Sequential	152	1
CPU – 8 cores	20	7.6
GPU	608	**No Speedup**
Optimized through mex	429	**No Speedup**

Table 6-2 shows a comparison of execution times and speed ups for all proposed implementations. The speedup obtained on parallel 8-core is 7.6. This is a considerable improvement for reconstruction of 3D MRI using compressed sensing algorithm. Whereas, other proposed implementations did not give considerable results.

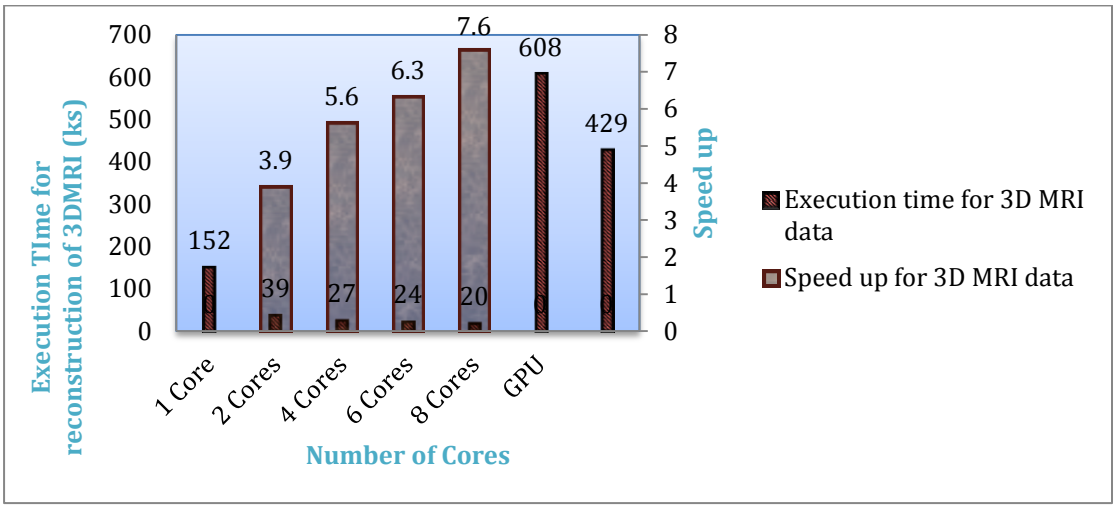


Figure 6.7 : Execution times and Speedup for all implementations

Our goal is to find computationally feasible algorithm that can successfully recover a signal ***x*** from measurement matrix ***y*** for smallest possible number of measurements. Finally, from this graph, we point out that different optimizations introduced in sequential algorithm might give limited results but multicore architecture based results have shown that multicore architectures are most suitable for algorithms with this type of repetitive structure.

7

Conclusion and Future Recommendations

7.1 Conclusions

This dissertation mainly focuses on providing a compressive sensing based solution using parallel processing. Furthermore, this work, though is based on CS, provides solution to two different applications. The first one is 3D-MRI processing and the second one is the parallel processing. In addition, low-complexity parallel algorithm is implemented, to provide a flexibility of use in practical scenarios.

Specifically, in the first part, a 3D MRI reconstruction algorithm is implemented based on CS principles. Firstly, a 2D collaborative sparsity based algorithm is used for reconstruction of each slice. Then all these slices are combined to make a single 3D MRI view. The algorithm is further compared slice by slice with the results of existing algorithm for 2D images. The proposed algorithm reconstructs a single 3D MRI without degrading the image quality. The results are validated based on the signal-to-noise ratio, root mean square error, histogram, topography measure, relative error and correlation coefficients

The Next, the already proposed algorithm is optimized to enhance the performance and increase efficiency for 3D-MRI. This was necessary due to the

high complexity and huge data processing requirements in 3D-MRI. Initially, speed was not one of the objectives for developing this system, but it was considered if the system had to be used for real-time applications. The most computation intensive stage indicated was the 3D reconstruction through the transform area. Since the system is implemented in MATLAB, speed benefits were achieved by implementing the computationally intensive parts through par-for loop, GPU and combination of low level and high level programming methods.

The results show an almost 50% speedup for parallel algorithm when all 8 cores of the machine are utilized instead of one which MATLAB uses by default to run sequential programs. This parallel algorithm is less complex and high performance compared to existing solutions available for MRI. From simulations, it is observed that the SNR results remain the same, while providing high throughput. We further showed that parallel approach can take the benefit of multicore architecture and sometime provides diminished computational barriers compared to GPU based implementation and attains appreciable speed-up. To endorse our technique, efficacy of this algorithm is investigated using 3D MRI data and the superiority of parallel implementation over sequential implementation was confirmed with the acceleration factor.

