DEPARTMENT OF COMPUTER SCIENCE AND BIOMEDICAL INFORMATICS, SCHOOL OF APPLIED SCIENCES, UNIVERSITY OF THESSALY, 2014

Simulation of Magnetic Resonance Imaging

(Προσομοίωση Μαγνητικής Τομογραφίας)

CHRISTOS G. XANTHIS, MSc

University of Thessaly

DISSERTATION
2014
Department of Computer Science and Biomedical Informatics,
School of Applied Sciences,
University of Thessaly,
Lamia, Greece

ADVISORY COMMITTEE

ANTHONY H. ALETRAS
UNIVERSITY OF THESSALY

GEORGIOS A. KYRIACOU
UNIVERSITY OF THRACE

GEORGIOS D. SERGIADIS
ARISTOTLE UNIVERSITY OF THESSALONIKI

DISSERTATION COMMITTEE

ANTHONY H. ALETRAS
UNIVERSITY OF THESSALY

KONSTANTINOS K. DELIBASIS
UNIVERSITY OF THESSALY

VASILIKI N. IKONOMIDOU
GEORGE MASON UNIVERSITY, WASHINGTON, USA

GEORGIOS A. KYRIACOU
DEMOCRITUS UNIVERSITY OF THRACE

ILIAS G. MAGLOGIANNIS
UNIVERSITY OF PIRAEUS

KONSTANTINA S. NIKITA
NATIONAL TECHNICAL UNIVERSITY OF ATHENS

GEORGIOS D. SERGIADIS
ARISTOTLE UNIVERSITY OF THESSALONIKI

Table of Contents

Acknowledgements

Over the past four years I have received support and inspiration from a great number of individuals.

I would like to express my deepest gratitude to my supervisor, Prof. Anthony H. Aletras, for his excellent guidance, for helping me to develop my background in the field of Magnetic Resonance Imaging and for being a mentor, colleague and friend. His unconditional support and patience made this a fruitful and rewarding journey.

I would like to thank Prof. Håkan Arheden and his team in the Department of Clinical Physiology of the Skåne University Hospital in Lund, Sweden, for supporting me the last two years by providing me with an excellent environment for doing research.

I would also like to thank my advisory committee of Prof. Georgios A. Kyriacou and Prof. Georgios D. Sergiadis.

Last, but not least, I would like to thank my parents and my brother. They were always supporting me and encouraging me with their best wishes.

This work was supported by the European Research Council (PIRG06-GA-2009-256569 FP7 MARIE CURIE IRG), by the Department of Clinical Physiology of the Skåne University Hospital Lund, Lund, Sweden and by the “Alexander S. Onassis public benefit foundation”. Two NVIDIA Tesla C2070 GPU computing boards were donated from NVIDIA through the “Professor Partnership” program whereas the University of Thessaly, Greece funded the purchase of four NVIDIA Tesla C2075 GPU computing boards.

Summary

A new step-by-step comprehensive MR physics simulator (MRISIMUL) of the Bloch equations is presented in this work. The aim was to develop a magnetic resonance imaging (MRI) simulator that makes no assumptions with respect to the underlying pulse sequence and also allows for complex large-scale analysis on a single-node computer without requiring simplifications of the MRI model.

We hypothesized that such a simulation platform could be developed with parallel acceleration of the executable core within the graphic processing unit (GPU) environment. MRISIMUL integrates realistic aspects of the MRI experiment from signal generation to image formation and solves the entire complex problem for densely spaced isochromats and for a densely spaced time axis. The simulation platform was developed in MATLAB whereas the computationally demanding core services were developed in CUDA-C. The MRISIMUL simulator imaged three different computer models: a user-defined phantom, a human brain model and a human heart model whereas three different motion models were introduced as well: cardiac motion, respiratory motion and flow.

The high computational power of GPU-based simulations was compared against other computer configurations. A speedup of about 228 times was achieved when compared to serially executed C-code on the CPU whereas a speedup between 31 to 115 times was achieved when compared to the OpenMP parallel executed C-code on the CPU, depending on the number of threads used in multithreading (2–8 threads). Moreover, simulations of motion with MRISIMUL on single-node and multi-node multi-GPU systems were also examined demonstrating an almost linear scalable performance with the increasing number of available GPU cards, in both single-node and multi-node multi-GPU computer systems.

MRISIMUL is the first MR physics simulator that allows for computationally intense realistic simulations on multi-GPU computer systems. The high performance of MRISIMUL can bring the computational power of a supercomputer or a large computer cluster to a single GPU personal computer. The incorporation of realistic aspects of MR experiments, such as motion, may benefit the design and optimization of existing or new MR pulse sequences, protocols and algorithms.

Summary in Greek

Τίτλος: Προσομοίωση με Μαγνητικό Συντονισμό

Σε αυτή την εργασία παρουσιάζεται ένας νέος, ολοκληρωμένος προσομοιωτής της φυσικής της Μαγνητικής Τομογραφίας (MRISIMUL) που βασίζεται στην επίλιηση του συστήματος των Bloch εξισώσεων. Κύριος στόχος αυτής της εργασίας ήταν η ανάπτυξη ενός προσομοιωτή της Μαγνητικής Τομογραφίας (MRI) που δεν κάνει υποθέσεις σε σχέση με την υποκείμενη ακολουθία παλμών και που επιτρέπει την πολύπλοκη ανάλυση προβλημάτων μεγάλης κλίμακας με τη χρήση ενός υπολογιστή, χωρίς να απαιτείται απλούστευση του πειραματικού μοντέλου.

Η εργασία αυτή βασίστηκε στην υπόθεση ότι μια τέτοια πλατφόρμα προσομοίωσης θα μπορούσε να αναπτυχθεί κάνοντας χρήση της παράλληλης τεχνολογίας που προσφέρει το προγραμματιστικό περιβάλλον των σύγχρονων καρτών γραφικών (GPU). Ο προσομοιωτής MRISIMUL ενσωματώνει ρεαλιστικές πτυχές του πειράματος της Μαγνητικής Τομογραφίας, από την παραγωγή του σήματος έως το σχηματισμό της εικόνας ενώ παράλληλα επιτρέπει την επίλυση του προβλήματος για πυκνά χωρικά και χρονικά μοντέλα. Η πλατφόρμα προσομοίωσης αναπτύχθηκε σε MATLAB ενώ οι υπολογιστικά απαιτητικές βασικές διεργασίες αναπτύχθηκαν στο περιβάλλον της CUDA-C. Ο προσομοιωτής MRISIMUL ενσωματώνει τρία διαφορετικά υπολογιστικά μοντέλα: ένα καθορισμένο από το χρήστη μοντέλο, ένα ανθρώπινο μοντέλο του εγκεφάλου και ένα ανθρώπινο μοντέλο της καρδιάς, ενώ τρία επιπλέον μοντέλα κίνησης εισήχθησαν: καρδιακή κίνηση, αναπνευστική κίνηση και ροή .

Η υψηλή υπολογιστική ισχύ των σύγχρονων καρτών γραφικών (GPU) στη προσομοίωση Μαγνητικής Τομογραφίας συγκρίθηκε έναντι άλλων διαμορφώσεων υπολογιστικών συστημάτων. Η χρήση μιας κάρτας γραφικών παρουσίασε επιτάχυνση περίπου 228 φορές συγκριτικά με την εκτέλεση σειριακού C-κώδικα σε μια CPU ενώ επιτάχυνση μεταξύ 31 και 115 φορές παρουσιάστηκε συγκριτικά με την OpenMP έκδοση του C-κώδικα σε μια CPU, ανάλογα με τον αριθμό των νημάτων που χρησιμοποιήθηκαν σε αρχιτεκτονική πολυνημάτωσης (2-8 νήματα). Επιπλέον, εξετάστηκαν προσομοιώσεις κίνησης με τον προσομοιωτή MRISIMUL σε συστήματα μονού κόμβου και πολλαπλών κόμβων με τη χρήση πολλαπλών καρτών GPU επιδεικνύοντας μια σχεδόν γραμμική σχέση ανάμεσα στην επιτάχυνση και στον αριθμό των διαθέσιμων καρτών GPU στα συστήματα πολλαπλών καρτών γραφικών, είτε μονού νόμβου είτε πολλαπλών κόμβων.

Ο προσομοιωτής MRISIMUL είναι η πρώτη πλατφόρμα προσομοίωσης της φυσικής της Μαγνητικής Τομογραφίας που επιτρέπει την εκτέλεση υπολογιστικά επιβαρυμένων, ρεαλιστικών προσομοιώσεων σε υπολογιστικά συστήματα πολλαπλών καρτών γραφικών. Η υψηλή απόδοση του MRISIMUL μπορεί να φέρει την υπολογιστική ισχύ ενός υπερυπολογιστή ή ενός μεγάλου συμπλέγματος υπολογιστών (cluster) σε έναν προσωπικό υπολογιστή που διαθέτει μια σύγχρονη κάρτα γραφικών. Η ενσωμάτωση ρεαλιστικών πτυχών των πειραμάτων Μαγνητικής Τομογραφίας , όπως πχ κίνηση, μπορεί να ωφελήσει στο μέλλον τον σχεδιασμό και τη βελτιστοποίηση των υφιστάμενων ή νέων παλμοσειρών, πρωτοκόλλων και αλγορίθμων της Μαγνητικής Τομογραφίας.

1. INTRODUCTION

Magnetic Resonance Imaging is a medical imaging technique that encompasses many areas of advanced science and engineering to image the anatomy and physiology of the human body. After the 1940s, when Bloch [1] and independently Purcell and Pound [2] first described the nuclear induction concept, it took more than 20 years before Raymond Damadian conceptualize the medical application of Nuclear Magnetic Resonance (NMR) in detecting certain mouse tumors [3].

In 1973, Paul Lauterbur submitted an article in Nature [4] proposing the use of a second magnetic field within the main magnetic field so that NMR signals originating from different locations of the object to be imaged could be localized in space. The concept of magnetic field gradients that Paul Lauterbur introduced led to the acquisition of the first MR image of a living mouse in January 1974 [5]. The first human body MRI scan was performed in 1977 by Damadian et. al [6].

Since its inception as a medical imaging modality in 1973, MRI has matured to become the technology of choice for visualizing soft tissue contrast without using ionizing radiation. Although advanced hardware designs and imaging techniques have been proposed during the last 20 years so that image quality and patient comfort can be secured, the basic principles of magnetic resonance imaging remain the same. In this section, these basic principles of clinical MRI will be introduced whereas more detailed information on MR physics can be found in the literature [7, 8].

1.1 Magnetic Resonance Imaging (MRI)

1. Basic principles

Clinical MRI is based on the intrinsic properties of the hydrogen nuclei (^{1}H) which are abundant in physiological tissues in the form of water and lipid molecules. The hydrogen nucleus contains a single positively charged proton which has a magnetic moment. This magnetic moment has a magnitude and a direction and can be considered as a small magnet that spins along its axis.

These magnets, which are called spins, are randomly oriented in the absence of an external magnetic field. However, once placed in the strong magnetic field of an MR scanner, a force is exerted on the spins causing them to:

i. align themselves either parallel or antiparallel to the direction of the external magnetic field and
ii. precess about the main magnetic field

In its simplest representation, each spin can be represented as a vector in a three dimensional space, as illustrated in figure 1. In this figure, the z-axis is aligned with the strong magnetic field whereas the xy plane (transverse plane) is perpendicular to this magnetic field.