7.2 Future Work

This discussion concludes with some recommendations of possible future work, which are extensions of the problems considered in this thesis:

- **Optimize Efficiency:** As the experimental results show, the proposed

algorithm can reconstruct the 3D MRI by applying 2D technique to all the data set and visualizing it in 3D view. As a whole 3D technique can be applied to increase the efficiency of 3D algorithm.

- **Other Hardware Architectures based Implementation:** Hardware architectures based fewer complexes and energy efficient system can be proposed, there is still room for hardware optimization. Once the target hardware (e.g., FPGA, ASIC) is chosen and efficient pipelining, place and route will provide a better performance.
- **Optimize Speedup:** Several other configurations such as GPU+par-for, Multi-GPU can be programmed and analyzed for speed and performance. Moreover, different nodes of cluster can be used for parallel implementation for optimized speed up.
- **Optimize GPU based implementation:** GPU based implementation of CS for 3D MRI is optimized through the combination of low level, high level language and CUDA. Still there is room for optimization by implementing the complete algorithm on GPU using low level language and CUDA.

8 Bibliography

1. Shannon, C.E., *A mathematical theory of communication.* ACM SIGMOBILE Mobile Computing and Communications Review, 1948. **5**(1): p. 3-55.
2. Shannon, C.E., *Communication in the presence of noise.* Proceedings of the IRE, 1949. **37**(1): p. 10-21.
3. Zhang, J., et al. *Structural group sparse representation for image compressive sensing recovery*. in *Data Compression Conference (DCC), 2013*. 2013. IEEE.
4. Candès, E.J. and M.B. Wakin, *An introduction to compressive sampling.* Signal Processing Magazine, IEEE, 2008. **25**(2): p. 21-30.
5. Candès, E.J., J. Romberg, and T. Tao, *Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information.* Information Theory, IEEE Transactions on, 2006. **52**(2): p. 489-509.
6. Donoho, D.L., *Compressed sensing.* Information Theory, IEEE Transactions on, 2006. **52**(4): p. 1289-1306.
7. Natarajan, B.K., *Sparse approximate solutions to linear systems.* SIAM journal on computing, 1995. **24**(2): p. 227-234.
8. Donoho, D.L., *Neighborly polytopes and sparse solutions of underdetermined linear equations.* 2005.
9. Chen, S. and D. Donoho. *Basis pursuit*. in *Signals, Systems and Computers, 1994. 1994 Conference Record of the Twenty-Eighth Asilomar Conference on*. 1994. IEEE.
10. Chen, S.S., D.L. Donoho, and M.A. Saunders, *Atomic decomposition by basis pursuit.* SIAM journal on scientific computing, 1998. **20**(1): p. 33-61.
11. Mallat, S.G. and Z. Zhang, *Matching pursuits with time-frequency dictionaries.* Signal Processing, IEEE Transactions on, 1993. **41**(12): p. 3397-3415.
12. Needell, D. and J.A. Tropp, *CoSaMP: Iterative signal recovery from incomplete and inaccurate samples.* Applied and Computational Harmonic Analysis, 2009. **26**(3): p. 301-321.
13. Tropp, J.A. and A.C. Gilbert, *Signal recovery from random measurements via orthogonal matching pursuit.* Information Theory, IEEE Transactions on, 2007. **53**(12): p. 4655-4666.
14. Ji, J. and T. Lang. *Dynamic MRI with compressed sensing imaging using temporal correlations*. in *Biomedical Imaging: From Nano to Macro, 2008. ISBI 2008. 5th IEEE International Symposium on*. 2008. IEEE.
15. Jung, H., et al., *k‐t FOCUSS: A general compressed sensing framework for high resolution dynamic MRI.* Magnetic Resonance in Medicine, 2009. **61**(1): p. 103-116.