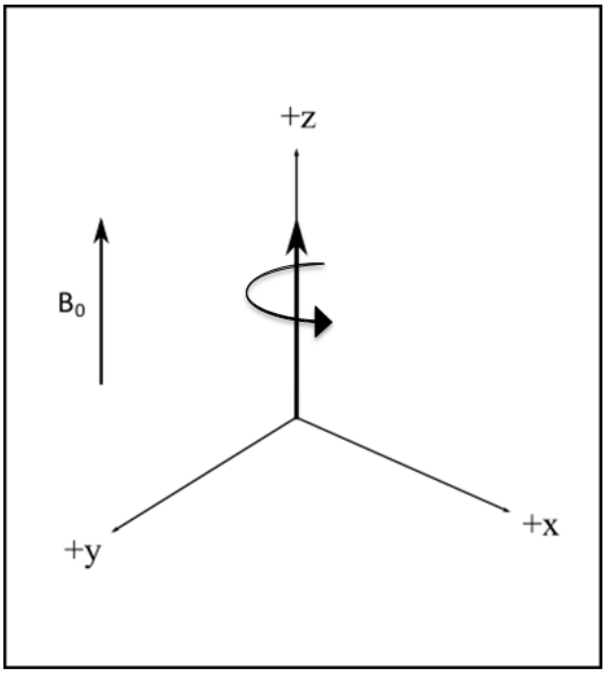

Fig. 1.The left-handed coordinate system: the z-axis is aligned with the strong magnetic field B_0 whereas the xy plane (transverse plane) is perpendicular to this magnetic field. Each spin is represented by a vector which is aligned with the direction of the main magnetic field B_0. B_0 forces spins to precess about it with an angular frequency ω_0 which is described by the Larmor equation.

The orientation of the spins may be either parallel or antiparallel to the direction of the magnetic field, with the parallel orientation being the preferred state of orientation (less energy state), as described by the Boltzmann equation

$$\frac{N\uparrow}{N\downarrow} = 1 + \frac{\gamma\!\!\!^{-}hB_0}{kT}$$

where $N\uparrow$ and $N\downarrow$ are the number of spins parallel and antiparallel with B_0, respectively, B_0 is the magnitude of the strong magnetic field (in Tesla), T is the temperature of the material (in Kelvin), $\gamma\!\!\!^{-}$ is the rationalized gyromagnetic ratio of protons (42.576 MHz/Tesla for the hydrogen nucleus) and k and h constants (Boltzmann and rationalized Planck constant, respectively). This spins parallel to the B_0 slightly outnumber the ones antiparallel to B_0 and this excess is directly related to the strength of the strong magnetic field.

Due to hydrogen nuclei mass and the same interaction of the spins with the strong magnetic field, spins start also to precess about the direction of the main magnetic field (figure 1) with an angular frequency ω_0 (in Hz) which is described by the Larmor equation

$$\omega_0 = \gamma\!\!\!^{-}B_0$$

For the hydrogen nucleus, the precession frequency is approximately 64MHz for a typical clinical MR scanner of 1.5Tesla.

The vector sum of all the spins form the net magnetization vector M which precesses about magnetic fields in the same way that the individual spins do. At equilibrium state, that is the initial state of placing the spins in the strong magnetic field, the net magnetization vector

lies along the direction of the applied magnetic field and is called the equilibrium magnetization M_0. It is the magnitude of the equilibrium magnetization M_0 that partially determines the maximum signal intensity of the final image. The magnitude of M_0 is described by the Pierre Curie equation

$$M_0 = PD\ I(I+1)\frac{B_0}{kT}$$

where PD is the Proton Density which is the concentration of hydrogen nuclei within a sample of tissue and I is the quantum spin number which is equal to $^1/_2$ in the case of hydrogen nuclei.

2. Pulse sequence

In order to produce an MR image, a pulse sequence has to be executed by the MR system. The pulse sequence consists of a set of radiofrequency pulses and gradient pulses that are applied at predetermined times. The pulse sequence also controls when signal is acquired.

Prior to the execution of a pulse sequence, the magnetization vector is aligned with the strong magnetic field along the z-axis. The creation of the MR signal that provides the diagnostic information is based on bringing the net magnetization vector M away from the z-axis, down to the transverse plane. This is accomplished by applying a secondary magnetic field B_1 through the transmission of a short radiofrequency (RF) pulse. In contrast to the main magnetic field B_0, B_1 is not static but rather it rotates perpendicular to the main magnetic field B_0 with a frequency f_1. The frequency of B_1 needs to be at the same frequency that the spins of the tissues to be imaged precess. RF pulses are generated by transmitter coils which surround the whole body or part of it. Typical characteristics of the RF pulses include also the duration T of the pulse (in sec), the RF bandwidth (BW) which is a measure of the frequency content of the pulse and is usually given by the full width at half maximum (FWHM) of the frequency profile (in Hz) and the flip angle α which describes the rotation angle that the RF pulse induces to the on-resonance spins.

For the spatial localization of the origin of the MR signal within the body, three distinct processes for MR imaging are used: slice selection, frequency encoding and phase encoding. For that purpose, gradient pulses are applied throughout the MR experiment. Gradient pulses refer to short linear spatial variations of the main magnetic field strength (z-axis). These variations are proportional to the distance of the spins to be imaged from the center of the magnet (isocenter) in any of the three orthogonal directions. Gradients are generated by three separate gradient coils, one for each of the three orthogonal directions, driven separately by their own gradient amplifiers. Gradient amplitudes are typically measured in mT/m whereas the gradient amplitudes usually increase linearly with respect to time, as defined by the slew rate of the gradients system measured in T/m/sec.

Last, the MR signal that is produced by the human body is detected by the receiver coils of the MR system. The receiver coils may be the same coils used for RF transmission or separate coils that are tuned to the base frequency of the MR signal. Prior to image reconstruction, the data acquisition system amplifies, demodulates, filters and digitizes the

acquired MR signal. After demodulation, the high frequency oscillations caused by the presence of the main magnetic field B_0 have been removed and the final signal has been left with a narrower spectrum of frequencies, as it is defined by the receiver bandwidth (rBW) which is typically several hundred kHz wide.

The process of demodulation follows the conceptualization of laboratory frame of reference (LAB) and rotating frame of reference (ROT) being frequently used for demonstration and comprehension purposes. In the rotating frame of reference the precession about B_1 is the only motion visible to the observer whereas in the laboratory frame of reference the trajectory of the magnetization vector M is driven by the two independent precessions about the two magnetic fields, B_1 and B_0.

3. Relaxation times

The final form of the MR signal depends on both the pulse sequence events and the intrinsic characteristics of the spins, namely relaxation times. As soon as the RF excitation pulse is switched off, the already excited spins begin to relax back to their equilibrium state following two independent mechanisms: longitudinal relaxation and transverse relaxation.

Longitudinal relaxation or spin-lattice relaxation refers to the realignment of the spins with the main magnetic field due to energy exchange between the spins and their surrounding environment, also known as lattice. This process is characterized by the spin-lattice relaxation time T1 and shows how fast spins release the excess energy they previously absorbed from the RF pulse back to the lattice. T1 changes with field strength and is getting longer as the magnetic field strength increases. The magnetization along the z-axis increases exponentially until it reaches the equilibrium magnetization after a long time, following the equation

$$M_z(t) = M_0[1 - e^{-\frac{t}{T_1}}]$$

Transverse relaxation or spin-spin relaxation refers to the time required for the spins to lose phase coherence as they are spinning perpendicular to the main magnetic field. As described earlier, spins themselves can be considered as tiny magnets that interact with each other. This interaction creates microscopic changes in the overall magnetic field that the spins experience, which in turn results in a reduction in the net transverse magnetization because the spins no longer precess in a coherent way. This process is characterized by the relaxation time constant T2, whose value depends on the mobility of the protons.

However, reduction in the transverse magnetization is also caused by the main magnetic field inhomogeneity since it forces individual spins to precess at different frequencies. The combined mechanism of relaxation due to spin-spin interactions and main magnetic field inhomogeneity ΔB_0 is referred to as T2* relaxation. The T2* value is always smaller than the T2 value and the signal loss on the transverse plane follows the equation

$$M_{xy}(t) = M_{xy\,(t=0)} e^{-\frac{t}{T_2^*}}$$

where

$$\frac{1}{T_2^*} = \gamma \Delta B_0 + \frac{1}{T_2}$$

4. Image Reconstruction

The MR signal contains a wide spectrum of frequencies that contain information about the spatial location of various tissues within the body. The data acquisition module of the MR system is responsible for acquiring and digitizing the MR signal prior to post processing. As described earlier, the receiver system demodulates the MR signal by removing the Larmor frequency. The complex signal that is received is suitable for Fourier transformation and MR image reconstruction.

The complex MR signal is then placed in a two-dimensional matrix, called k-space, following a filling trajectory, which is defined by the frequency and phase encoding processes of the pulse sequence. The trajectory being selected each time defines the acquisition approach, determines the image reconstruction algorithm to be employed and influences the final reconstructed image. The final image is then reconstructed by applying the two-dimensional Fourier transformation (2D-FT) of k-space. The area around the center of k-space corresponds to the lower frequencies of the image and holds contrast information, while the outer areas of the k-space correspond to image edge details. The most popular k-space trajectory is the Cartesian trajectory whereas other trajectory schemes, such as the spiral trajectory, have been developed to decrease the total scan time.

1.2 MRI simulations

Compared to other diagnostic imaging methodologies, MRI requires a higher level of understanding of the underlying physics in order to acquire and interpret the images. The optimization of MRI pulse sequences and imaging protocols usually involves an iterative process of parameter modification until the desired image characteristics are obtained. Since this process is time consuming, it is unlikely that the entire parameter space is considered when optimizing imaging protocols. Moreover, it is not always easy to differentiate between artifacts originating from different sources.

In MRI, motion is one major source of artifacts, which may degrade image quality. Such artifacts are generated from rhythmic motion such as blood flow, respiration, cardiac motion, and, sometimes, from unpredictable patient body motion while MRI data are acquired. In many cases, even a small amount of motion may induce large errors in some MR-related applications, such as in functional magnetic resonance imaging (fMRI) analysis [9]. Image acquisition becomes more challenging in cardiac MR applications due to heart motion caused by both the beating heart during the cardiac cycle as well as by its motion as a result of the respiratory cycle [10].

Therefore, the optimization of MRI pulse sequences and imaging protocols may involve techniques and algorithms that may be time consuming and/or may involve human volunteer experimentation, animal models or the development of advanced phantoms to simulate, for example, physiological motion [11-13].

In the past, MRI physics simulators have been used for optimizing imaging protocols, tracking artifact sources but also for training purposes. Since the introduction of the first nuclear magnetic resonance physics simulator by Summers, Axel and Israel in 1986 [14] there has been an effort to introduce as much realism as possible into the simulators. In most cases, simulations were used to answer a particular methodological problem and to provide insight with respect to future pulse sequence or imaging protocol development. Recently, a method for simulating stent MRI artifacts was developed [15]. Also, simulations for the purpose of optimizing MRI acquisition times have been implemented [16], static and rotating field imperfections have been incorporated [17], dynamic signals have been simulated [18], gradient fields have been taken into account [19] and analytical solutions have been proposed [20]. Last, a software tool for pulse sequence design has been proposed [21]. While these applications show that there is a role for simulations in MRI, simulations remain mostly a tool for physicists and engineers. One exception is the use of MRI simulators for training clinical personnel and students [22-25]. However, even in these cases the MRI simulators are confined to only a few pulse sequences and they address basic magnetic resonance (MR) physics concepts.