16. Montefusco, L.B., et al., *A fast compressed sensing approach to 3D MR image reconstruction.* Medical Imaging, IEEE Transactions on, 2011. **30**(5): p. 1064-1075.
17. Haldar, J.P., D. Hernando, and Z.-P. Liang, *Compressed-sensing MRI with random encoding.* Medical Imaging, IEEE Transactions on, 2011. **30**(4): p. 893-903.
18. Candes, E. and J. Romberg, *Sparsity and incoherence in compressive sampling.* Inverse problems, 2007. **23**(3): p. 969.
19. Majumdar, A. and R.K. Ward, *Joint reconstruction of multiecho MR images using correlated sparsity.* Magnetic resonance imaging, 2011. **29**(7): p. 899-906.
20. Feng, Z., et al., *Improved l1-SPIRiT using 3D walsh transform-based sparsity basis.* Magnetic resonance imaging, 2014.
21. Deutsch, S., A. Averbush, and S. Dekel. *Adaptive compressed image sensing based on wavelet modeling and direct sampling*. in *SAMPTA'09*. 2009.
22. Qu, X., et al., *Iterative thresholding compressed sensing MRI based on contourlet transform.* Inverse Problems in Science and Engineering, 2010. **18**(6): p. 737-758.
23. Plonka, G. and J. Ma, *Curvelet-wavelet regularized split Bregman iteration for compressed sensing.* International Journal of Wavelets, Multiresolution and Information Processing, 2011. **9**(01): p. 79-110.
24. Hong, M., et al., *Compressed sensing MRI with singular value decomposition-based sparsity basis.* Physics in medicine and biology, 2011. **56**(19): p. 6311.
25. Trinh, C.V., et al. *Total variation reconstruction for compressive sensing using nonlocal Lagrangian multiplier*. in *Signal Processing Conference (EUSIPCO), 2013 Proceedings of the 22nd European*. 2014. IEEE.
26. Huang, J. and F. Yang. *Compressed magnetic resonance imaging based on wavelet sparsity and nonlocal total variation*. in *Biomedical Imaging (ISBI), 2012 9th IEEE International Symposium on*. 2012. IEEE.
27. Xu, J., et al., *Improved total variation minimization method for compressive sensing by intra-prediction.* Signal Processing, 2012. **92**(11): p. 2614-2623.
28. Zhang, J., et al. *Improved total variation based image compressive sensing recovery by nonlocal regularization*. in *Circuits and Systems (ISCAS), 2013 IEEE International Symposium on*. 2013. IEEE.
29. Yu, G. and G. Sapiro, *Statistical compressed sensing of Gaussian mixture models.* Signal Processing, IEEE Transactions on, 2011. **59**(12): p. 5842-5858.
30. Yu, G., G. Sapiro, and S. Mallat, *Solving inverse problems with piecewise linear estimators: From Gaussian mixture models to structured sparsity.* Image Processing, IEEE Transactions on, 2012. **21**(5): p. 2481-2499.
31. Yang, J., et al. *Compressive Sensing of Signals from a GMM with Sparse Precision Matrices*. in *Advances in Neural Information Processing Systems*. 2014.

32. Rubinstein, R., M. Zibulevsky, and M. Elad, *Double sparsity: Learning sparse dictionaries for sparse signal approximation.* Signal Processing, IEEE Transactions on, 2010. **58**(3): p. 1553-1564.
33. Aghagolzadeh, M. and H. Radha. *Compressive dictionary learning for image recovery*. in *Image Processing (ICIP), 2012 19th IEEE International Conference on*. 2012. IEEE.
34. Mairal, J., et al. *Online dictionary learning for sparse coding*. in *Proceedings of the 26th Annual International Conference on Machine Learning*. 2009. ACM.
35. Ni, Z.W., et al., *Image compressed sensing based on data-driven adaptive redundant dictionaries.* Progress In Electromagnetics Research M, 2012. **22**: p. 73-89.
36. Yaghoobi, M. and M.E. Davies. *Compressible dictionary learning for fast sparse approximations*. in *Statistical Signal Processing, 2009. SSP'09. IEEE/SP 15th Workshop on*. 2009. IEEE.
37. Qu, X., et al., *Combined sparsifying transforms for compressed sensing MRI.* Electronics letters, 2010. **46**(2): p. 121-123.
38. Shu, X., J. Yang, and N. Ahuja. *Non-local compressive sampling recovery*. in *Computational Photography (ICCP), 2014 IEEE International Conference on*. 2014. IEEE.
39. Lustig, M., D. Donoho, and J.M. Pauly, *Sparse MRI: The application of compressed sensing for rapid MR imaging.* Magnetic resonance in medicine, 2007. **58**(6): p. 1182-1195.
40. Plan, Y. and R. Vershynin, *One‐Bit Compressed Sensing by Linear Programming.* Communications on Pure and Applied Mathematics, 2013. **66**(8): p. 1275-1297.
41. Daubechies, I., M. Defrise, and C. De Mol, *An iterative thresholding algorithm for linear inverse problems with a sparsity constraint.* Communications on pure and applied mathematics, 2004. **57**(11): p. 1413-1457.
42. Kim, D., et al. *High-performance 3D compressive sensing MRI reconstruction*. in *Proceedings of the IEEE Engineering in Medicine and Biology Society*. 2010.
43. Trzasko, J., C. Haider, and A. Manduca. *Practical nonconvex compressive sensing reconstruction of highly-accelerated 3D parallel MR angiograms*. in *Biomedical Imaging: From Nano to Macro, 2009. ISBI'09. IEEE International Symposium on*. 2009. IEEE.
44. Kulkarni, A.M., H. Homayoun, and T. Mohsenin. *A parallel and reconfigurable architecture for efficient OMP compressive sensing reconstruction*. in *Proceedings of the 24th edition of the great lakes symposium on VLSI*. 2014. ACM.
45. Combettes, P.L. and V.R. Wajs, *Signal recovery by proximal forward-backward splitting.* Multiscale Modeling & Simulation, 2005. **4**(4): p. 1168-1200.

46. Hale, E.T., W. Yin, and Y. Zhang, *A fixed-point continuation method for l1-regularized minimization with applications to compressed sensing.* CAAM TR07-07, Rice University, 2007.
47. Wright, S.J., R.D. Nowak, and M.A. Figueiredo, *Sparse reconstruction by separable approximation.* Signal Processing, IEEE Transactions on, 2009. **57**(7): p. 2479-2493.
48. Pruessmann, K.P., et al., *SENSE: sensitivity encoding for fast MRI.* Magnetic resonance in medicine, 1999. **42**(5): p. 952-962.
49. Doyle, M., et al., *Block Regional Interpolation Scheme for k‐Space (BRISK): A Rapid Cardiac Imaging Technique.* Magnetic resonance in medicine, 1995. **33**(2): p. 163-170.
50. Tsao, J., P. Boesiger, and K.P. Pruessmann, *k‐t BLAST and k‐t SENSE: Dynamic MRI with high frame rate exploiting spatiotemporal correlations.* Magnetic Resonance in Medicine, 2003. **50**(5): p. 1031-1042.
51. Khamene, A., et al., *A novel projection based approach for medical image registration*, in *Biomedical Image Registration*. 2006, Springer. p. 247-256.
52. Lefohn, A.E., J.E. Cates, and R.T. Whitaker, *Interactive, GPU-based level sets for 3D segmentation*, in *Medical Image Computing and Computer-Assisted Intervention-MICCAI 2003*. 2003, Springer. p. 564-572.
53. Agulleiro, J. and J.-J. Fernandez, *Fast tomographic reconstruction on multicore computers.* Bioinformatics, 2011. **27**(4): p. 582-583.
54. Lenkiewicz, P., et al. *A new 3D image segmentation method for parallel architectures*. in *Multimedia and Expo, 2009. ICME 2009. IEEE International Conference on*. 2009. IEEE.
55. Jiang, J., W. Luk, and D. Rueckert. *FPGA-based computation of free-form deformations in medical image registration*. in *Field-Programmable Technology (FPT), 2003. Proceedings. 2003 IEEE International Conference on*. 2003. IEEE.
56. Dillinger, P., et al., *FPGA-based real-time image segmentation for medical systems and data processing.* Nuclear Science, IEEE Transactions on, 2006. **53**(4): p. 2097-2101.
57. Li, Q., et al., *Accelerating patch-based directional wavelets with multicore parallel computing in compressed sensing MRI.* Magnetic Resonance Imaging, 2015.
58. Tian, J., et al. *High Performance Parallel Implementation of Compressive Sensing SAR Imaging*. in *Proceedings of the Progress In Electromagnetics Research Symposium (PIERS), Kuala Lumpur, Malaysia, March*. 2012.
59. Borghi, A., et al., *A simple compressive sensing algorithm for parallel many-core architectures.* CAM Report, 2008: p. 08-64.
60. Kim, D., et al., *High-performance 3D compressive sensing MRI reconstruction using many-core architectures.* Journal of Biomedical Imaging, 2011. **2011**: p. 2.
61. Kulkarni, A. and T. Mohsenin, *High performance architectures for omp compressive sensing reconstruction algorithm.* Sort, 2014: p. 1.58.

62. Borghi, A., et al., *A simple compressive sensing algorithm for parallel many-core architectures.* Journal of Signal Processing Systems, 2013. **71**(1): p. 1-20.
63. Che, S., et al. *Accelerating compute-intensive applications with GPUs and FPGAs*. in *Application Specific Processors, 2008. SASP 2008. Symposium on*. 2008. IEEE.
64. Zhang, J., et al., *Image compressive sensing recovery via collaborative sparsity.* Emerging and Selected Topics in Circuits and Systems, IEEE Journal on, 2012. **2**(3): p. 380-391.
65. http://www.pauldebevec.com/Knee/.
66. MathWorks, *MATLAB Parallel Computing Toolbox.*
67. Sharma, G. and J. Martin, *MATLAB®: a language for parallel computing.* International Journal of Parallel Programming, 2009. **37**(1): p. 3-36.