To date, incorporation of motion and motion artifacts has been proposed only by a few MR simulators. Petersson et al. [26] developed a simulation method based on k-space formalism. Although this simulator simulates movement or flow by allowing signal phase changes during the sampling procedure, it does not allow for the simulation of more realistic experiments. POSSUM [9] is a more advanced simulation platform which focuses on fMRI and incorporates realistic rigid-body motion of the head anatomical object. Although POSSUM was designed and developed taking into account the simulation of spin motion, it is limited to applications relevant to fMRI and simulates simple translational and/or rotational motion limited to the time period of the pulse sequence when the signal is acquired. This is mainly due to the high computational load introduced by the simulation of motion during the entire course of the pulse sequence. For the same reason, POSSUM usually simulates spins only within the slice of interest and not within the entire 3D anatomical object. However, POSSUM can achieve shorter execution times by parallelizing simulations on computer systems of multiple nodes

Two more advanced MRI simulation platforms based on open source code have been presented to date. SIMRI [27] is a simulator based on discrete-event Bloch equation computations that was developed for training purposes. It consists of a user interface for manipulating a number of protocol parameters and allows for visualizing the resulting magnetization vector. On the other hand, the JEMRIS simulator [28] is based on the numerical solutions of the Bloch equations and it is the most disseminated advanced simulator platform to date. JEMRIS is usually used for answering more complex questions on pulse sequence design whereas it incorporates simulation of motion as well. Both SIMRI and JEMRIS simulators have been shown to perform well on a desktop computer for simple

experiments and can improve their execution speeds by utilizing parallel processing for more complicated large-scale experiments on computer clusters supporting the message passing interface (MPI) communications protocol.

While both open source MRI simulation platforms represent the state of the art, they also have certain limitations. SIMRI is confined to simulating a handful of pulse sequences. Also, some compromises have been made (e.g. T2* is computed rather than simulated) [27] in order to allow for reasonable execution times. Moreover, both SIMRI and JEMRIS environments, along with their installation procedures, are geared towards the technical savvy individual. In order to run more complex experiments both SIMRI and JEMRIS systems require an advanced computer cluster setup of multiple nodes, implementation knowledge and the ability to perform source code modifications [29]. In terms of motion, JEMRIS, just like POSSUM, simulates simple translational and rotational motion only within the slice of interest and not within the entire 3D anatomical object. However, this assumption is not related to any specific design limitation of these simulation platforms but rather imposed due to the high computational load introduced by the simulation of 3D motion during the entire course of the pulse sequence. Last, none of the above mentioned simulators do not incorporate cardiac motion, respiratory motion or other more complicated forms of motion originating from the patient.

1.3 Graphic Processing Unit (GPU) and CUDA technology

A graphics processing unit (GPU) is a specialized single-chip processor that handles computationally-intensive tasks needed for computer graphics. Within the last decade, it has become increasingly common the utilization of GPU cards to perform computations traditionally handled by the central processing unit (CPU). The high computational power of the shader pipeline, initially planned to work solely for computer graphical operations, has turned GPUs into general purpose computing tools that, nowadays, find application in several scientific fields.

GPU-computing refers to the use of GPU together with CPU to accelerate computationally intensive applications. The compute-intensive portion of the application is offloaded to the GPU, while the remainder of the code still runs on the CPU which now unfetters cycles that can be used for other processes. General-purpose computing on graphics processing units (GPGPU) is generally fitted to applications that exhibit data parallelism, that is multiple computations but only a single instruction each time, known as single instruction-multiple data (SIMD). GPGPU applications usually handle large data sets, present high parallelism behavior and have minimal dependency between data elements. Additionally, concurrent use of multiple GPUs parallelizes even more the GPGPU applications.

In 2007, NVIDIA Inc. (NVIDIA, Santa Clara, CA) presented the Compute Unified Device Architecture (CUDA) which is an architecture scheme by which NVIDIA built GPUs in order to perform both graphical and general-purpose computing operations. CUDA is a parallel computing platform and programming model that allows software developers direct access to the GPU resources (memory, registers, etc) through the CUDA C language environment.

Since the introduction of GPGPU computing, several research and engineering applications have achieved high performance by utilizing GPU and CUDA computing. However, to date, the utilization of GPUs to perform computations in MR applications has been limited usually to applications relevant to image reconstruction techniques [30, 31].

1.4 Specific aims

In this work, a new MR physics simulator platform by the name MRISIMUL is presented. MRISIMUL is a step-by-step comprehensive simulator of the Bloch equations which integrates realistic aspects of the MRI experiment from signal generation to image formation. The specific aim of this study was to develop an enhanced comprehensive computer simulation platform of MRI physics that makes no assumptions with respect to the underlying pulse sequence, integrates realistic aspects of the MRI experiment (such as motion) and also allows for complex large-scale analysis on a single computer without requiring simplifications of the MRI model so as to yield reasonable execution times.

We hypothesized that such a simulation platform could be developed with parallel acceleration of the executable core within the GPU environment and that cardiac motion, respiratory motion and flow during the entire course of the MR pulse sequence could be simulated within a reasonable amount of time by distributing the computational load on a multi-GPU multi-node system. Finally, we hypothesized that an MR simulator could be designed and implemented around a user-defined motion model for simulation of MRI physics within the entire space occupied by a three dimensional anatomical object.

References

[1] F. Bloch, "Nuclear Induction," *Phys. Rev,* vol. 70, p. 460, 1946.

[2] E. M. Purcell, H.C. Torrey, R.V. Pound, "Resonance Absorption by Nuclear Magnetic Moments in a Solid," *Phys. Rev,* vol. 69, pp. 37-38, 1946.

[3] R. Damadian, "Tumor detection by nuclear magnetic resonance," *Science,* vol. 171, pp. 1151-3, Mar 19 1971.

[4] P. C. Lauterbur, "Image Formation by Induced Local Interactions: Examples Employing Nuclear Magnetic Resonance," *Nature,* vol. 242, pp. 190-191, 03/16/print 1973.

[5] P. C. Lauterbur, "Magnetic resonance zeugmatography," in *Pure and Applied Chemistry* vol. 40, ed, 1974, p. 149.

[6] R. Damadian, M. Goldsmith, and L. Minkoff, "NMR in cancer: XVI. FONAR image of the live human body," *Physiol Chem Phys,* vol. 9, pp. 97-100, 108, 1977.

[7] M. A. Bernstein, K. F. King, X.J. Zhou, *Handbook of MRI pulse sequences*: Elsevier, 2004.

[8] D. W. McRobbie, E. A. Moore, M. J. Graves, and M. R. Prince, *MRI from Picture to Proton*: Cambridge University Press, 2006.

[9] I. Drobnjak, D. Gavaghan, E. Suli, J. Pitt-Francis, and M. Jenkinson, "Development of a functional magnetic resonance imaging simulator for modeling realistic rigid-body motion artifacts," *Magn Reson Med,* vol. 56, pp. 364-80, Aug 2006.

[10] K. McLeish, D. L. Hill, D. Atkinson, J. M. Blackall, and R. Razavi, "A study of the motion and deformation of the heart due to respiration," *IEEE Trans Med Imaging,* vol. 21, pp. 1142-50, Sep 2002.

[11] C. Constantinides, X. Zhong, V. Tzagkarakis, G. Cofer, and R. Gravett, "Emulation of human and rodent cardiac motion with a computer-controlled cardiac phantom using DENSE MRI," *Concepts in Magnetic Resonance Part A,* vol. 42, pp. 59-71, 2013.

[12] K. P. Forbes, J. G. Pipe, J. P. Karis, V. Farthing, and J. E. Heiserman, "Brain imaging in the unsedated pediatric patient: comparison of periodically rotated overlapping parallel lines with enhanced reconstruction and single-shot fast spin-echo sequences," *AJNR Am J Neuroradiol,* vol. 24, pp. 794-8, May 2003.

[13] J. G. Pipe, "Motion correction with PROPELLER MRI: application to head motion and free-breathing cardiac imaging," *Magn Reson Med,* vol. 42, pp. 963-9, Nov 1999.

[14] R. M. Summers, L. Axel, and S. Israel, "A computer simulation of nuclear magnetic resonance imaging," *Magn Reson Med,* vol. 3, pp. 363-76, Jun 1986.

[15] G. Yan and J. Xiaohua, "Simulations of the Stent Artifacts in Magnetic Resonance Imaging," *Magnetics, IEEE Transactions on,* vol. 48, pp. 659-662, 2012.

[16] G. Andria, F. Attivissimo, G. Cavone, and A. M. L. Lanzolla, "Acquisition Times in Magnetic Resonance Imaging: Optimization in Clinical Use," *Instrumentation and Measurement, IEEE Transactions on,* vol. 58, pp. 3140-3148, 2009.

[17] M. B. Olsson, R. Wirestam, and B. R. Persson, "A computer simulation program for MR imaging: application to RF and static magnetic field imperfections," *Magn Reson Med,* vol. 34, pp. 612-7, Oct 1995.

[18] J. X. Ji and Y. Jiraraksopakun, "Model-based simulation of dynamic magnetic resonance imaging signals," *Biomedical Signal Processing and Control,* vol. 3, pp. 305-311, 10// 2008.

[19] T. H. Jochimsen, A. Schafer, R. Bammer, and M. E. Moseley, "Efficient simulation of magnetic resonance imaging with Bloch-Torrey equations using intra-voxel magnetization gradients," *J Magn Reson,* vol. 180, pp. 29-38, May 2006.

[20] I. Nazarova and M. A. Hemminga, "Analytical analysis of multi-pulse NMR," *J Magn Reson,* vol. 170, pp. 284-9, Oct 2004.

[21] W. R. Overall and J. M. Pauly, "An Extensible, Graphical Environment for Pulse Sequence Design and Simulation," in *International Society for Magnetic Resonance in Medicine 15*, Berlin, 2007.

[22] C. A. Cocosco, V. Kollokian, R. K. Kwan, B. Pike, and A. C. Evans, "BrainWeb: Online Interface to a 3D MRI Simulated Brain Database," in *3rd International Conference on Functional Mapping of the Human Brain*, Copenhagen, 1997.

[23] T. Hacklander, H. Mertens, and B. M. Cramer, "[Computer simulation of a clinical magnet resonance tomography scanner for training purposes]," *Rofo,* vol. 176, pp. 1151-6, Aug 2004.

[24] L. G. Hanson, "A graphical simulator for teaching basic and advanced MR imaging techniques," *Radiographics,* vol. 27, p. e27, Nov-Dec 2007.

[25] D. Rundle, S. Kishore, S. Seshadri, and F. Wehrli, "Magnetic resonance imaging simulator: a teaching tool for radiology," *J Digit Imaging,* vol. 3, pp. 226-9, Nov 1990.

[26] J. S. Petersson, J. O. Christoffersson, and K. Golman, "MRI simulation using the k-space formalism," *Magn Reson Imaging,* vol. 11, pp. 557-68, 1993.

[27] H. Benoit-Cattin, G. Collewet, B. Belaroussi, H. Saint-Jalmes, and C. Odet, "The SIMRI project: a versatile and interactive MRI simulator," *J Magn Reson,* vol. 173, pp. 97-115, Mar 2005.

[28] T. Stocker, K. Vahedipour, D. Pflugfelder, and N. J. Shah, "High-performance computing MRI simulations," *Magn Reson Med,* vol. 64, pp. 186-93, Jul 2010.

[29] K. G. Baum, G. Menezes, and M. Helguera, "Simulation of High-Resolution Magnetic Resonance Images on the IBM Blue Gene/L Supercomputer Using SIMRI," *International Journal of Biomedical Imaging,* vol. 2011, 2011.

[30] D. S. Smith, J. C. Gore, T. E. Yankeelov, and E. B. Welch, "Real-Time Compressive Sensing MRI Reconstruction Using GPU Computing and Split Bregman Methods," *Int J Biomed Imaging,* vol. 2012, p. 864827, 2012.

[31] T. S. Sorensen, T. Schaeffter, K. O. Noe, and M. S. Hansen, "Accelerating the nonequispaced fast Fourier transform on commodity graphics hardware," *IEEE Trans Med Imaging,* vol. 27, pp. 538-47, Apr 2008.

2. METHODS

2.1 Simulator overview

A block diagram of the MRISIMUL GPU-based simulator is shown in Fig. 2.1. The simulator's four inputs were the 3D computer model of a 3D object, the pulse sequence, the Bo inhomogeneity map and a set of MR system parameters, which described the MRI scanner's hardware (e.g. gradients slew rates, max B1 field etc.) Based on these inputs, the computational kernel of the simulator computed in parallel the magnetization vectors within the 3D object over time. Afterwards, the total magnetization vector was recorded in order to fill k-space. It should be noted that the simulation takes place within the entire 3D object and not just within a given slice. Before applying the 2D Fourier transform and reconstructing the MR image, thermal noise was added to the k-space data. A more detailed description of each component follows.

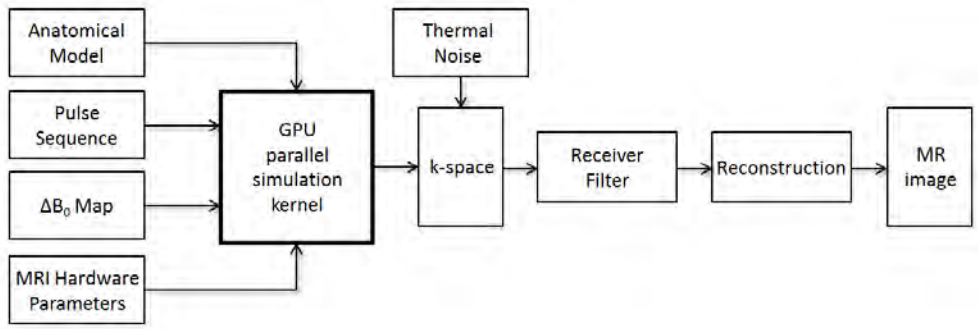

Fig. 2.1. Block diagram of the MRI simulator. The computational kernel of the simulation platform (in bold frame) was executed in parallel on the GPU whereas the rest of the software was executed serially.

2.2 Anatomical model

The first input to the simulation platform was the 3D computer model of a realistic spin system. An isochromat was defined as a microscopic group of spins with common MR characteristics (T1, T2, proton density, precessional frequency). These characteristics were represented by the following notation:

$$O(\overrightarrow{r(t)}, tissue) = \{r_x(t), r_y(t), r_z(t), T_1(tissue), T_2(tissue), \rho(tissue), \omega(tissue)\} \quad (1)$$

where T_1, T_2, and ρ are, respectively, the spin-lattice relaxation time, spin-spin relaxation time and the proton density of the corresponding tissue, ω was the precession frequency of that tissue within the main magnetic field and $\vec{r} = (r_x, r_y, r_z)$ defined the spatial coordinates. These coordinates were updated depending on the actual placement of the 3D object within the bore of the virtual scanner according to the following equation:

$$\vec{r}_{new} = \vec{r} - \vec{r}_{isocenter} \quad (2)$$

Equation [1] allowed for the use of multiple tissues in every location $\vec{r}$ within the digital 3D object. Therefore, chemical-shift, intra-voxel dephasing and partial volume effects could be simulated. Moreover, the time dependence of the isochromat spatial coordinates allowed for the introduction of a motion model for all isochromats within the anatomical object. A more detailed description of the anatomical models and motion models examined in this study follows in separate sections.

2.3 Pulse sequence

The pulse sequence consisted of the excitation of the 3D object by RF pulses, the modulation of the main magnetic field B_0 by the gradients and the acquisition of the sum of the magnetization vectors of the object (summed magnetization). At the start of the pulse sequence, the initial magnetization vector M of every isochromat was along the z-axis (i.e. M_x=M_y=0) and was represented by

$$\vec{M}_{(t=0)} = \rho M_o \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \qquad (3)$$

where M_0 is the equilibrium magnetization value of the isochromat and ρ its proton density.

In MRISIMUL, the pulse sequence was represented as a two dimensional “pulse sequence matrix”. This matrix was constructed by grouping together six vectors. Each of the first five vectors described the time evolution of a magnetic field modifier i.e. the RF excitation (B_{1x}, B_{1y}) and the gradient of the main magnetic field along the three axes (G_x, G_y, G_z). The sixth vector (ACQ) described the times when the receiver was on.

Note that the time-step Δt of the ”pulse sequence matrix” sets the lower limit of the dwell time, DW_{min}, and therefore the maximum bandwidth, BW_{max}, of the quadrature detection receiver as well as the maximum sampling frequency:

$$f_{s_{max}} = BW_{max} = \frac{1}{DW_{min}} = \frac{1}{\Delta t} \qquad (4)$$

Sampling frequencies lower than $f_{s_{max}}$ were simulated by integrating the signal during the dwell time in a manner compatible to that of a sample-and-hold circuit.

2.4 Simulation computational kernel

The computational kernel of the simulator was executed for the duration of the application of the MR pulse sequence on the entire 3D virtual object. Its design and development was based on the solutions of the Bloch equations at every point of the object at each time point of the pulse sequence [1]. The solutions of the Bloch equations provided the temporal evolution of the spin magnetization vectors under the influence of the RF pulses and magnetic field gradients applied to every location within the object.

The vector expression of the Bloch equation was

$$\frac{d\vec{M}}{dt} = \gamma\left(\vec{M} \times \vec{B}_{eff}\right) - \begin{bmatrix} \frac{M_x}{T_2} \\ \frac{M_y}{T_2} \\ \frac{M_z - \rho M_0}{T_1} \end{bmatrix} \quad (5)$$

where γ was the gyromagnetic ratio, M_x, M_y, M_z were the components of the magnetization vector M at time t, M_o was the equilibrium magnetization and T_1, T_2 and ρ were, respectively, the spin-lattice relaxation time, spin-spin relaxation time and the proton density of the tissue. The cross product in the first term describes the rotation of the magnetization vector M about the magnetic field B_{eff} with a rotational speed of

$$\omega_{eff} = \gamma B_{eff} \quad (6)$$

B_{eff} was the effective magnetic field experienced by the isochromat on location $(\vec{r})$ at time t and can be modeled as follows in the rotating frame of reference:

$$B_{eff}(\vec{r}, t, tissue) = \left(xG_x(t) + yG_y(t) + zG_z(t)\right)\vec{z} + \Delta B(\vec{r}, t, tissue)\vec{z} + B_{1x}(\vec{r}, t)\vec{x} + B_{1y}(\vec{r}, t)\vec{y} \quad (7)$$

where G_x, G_y and G_z were the imaging static magnetic field gradients and $B_1(\vec{r}, t)$ was the applied RF pulse. ΔB accounted for any off-resonance effects and was given by:

$$\Delta B(\vec{r}, t, tissue)\vec{z} = \Delta B_0(\vec{r}, t)\vec{z} + \frac{\Delta\omega_{ChemicalShift}(tissue)}{\gamma}\vec{z} + \sum_{k=1}^{nTX}\left(\frac{\gamma Bo - \omega_{RF}^{k}(\vec{r}, t)}{\gamma}\right)\vec{z} \quad (8)$$

where ΔB_0 corresponded to the inhomogeneity of the main magnetic field. The second term represents chemical shift and the third term represents the difference between the resonant frequency of the spins and the carrier frequency of the RF pulse. The time dependency of ω_{RF} allows the application of frequency modulated RF pulses (e.g. adiabatic pulses). The sum operator of the third term in equation 8 allows for the application of nTX RF pulses at different frequencies over nTX transmit coils.

In equation 7, both ΔB and B_1 are both time and space dependent. The time dependence of ΔB allows for the introduction of a time varying inhomogeneity field map in the simulator whereas the spatial dependency of the RF field B_1 allows for the introduction of RF pulse sensitivity maps in the simulator. However, as it will be discussed below in more detail, the application of the time varying inhomogeneity field map requires increased memory space and may be limited by the available hardware.

The different events of the pulse sequence dictated the temporal evolution of each spin within the object. This evolution was calculated by the computational kernel for every time-step Δt of the pulse sequence with the use of rotation and scaling matrices according to the following equation:

$$\vec{M}\left(\overrightarrow{r(t+\Delta t)}, t+\Delta t\right) = (1 - E_1)\vec{M}_{(t=0)} + SCAL_{relaxation}ROT_{Beff}(alpha)\,\vec{M}\left(\overrightarrow{r(t)}, t\right) \quad (9)$$

where $E_1 = exp(-\Delta t/T_1)$, where $\vec{M}_{(t=0)}$ was the magnetization vector at the beginning of the pulse sequence, $SCAL_{relaxation}$ was the scaling matrix that described the relaxation effects and, $ROT_{Beff}(alpha)$ was the rotation matrix about the axis of the B_{eff} by an angle $alpha$. For these calculations a left-handed coordinate system was used (figure 2.2A).

The scaling matrix $SCAL_{relaxation}$ is given by:

$$SCAL_{relaxation} = \begin{bmatrix} E_2 & 0 & 0 \\ 0 & E_2 & 0 \\ 0 & 0 & E_1 \end{bmatrix} \qquad (10)$$

where $E_2 = exp(-\Delta t/T_2)$ and E_1 are as described above.

The angle $alpha$ is calculated according to the following equation:

$$alpha = \gamma \left\| B_{eff}\left(\overrightarrow{r(t)}, t\right)\right\| \Delta t \qquad (11)$$

In the absence of an RF pulse excitation, $\vec{B}_{eff}$ is always aligned to the z axis and ROT_{Beff} is associated to the angle $alpha$ by the following rotation matrix (which describes rotation about the z axis):

$$ROT_{Beff}(alpha) = ROT_z(alpha) = \begin{bmatrix} cos(alpha) & -sin(alpha) & 0 \\ sin(alpha) & cos(alpha) & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (12)$$

In the case, where there is no component of $\vec{B}_{eff}$ along the z-axis (i.e. no gradient is applied and no off-resonance effect exists) and an RF pulse is applied (Fig. 2.2B and 2.2C), the rotation matrix ROT_{Beff} can be calculated as follows:

$$ROT_{Beff}(alpha) = ROT_z(phi)ROT_y(alpha)ROT_z(-phi) \qquad (13)$$

where phi is defined to be the angle:

$$phi = tan^{-1}\left(\frac{B_{1x}\left(\overrightarrow{r(t)}, t\right)}{B_{1y}\left(\overrightarrow{r(t)}, t\right)}\right) \quad (14)$$

and

$$ROT_y(alpha) = \begin{bmatrix} cos(alpha) & 0 & sin(alpha) \\ 0 & 1 & 0 \\ -sin(alpha) & 0 & cos(alpha) \end{bmatrix} \qquad (15)$$

In the most complex case where $\vec{B}_{eff}$ has non-zero components on the three axes (Fig. 2.2D, 2E and 2F), the rotation matrix $ROT_{Beff}(alpha)$ can be calculated as follows:

$$ROT_{Beff}(alpha) = ROT_z(phi)ROT_x(-beta)ROT_y(alpha)ROT_x(beta)ROT_z(-phi) \quad (16)$$

where

$$beta = tan^{-1}\left(\frac{B_{eff\,z}\left(\overrightarrow{r(t)},t\right)}{\sqrt{B_{1x}^2\left(\overrightarrow{r(t)},t\right)+B_{1y}^2\left(\overrightarrow{r(t)},t\right)}}\right) \quad (17)$$

and

$$ROT_x(beta) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(beta) & -\sin(beta) \\ 0 & sin(beta) & \cos(beta) \end{bmatrix} \quad (18)$$

The evolution of the magnetization vector $\vec{M}\left(\overrightarrow{r(t+\Delta t)}, t+\Delta t\right)$ was recorded for every step Δt.

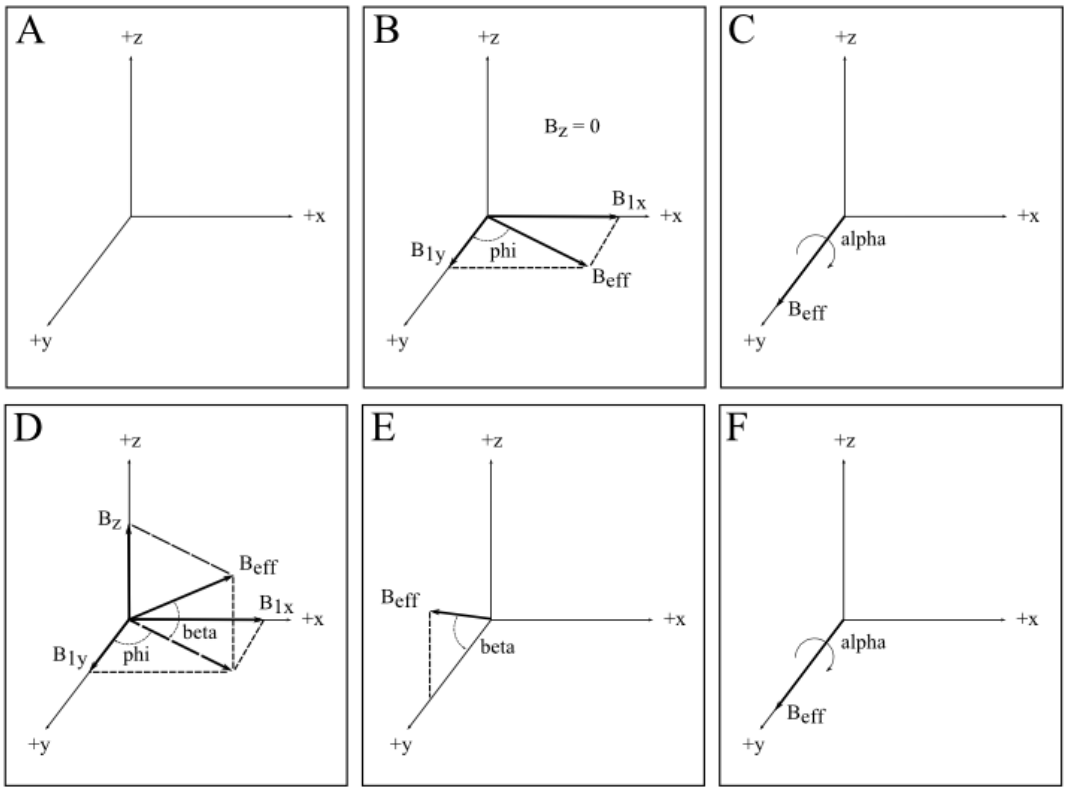

Fig. 2.2. Rotation of the spins about the B_{eff} vector: (A) The left-handed coordinate system is used by the MRISIMUL. (B - C) In the absence of a B_{eff} component along the z-axis, B_{eff} is rotating towards the +y axis through an angle phi and followed by an alpha rotation about the +y axis. (D – E) In the presence of non-zero components of B_{eff} along all three axes, the phi rotation about the +z axis is followed by a beta rotation towards the +y axis. Then, an alpha rotation about the +y axis was applied. In all cases where rotations about an axis on the transverse plane occurred, inverse rotations were applied so as to bring the B_{eff} to its initial direction.

2.5 Motion models

The introduction of motion involved the update of spatial coordinates of the isochromats for every single time step of the pulse sequence. Equations 1 and 9 show that MRISIMUL take into account the time dependence of the isochromat spatial coordinates. In this work, three different motion models were examined: respiratory motion, heart motion and blood flow.

a. Respiratory motion

Initially, respiratory motion of the diaphragm was simulated. Previous studies [2, 3], have demonstrated that the motion of the diaphragm due to breathing in most cases can be described by periodicity and asymmetry. In this study, we applied the respiratory model described by Lujan et al. [4], which approximates the position of the diaphragm as a function of time based on

$$z(t) = z_0 - b\left(cos\left(\frac{\pi t}{T} - \varphi\right)\right)^{2n} \quad (19)$$

where z_0 was the position of diaphragm during end exhalation, z_0-b was the position of diaphragm during end inhalation, T was the period of the breathing cycle, n defined the steepness and flatness of the motion model and φ defined the initial phase of the breathing cycle.

b. Heart motion

For simulating heart motion, a mathematical model of myocardial deformation was applied on a cylinder [5, 6]. This model accounted for longitudinal and radial contraction, axial torsion and rigid body displacement. The displacement in every point of the cylinder was computed by the following analytical expression in cylindrical coordinates [5]:

$$r = \sqrt{r_i^2 + \frac{(R^2 - R_i^2)}{\lambda}} \quad (20)$$

$$\theta = \phi R + \Theta + \gamma \mathrm{Z} + \epsilon \quad (21)$$

$$z = \omega R + \lambda \mathrm{Z} + \delta \quad (22)$$

where (r, θ, z) described the position of tissue in the deformed state and (R, Θ, Z) its position prior to deformation, R_i and r_i were the inner radii of the cylinder before and after the deformation respectively, λ described the longitudinal contraction, φ described the R-Θ shear, γ described the axial torsion, ε described the rigid body rotation, ω described the R-Z shear and δ the rigid body displacement.

c. Blood flow

Laminar flow of a homogeneous liquid within a straight tube was simulated. The displacement in every point of the liquid was simulated along one axis only and it was expressed as a function of time by the following equation:

$$z(t) = z_0 + vt \quad (23)$$

where z_0 described the initial position of the point and v described its velocity according to the velocity profile for laminar flow. The velocity profile of laminar flow within the tube as a function of the radius r was given by:

$$v(r) = v_{max}\left(1 - \frac{r^2}{R^2}\right) \quad (24)$$

where v_{max} was the maximum velocity of the liquid at the center of the tube and R was the inner radius of the tube.

2.6 Signal acquisition

Quadrature MR signal acquisition was achieved by detecting the complex magnetization vector of the object which generates a complex signal $S(t)$. In MRSIMUL, the signal $S(t)$ was calculated by the summation of the magnetization of all the spins within the object according to the following equation:

$$S(t) = S_R(t) + iS_I(t) = K(\vec{r})\left[\sum_{VOLUME}\left(\vec{M}(\vec{r},t)\vec{x}\right) + i\sum_{VOLUME}\left(\vec{M}(\vec{r},t)\vec{y}\right)\right] \quad (25)$$

where $K(\vec{r})$ allows for the introduction of receiver coil sensitivity maps in the simulator.

The acquisition vector in the pulse sequence matrix defined the time steps when the $S(t)$ signal would fill lines of the k-space matrix. Thermal noise was added to the real and imaginary components in k-space. A Fermi function k-space data filter was applied to reduce signal apodization artifacts prior to 2D Fourier Transformation.

2.7 Computational demands, assumptions and simplifications

In most cases, MRI simulators make several assumptions in order to reduce the complexity of the problem so as to execute the simulation within reasonable times. Otherwise, artifacts may be generated. MRISIMUL made no such assumptions and solved the entire complex problem for densely spaced isochromats and for a densely spaced time axis.

The total computational time of simulation platforms was proportional to the product of the total number of time steps of the execution of the pulse sequence times the total number of isochromats within the computer model. The total number of time steps was defined by the following equation:

$$T = \frac{D}{\Delta t} \quad (26)$$

where D was the total duration of the pulse sequence and Δt was the simulation kernel execution time-step.

For a given experiment, the total computational time of the simulation could be reduced by either decreasing the total number of isochromats within the computer model or by increasing the kernel execution interval Δt. However, prior research [7, 8] has shown that a smaller number of isochromats within the computer model and/or an insufficient frequency spacing among neighboring isochromats may produce truncation artifacts and/or spurious non-realistic echo refocusing.

Prior to initiating an experiment, MRISIMUL examined whether the imaging gradients would impose a phase difference of more than 180 degrees among neighboring isochromats. It was assumed that the frequency modulation within the object was only imposed by the imaging gradients and that there was a uniform distribution of magnetization within the object. If the phase difference was found to be greater than 180 degrees then actions were taken to prevent spurious echo formation as follows: either the isochromat density was increased along the problematic direction or the applied gradient amplitude was reduced (only if the pulse sequence design allowed for it). According to Shkarin et al. [9], this approach should result in an error of less than 3.5% among the exact signal and the approximate signal of the simulation experiment. However, even in the case where isochromat frequency spacing was sufficient, a small number of isochromats may have led to truncation artifacts, mainly due to the finite extent of isochromat frequencies within the object. Previous work [7, 8] suggested different numbers with respect to the isochromats needed to avoid such artifacts. Last, it should be noted that a wide receiver bandwidth results in shorter dwell times and therefore shorter kernel execution time-step Δt. This in turn may increase the computational load of the simulator and result in long execution times. These execution times could become prohibitively long for some simulations (e.g. CINE pulse sequences). The maximum volume size of the isochromats may also need to be further decreased as a result of myocardial strain, which introduces a variable isochromat density over time.

Taking into account all the aforementioned, a simulation experiment with a high isochromat density and a high temporal resolution would be closer to a real experiment but at the expense of the computational performance of the simulator. Though, the utilization of GPU-based parallel computing allowed the full application of MRISIMUL in large-scale analysis without model simplifications. However, in simulations that involve real crushers and anatomical model deformations further consideration needs to be taken, as it will be discussed in a following section.

Finally, MRISIMUL allowed its user to further reduce execution times by utilizing another form of the algorithm (“fast algorithm”). If recording the entire the magnetization vector for every time–step was not a requirement then Δt might be temporarily prolonged during kernel execution only if no temporal changes occurred within the pulse sequence or the anatomical model (cases with no motion) and only if the acquisition window was off. This resulted effectively in a variable Δt. Figure 2.3 illustrates how the fast algorithm was implemented. For MR simulations involving motion models, the “fast algorithm” was

updated so that longer time steps could be used when no RF and no static field gradients were applied. Fig. 2.4 illustrates how the variable time step size was implemented in MR simulations that involve motion.

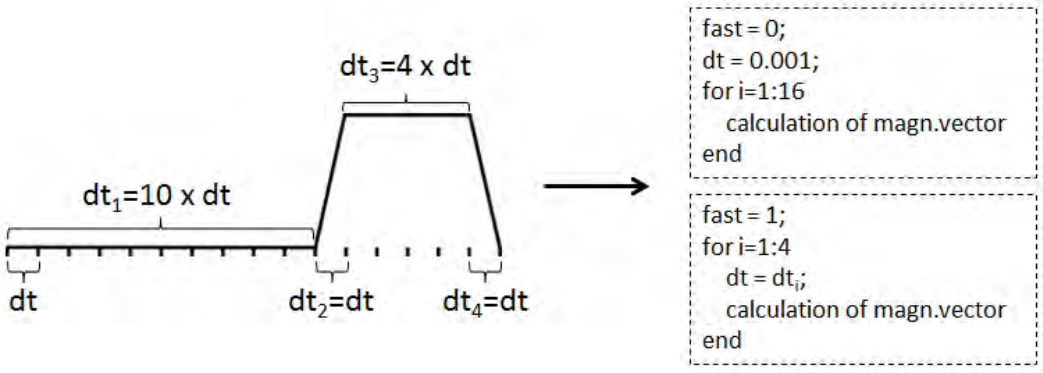

Fig. 2.3. The utilization of the fast algorithm may reduce the total number of time-steps and as a result the total execution time. This is done by prolonging the temporal step Δt when no temporal changes occur within the pulse sequence and when the acquisition window is off.

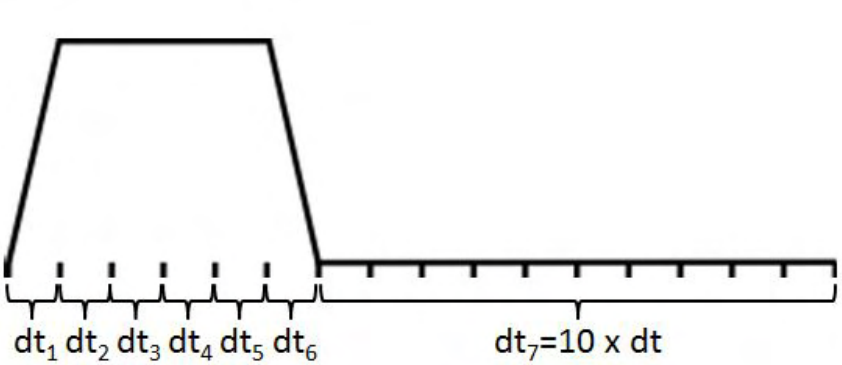

Fig. 2.4. Prolongation of the time step size can be done 1. when no temporal changes occur within the pulse sequence, 2. when the acquisition window is off, 3. when no pulses (RF and gradient) are on and 3. when no magnetic field inhomogeneities are considered.

References

[1] J. Bittoun, J. Taquin, and M. Sauzade, "A computer algorithm for the simulation of any nuclear magnetic resonance (NMR) imaging method," *Magn Reson Imaging,* vol. 2, pp. 113-20, 1984.

[2] J. M. Balter, R. K. Ten Haken, T. S. Lawrence, K. L. Lam, and J. M. Robertson, "Uncertainties in CT-based radiation therapy treatment planning associated with patient breathing," *Int J Radiat Oncol Biol Phys,* vol. 36, pp. 167-74, Aug 1 1996.

[3] H. D. Kubo and B. C. Hill, "Respiration gated radiotherapy treatment: a technical study," *Phys Med Biol,* vol. 41, pp. 83-91, Jan 1996.

[4] A. E. Lujan, E. W. Larsen, J. M. Balter, and R. K. Ten Haken, "A method for incorporating organ motion due to breathing into 3D dose calculations," *Medical Physics,* vol. 26, pp. 715-720, 1999.

[5] S. R. Tecelao, J. J. Zwanenburg, J. P. Kuijer, and J. T. Marcus, "Extended harmonic phase tracking of myocardial motion: improved coverage of myocardium and its effect on strain results," *J Magn Reson Imaging,* vol. 23, pp. 682-90, May 2006.

[6] A. A. Young and L. Axel, "Three-dimensional motion and deformation of the heart wall: estimation with spatial modulation of magnetization--a model-based approach," *Radiology,* vol. 185, pp. 241-7, Oct 1992.

[7] R. K. Kwan, A. C. Evans, and G. B. Pike, "MRI simulation-based evaluation of image-processing and classification methods," *IEEE Trans Med Imaging,* vol. 18, pp. 1085-97, Nov 1999.

[8] P. Shkarin and R. Spencer, "Direct simulation of spin echoes by summation of isochromats," *Concepts in Magnetic Resonance,* vol. 8, pp. 253 - 268, 1996.

[9] P. Shkarin and R. Spencer, "Time domain simulation of fourier imaging by summation of isochromats," *International Journal of Imaging Systems and Technology,* vol. 8, pp. 419 - 426, 1998.

3. IMPLEMENTATION

In short, the simulation wrapper and user interface were developed in MATLAB (The Mathworks Inc., Natick, MA) while the computationally demanding core services were developed in CUDA-C (NVIDIA, Santa Clara, CA). As previously mentioned, the simulation computational kernel was executed in parallel within the graphic processing unit (GPU) environment, exploiting the fact that Bloch equations can be solved independently for every isochromat of the anatomical model.

3.1 MATLAB wrapper

The basic MR imaging processes (i.e. pulse sequence programming, anatomical model development, reconstruction etc.) were implemented within the MATLAB environment. The pulse sequence was developed by functions which implemented different events e.g. gradient pulses, RF pulses, etc. For instance, gradient imaging pulses were implemented as trapezoids with user configurable total duration, plateau strength and the ramp duration, similarly to real-life pulse sequence development environments. The RF pulse design included four sample RF pulses: rectangular-shaped, sinc-shaped, adiabatic and user defined shapes. The RF pulses were described at minimum by total duration, flip angle and rotation axis. More attributes were used so as to define special RF pulses (e.g. number of cycles for sinc pulses, the frequency modulation for adiabatic RF pulses etc.).

3.2 Simulator design

The simulation computational kernel, which constituted the core of the simulation platform, executed the matrix operations for the calculation of the object's magnetization vector. As previously described, this involved multiplication and summation of large matrices whose values could change over time. Moreover, realistic simulation experiments required high temporal resolution for the execution of the pulse sequence and high isochromat density of the computer model. Therefore, to mitigate the long simulator execution times, this approach demanded high computational power in terms of floating-point operations per second (FLOPS) and memory usage. The highly-parallelizable nature of the simulation kernel was compatible with new hardware solutions that offer higher computational power through a parallel computing environment. For this purpose, the computational kernel of MRISIMUL was written in CUDA-C and was executed in parallel within the graphic processing unit (GPU) environment.

Prior to executing kernel computations within the CUDA environment, the entire pulse sequence was transferred to the global memory of the GPU. The pulse sequence was represented as a matrix of 6 row vectors. The first five row vectors handled the time evolution of the pulse sequence events, that is, the RF excitation on x and y axis and the gradients amplitude along the three axis. The sixth row represented the off/on state (0 or 1 respectively) of the RF receiver. Along with the pulse sequence, the matrix holding the MR characteristics of the different tissues within the 3D anatomical model (i.e. T1 and T2

relaxation times, proton density and chemical shift) was also transferred to the global memory of the GPU.

The number of times that the kernel was called was determined by an algorithm that divided the anatomical model evenly in space based on the available resources of the GPU card (e.g. number of multi-processors, global and shared memory) as well as based on benchmarking results of previous experiments. In every call of the kernel a different part of the object was transferred to the global memory of the GPU card and was simulated. This involved the transfer of the following matrices: the spatial coordinates of the isochromats, the id of the tissue where these isochromats belonged, the components of the magnetization vectors and finally the matrices that described the inhomogeneity field map and RF pulse sensitivity map object. In the cases of motion-related simulations, depending on the type of motion being simulated, additional matrices were transferred to the global memory of the GPU board. In particular, for flow simulations, a matrix containing the velocities of the isochromats within the specific part of the object, based on equation (24), was transferred to the GPU. For heart motion, the matrices containing the spatial coordinates of both the undeformed and deformed states of the heart model were transferred to the global memory of the GPU.

Then, for every kernel call, the magnetization vectors were computed independently during pulse sequence execution. For each time-step of the pulse sequence that belonged to the acquisition window, the components of the magnetization vector of each isochromat were stored in shared memory and a fully unrolled reduction process was performed. The product of the reduction was stored to a matrix that had been previously allocated within the global memory of the GPU. At the end of every kernel call this matrix was transferred to the host CPU, it was added to the previously computed raw signals of the other parts of the object and the k-space matrix of the object was formed (Fig. 3.1). As an example, a kernel call of 56 blocks of 128 threads each resulted in dividing the anatomical object in groups of 7168 isochromats each. The number of blocks was chosen to be an integer multiple of the available GPU multiprocessors while the number of threads per block was being limited by the available shared memory per block that would store the components of the magnetization vector of each isochromat prior to the reduction process.

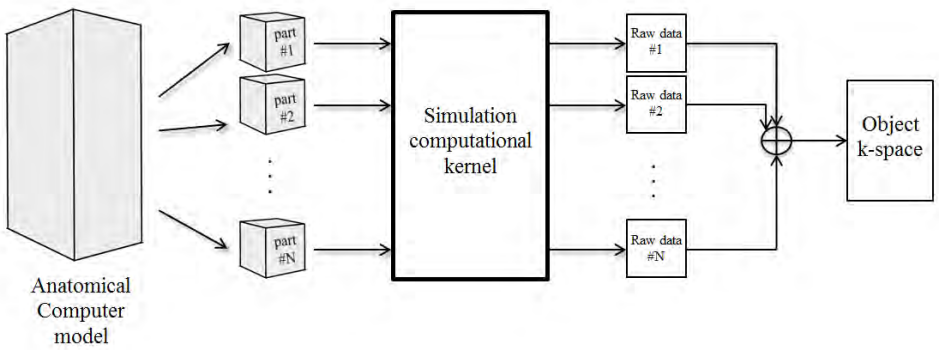

Fig. 3.1. Dividing the anatomical computer model on the computational kernel of the simulation platform: A different part of the computer model was simulated in every kernel call. The size of the group of isochromats being used in every kernel call was determined by the available GPU resources and the kernel design (see section 3.3). At the end of the kernel execution, the resulting signals of every call were summed together so as to form k-space.

3.3 Parallel computing and GPU systems

Experiments were performed on three different computer systems. The first was a single-node system consisting of a server style computer of 2 hexa-core (Intel E5-2630, 2.30GHz) processors, 32 GB RAM and four Tesla C2075 GPU cards. The second was a desktop computer with 2 quad-core processors (Intel E5520, 2.27GHz), 48GB RAM and two Tesla C2070 GPU boards. The third was a multi-node system which included the two aforementioned computer systems. For this system, the two computers were connected via 1 Gbps Ethernet within the same subnet, resulting in a total number of 6 available GPU cards. One of the two single computers was set up as a job manager for distributing the computational load to the 6 available GPU cards and for reconstructing the total k-space and image. Fig. 3.2 shows the block diagram for both single-node and multi-node system architectures.

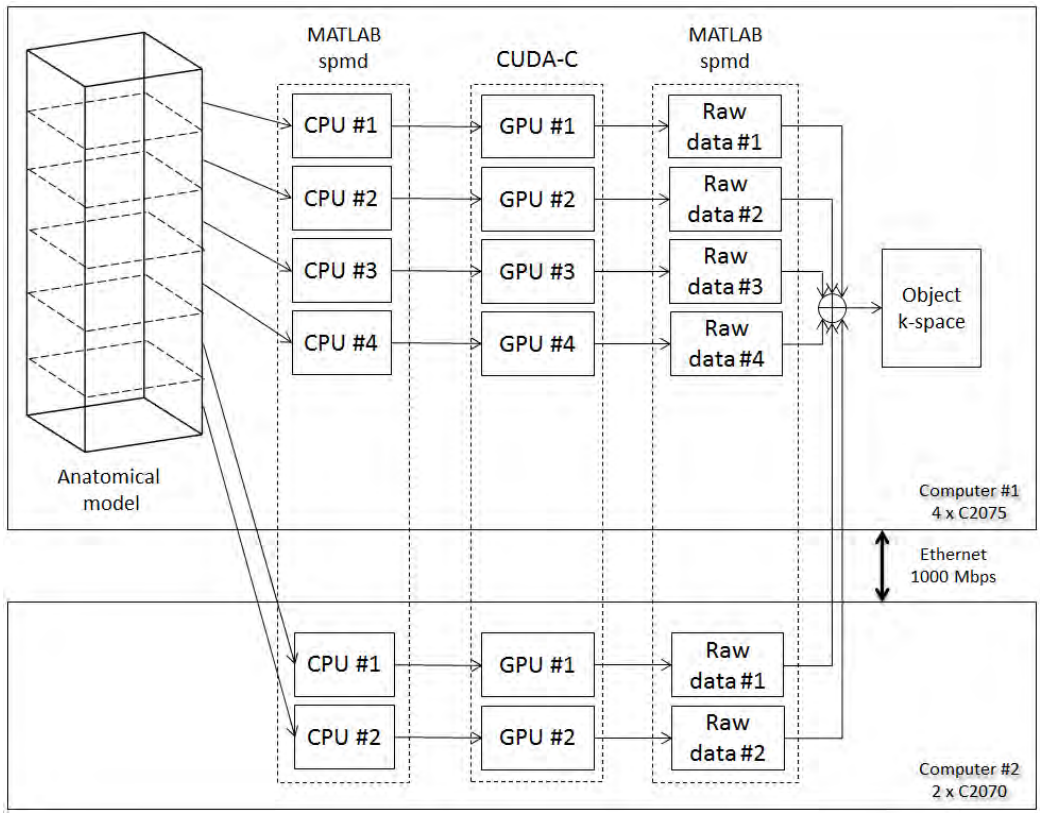

Fig. 3.2. Block diagram of the multi-node, multi-GPU approach. One of the two computers was set as a job manager. Each CPU core was assigned a part of the anatomical model and a certain GPU-card. A balanced load among the GPUs was implemented.

Both C2075 and C2070 GPU boards share the same architecture: global memory of 6GB GDDR5 and 448 GPU cores grouped into 14 streaming multiprocessors of 32 processors each. The Tesla C2070 was based on the NVIDIA Tesla T20 graphics processing unit whereas the Tesla C2075 was based on the NVIDIA Tesla T20A graphics processing unit [1, 2]. Both GPU cards had a compute capability of 2.0 and were targeted for high performance computing.

To demonstrate the efficacy of MRISIMUL on single-node and multi-node multi-GPU systems, MRISIMUL supported the MATLAB single-program-multiple-data (spmd) statement. The total number of the available GPU boards in a system defined the processes spawned, each one accessing a different CPU. A GPU board along with a part of the anatomical object were assigned to each process and, in the end, the outputs of each process were summed together so as to form the total k-space matrix of the object. A balanced load among the GPUs was implemented in this implementation.

3.4 Stationary anatomical computer models

In this work, the MRISIMUL simulator platform simulated imaging of three different anatomical computer models: a user-defined phantom, a human brain model and a human heart model.

The simplest computer model was a user-defined phantom. By means of a simple Graphical User Interface (GUI) the user could develop three-dimensional rectangular objects, place them in 3D space and assign to each of them unique MR characteristics (T1, T2, ρ, chemical shift). The total number of isochromats was variable and allowed for the development of non-isotropic voxels.

The brain anatomical model used for simulations is a brain phantom available online by the McConnell Brain Imaging Center at McGill University (Montreal, Canada) [3, 4]. It is a three-dimensional digital brain phantom that describes the “fuzzy” spatial distribution of 11 different tissue types: grey matter, white matter, cerebrospinal fluid, fat, tissue around fat, muscles, muscle and skin, skull, vessels, dura matter and bone marrow. Based on this model, every tissue was accompanied by its own relaxation times T1 and T2 whereas the fraction of tissue within the voxel defined its proton density and allowed for simulations of multi-component voxels. Moreover, a discrete version of the brain phantom supported only one tissue per voxel and was used to illustrate the spatial relationships between the different tissues of the brain.

Last, a detailed 3D computer model of the anatomy of the human heart was also developed. This model was based on the dataset of high resolution images provided by the Visible Human Project (VHP) of the United States National Library of Medicine (Bethesda, MD). The Visible Human Project dataset was generated from male and female cadavers and includes MR, CT as well as true color high resolution axial anatomical images of the human body [5]. A semi-automatic segmentation algorithm was applied to the dataset of the female cadaver and a computer model of the human heart was extracted. The heart computer model had a voxel size of 0.33mm x 0.33mm x 0.99mm and consisted of more than 8 million voxels. It was based on a discrete classification and included only one tissue per voxel. An illustration of the three dimensional computer model of the heart is given in Fig. 3.3.

Fig. 3.3. Illustration of the 3D computer model of the human heart produced from the Visible Human Project [5].

3.5 Motion models

For respiratory motion simulations, a homogeneous 3D user-defined phantom was used. The size of this 3D model was of 10cm x 2.5cm x 20cm. The isochromat volume size was 0.5mm x 0.5mm x 1mm resulting in 2000000 isochromats within the object. The object was placed in space and translational motion was induced on the y-axis (Phase Encoding axis) only, according to equation 19. In this motion model, z_0 and b were set equal to 0.012m, ϕ was set to 0 and n to 3. According to Lujan et al. [6] these values represent an average observed diaphragm motion. A period T of 4 sec (respiratory rate = 15 breaths per minute) was selected for the breathing cycle. The position of the diaphragm is illustrated in figure 3.4.

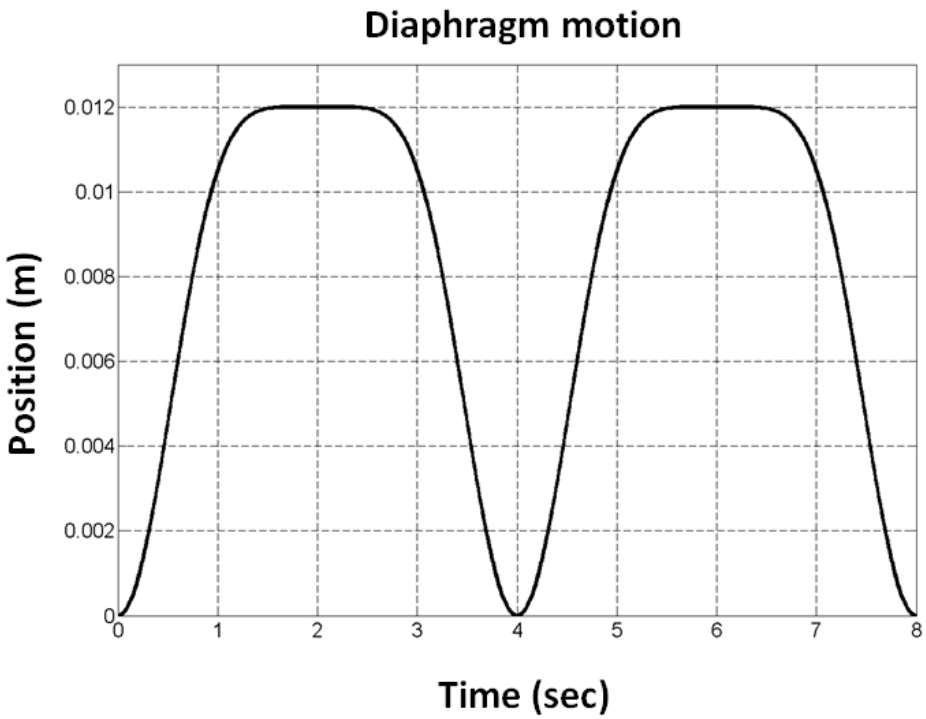

Fig. 3.4. Respiratory model simulations: Illustration of the temporal evolution of the position of the diaphragm for two consecutive breathing cycles.

The computer model used for heart motion simulations was a homogeneous cylindrical 3D user-defined object. The size of the cylinder was 100 mm x 100 mm x 80 mm, the inner diameter was 50 mm and the isochromat volume size was 0.25mm x 0.25mm x 1mm. Based on this configuration, the total number of isochromats was 7632792. T1 and T2 relaxation times were set to 0.9sec and 0.05sec respectively. The deformation of the cylinder followed equations (20-22) where the values of model parameters were chosen so as to represent human heart motion, according to Tecelao et al. [7]. In detail, the model parameters were set: λ = 1, ϕ = 0.556, γ = 0.6, ε = 18.334, ω = 0.278 and δ = 4.167.

The cylinder of the myocardial model was initialized in the undeformed state (i.e. simulating cardiac end diastole). A linear interpolation of the cylindrical coordinates between the initial and the deformed state was performed for every timestep of the cardiac cycle. Figure 3.5 shows the time evolution for an inner radius R_i of 25mm in the undeformed cylinder and an inner radius r_i of 10mm in the deformed cylinder (total duration 0.8 sec).

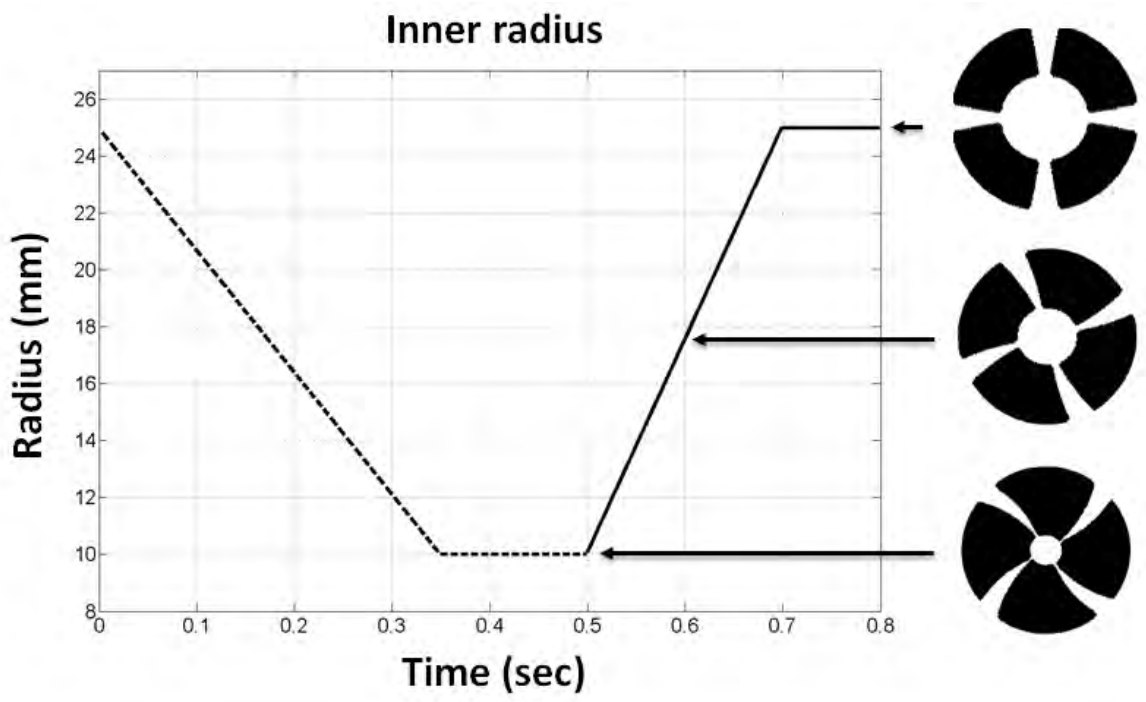

Fig. 3.5. Cardiac model simulations: plot of the temporal evolution of the inner radius of the heart motion model. Systole is shown with the dashed line whereas diastole is shown with the solid line. The corresponding phases of the heart model are shown on the right. The artificial radial white line tags have been placed intentionally to demonstrate the deformation of the heart model throughout the cardiac cycle

Last, for flow simulations, a custom 3D object of two cylinders, one inside the other, was used (figure 3.6). In this anatomical model, the outer cylinder had a diameter of 10cm and length of 1m whereas the inner cylinder had a diameter of 5 cm and length also 1m. The outer cylinder was stationary whereas the inner cylinder represented fluid with particles flowing along the longitudinal axis of the cylinder with a fixed velocity profile given by equation (24). The maximum velocity of the liquid at the center of the inner cylinder was set to 0.05 m/sec, which is within the range of aortic blood flow velocities during cardiac ejection [8]. The 3D velocity profile of laminar flow is illustrated in figure 3.6. The isochromat volume size was 0.25mm x 0.25mm x 1mm resulting in 30779693 isochromats. The outer cylinder had relaxation times T1 and T2 of 0.9sec and 0.05sec respectively. The corresponding values of the inner cylinder were set to 2sec and 0.4sec respectively.

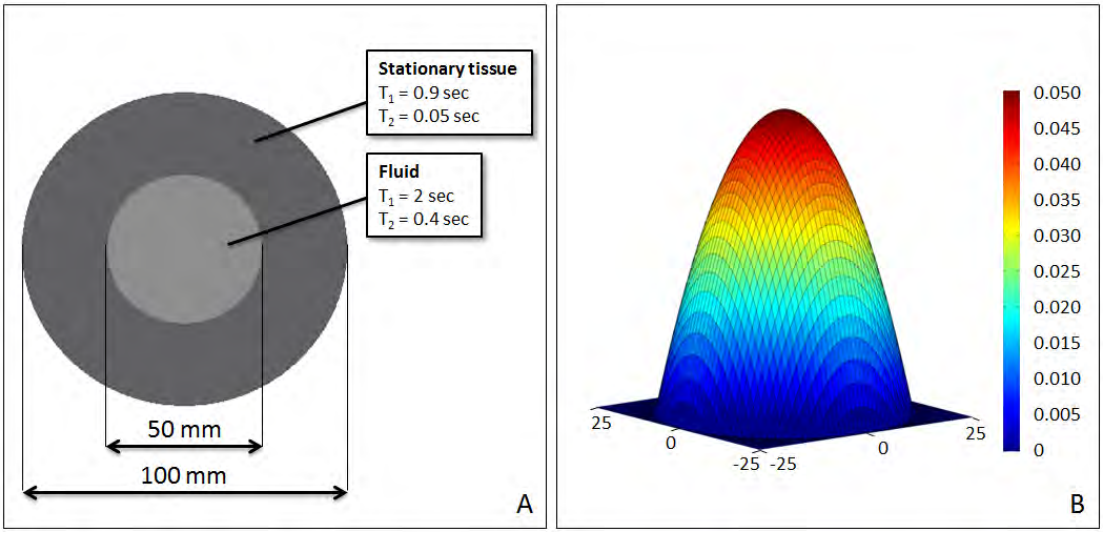

Fig. 3.6. Flow simulations. (A) Illustration of the geometry of the flow anatomical model. The outer cylinder represents stationary tissue whereas the inner cylinder represents the fluid. (B) Laminar velocity profile of the fluid within the cylindrical flow model.

3.6 Non-kernel embedded motion models

The motion models described by equations (19) through (24) were embedded in the design of the CUDA kernel as previously described. However, motion models may not always be available in an analytical expression, for example, particle flow measure recorded from patients. By definition, such measurements cannot be embedded in the design of the CUDA kernel. Even when analytical expressions of motion are available, in many cases it may be desirable not to redesign the CUDA kernel in order to embed a particular analytical motion expression to it. In these cases, the motion model may become a separate input to the kernel as a “non-predefined motion model”. In these cases, the motion model would need to be loaded or calculated directly on the host CPU prior to its transfer to the global memory of the GPU. Such non-predefined motion models were also tested. This approach required large amounts of memory space in both CPU and GPU since motion (i.e. the position of all the isochromats within the 3D anatomical object) needed to be saved for every time step of the pulse sequence. The limited memory space in both CPU and GPU required that the anatomical object and the pulse sequence be divided and that the kernel be called multiple times more than with the use of kernel embedded motion models.

3.7 Pulse sequences

The simulations were based on a Gradient Echo (GE) pulse sequence applied to all the available anatomical and motion models of this study. A segmented CINE pulse sequence was applied on respiratory and heart models with motion. For both pulse sequences, the static magnetic field strength was set to 1.5 Tesla, the temporal resolution of the simulator was set to 10μsec, the bandwidth of the receiver was 50 kHz, the slice thickness was 10mm

and the k-space matrix was 256x256. The spin system was brought to steady state by means of dummy excitations prior to recording any signal. For respiratory motion simulations the FOV was set to 150 mm x 150 mm whereas for heart motion simulations it was set to 360 mm x 270 mm.

The GE pulse sequence was applied with a three lobe sinc-shaped RF pulse of 15^{o} and 2msec duration and a TE/TR of 4.5/50 msec. Gradient crushers were introduced at the end of each TR. This configuration of the GE pulse sequence resulted in a total number of 1280000 time-steps which were reduced to 210944 by utilizing the fast algorithm, previously described in Section "Computational demands, assumptions and simplifications". However, the fast algorithm was not activated in motion-related simulations that involved temporal changes of the anatomical model due to translational/rotational motion and/or deformation.

For flow model simulations a bipolar velocity-encoding (venc) gradient pair was introduced after the RF excitation pulse. The bipolar gradient was applied along the slice direction with a venc of 10 cm/sec. The velocity encoded GE pulse sequence was run twice with different polarity of the venc gradient pair. The velocity encoded phase image was computed from the two complex images.

In the cases of respiratory and heart motion simulations, the CINE pulse sequence was based on the same design as the GE pulse sequence with a flip angle of 40^{o} and TR of 8msec. The acquisition was segmented along the respiratory and cardiac cycles with different combinations of phases per cycle and views per segment (vps).

Last, myocardial tagging was also investigated. For this purpose, spatial modulation of magnetization (1-1 SPAMM) [9] was introduced upon detection of the R-wave in the CINE pulse sequence with 100 phases and 1 vps. The two SPAMM RF pulses were three-lobe sinc-shaped with a flip angle of 45^{o}, 3msec duration. The gradient pulse was a rectangular-shaped gradient pulse of 0.6msec duration and 0.004T/m strength. Gradient crushers were introduced at the end of the 1-1 SPAMM preparation (see next section).

3.8 Pulse sequence design and computational demands

As described before, the imaging gradients may induce a phase difference of more than 180^{o} among neighboring isochromats within the anatomical object, which in turn may produce spurious nonrealistic echo refocusing. Based on the design of the pulse sequence, MRISIMUL set the maximum isochromat size so that there was no image quality degradation due to the introduction of strong crushers into the pulse sequence. Such strong gradients result in artificial spurious echoes and degradation of the final image quality due to low isochromat density. Avoidance of this effect required decreasing the isochromat size within the anatomical object. This in turn increased the computational load of the simulation and resulted in long execution times. These execution times could become prohibitively long for some simulations (e.g. CINE pulse sequences). The maximum volume

size of the isochromats may also need to be further decreased as a result of myocardial strain, which introduces a variable isochromat density over time.

For this purpose, the simulator allowed the introduction of another form of crushers, namely “software crushers”. In this case, the software crushers involved the transfer to the GPU card of an extra matrix consisting of 0s and 1s pointing out when the kernel should induce nullification of the transverse components of every isochromat within the anatomical object. The introduction of software crushers may result in not allowing the appearance of some realistic artifacts and contrast (e.g. stimulated echo based artifacts and contrast). However, software crushers allow for larger isochromats volumes and reduced computational load and, therefore, reduced GPU resources. This allows for the execution of otherwise prohibitively long simulations. In any case, it is up to the MRISIMUL user to decide whether to compromise Stimulated Echo artifacts and contrast for faster execution times.

In the next section, for each simulation experiment it will be mentioned the utilization of software crushers or not.

Finally, as an example, the variable time step “fast algorithm” was applied for Gradient Echo and CINE pulse sequences with respiratory motion so as to measure the potential speedup gain. Moreover, for educational purposes, where speed is needed, the problem complexity should be limited to two dimensions (including a 2D motion model). For this purpose, as an example, the Gradient Echo pulse sequence simulation with respiratory motion was confined to two dimensions and the kernel execution times were recorded with and without the variable time step “fast algorithm”.

3.9 Simulations

1. Simulator performance

Prior to all the experiments, a benchmarking test was performed so as to ascertain the optimum combination of blocks and threads per block. For this test, the NVIDIA Visual profiler [10] was used and the kernel execution time was recorded for the application of a pulse sequence of 220800 time steps on a 3D object of 500000 isochromats. The number of blocks was always selected to be an integer multiple of the 14 streaming multiprocessors (available on both GPU card models) whereas the number of threads per block was always selected based on the available shared memory size per block [11]. Since the two different GPU cards models (C2070 and C2075) shared the same architecture, the resulting combination of blocks and threads per block was kept for all the experiments throughout this paper.

To demonstrate the high computational power of GPU-based simulations, execution times of the MRISIMUL computational kernel were recorded for: 1) serially executed C-code of the kernel on the CPU 2) OpenMP parallel executed C-code of the kernel on the CPU with multi-threading (1, 2, 4 and 8 threads) and 3) CUDA-C parallel executed C-code of the kernel on the desktop computer utilizing one single C2070 GPU card. The simulations were based on the simulation of the simple GE pulse sequence of 210944 time-steps (fast algorithm enabled) applied to a 3D object of 1350000 isochromats.

To demonstrate how problem size relates to kernel execution times, four different test cases were considered. In the first test case, a computationally simple experiment was run where a small number of isochromats (3600000 isochromats) was combined with a small number of time-steps (72704 time-steps). In the second test case, a larger number of isochromats (14400000 isochromats) was combined with the same small number of time-steps (72704 time-steps) so that the simulator's performance with a higher spatial resolution could be examined. In the third test case, a small number of isochromats (3600000 isochromats) was combined with a large number of time-steps (210944 time-steps) so that the simulator's performance to increased temporal demands could be tested. Last, in the fourth test case, a large number of isochromats (14400000 isochromats) and a large number of time-steps (210944 time-steps) were combined so that the simulator's performance in a heavy computational environment could be assessed.

Additionally, to evaluate the efficacy of the fast algorithm on the total kernel execution time two extra test cases were considered. In both test cases, the abovementioned GE pulse sequence was run in a user-defined object of 3200000 isochromats. In the first test case, the fast algorithm was disabled and the total number of time-steps remained 1280000 whereas in the second test case the fast algorithm was enabled and the total time-steps were reduced to 210944.

2. Simulations on stationary anatomical computer models

The Gradient Echo pulse sequence was also applied to the anatomical models of the brain (k-space matrix 512x512, FOV 30 cm x 30 cm, slice thickness 20mm) and the heart (k-space matrix 256x256, FOV 40 cm x 40 cm, slice thickness 10 mm). In these simulations, a real gradient crusher pulse was applied along the slice axis at the end of each TR.

3. Simulations on motion models

The simple Gradient Echo pulse sequence was also applied to both respiratory motion model and heart motion model, with and without motion. A segmented CINE pulse sequence was also applied on respiratory and heart models with motion. For the respiratory model simulation, 250 phases and 2 vps, 125 phases and 4 vps and, last, 100 phases and 5 vps were tested. For the heart model simulation 100 phases and 1 vps, 50 phases and 2 vps and, last, 20 phases and 5 vps were tested. Software crushers were used in the simulations on motion models, where applicable.

Last, myocardial tagging was also applied on the heart motion model. 1-1 SPAMM preparation was introduced in the CINE pulse sequence with 100 phases and 1 vps upon the onset of the cardiac cycle. Software crushers were used at the end of the 1-1 SPAMM preparation.

4. Spurious echoes formation and image quality

The formation of spurious echoes due to reduced isochromats density within the object was investigated and their effects on image quality are presented. In this experiment and for demonstration purposes only, the GE pulse sequence (k-space matrix 256x256) was applied on an object of size 10 cm x 10 cm x 30 cm, slice thickness 10 mm and of isochromat volume size 1 mm x 1 mm x 1 mm (total number of isochromats 3000000). This experiment ran three times with different isochromat volume size (1mm x 1mm x 0.3mm, 1mm x 1mm x 1mm and 1mm x 1 mm x 5 mm) being selected each time. In these simulations, a real gradient crusher pulse was applied along the slice axis at the end of each TR.

5. Software crushers utilization

To demonstrate the effects on image quality and computational demands of realistic crushers (which impart phase to the isochromats) against software crushers, four test cases were considered with a CINE pulse sequence (100 phases and 5 vps) on the respiratory motion model. For all four test cases, the respiratory model consisted of only one layer of isochromats in the slice selection direction. All gradients along the slice direction were turned off so that spurious echoes could only be potentially induced by the gradient crushers along the remaining directions. In the first three test cases real crushers were introduced along the readout direction at the end of each TR. The crushers had duration of 0.5msec and areas equal to the area of the readout gradient. For the first test case, the isochromat volume size was set to 0.5mm x 0.5mm x 0.5mm (total number of isochromats: 10000). For the second test case, the isochromat volume size was set to 0.25mm x 0.25mm x 1mm (total number of isochromats: 40000). For the third case, the isochromat volume size was set to 0.1mm x 0.1mm x 1mm (total number of isochromats: 250000). This case represented the maximum isochromat volume in order to avoid the formation of spurious echoes, thus resulting in the minimum possible computational load for real crushers. For the fourth test case, software crushers were used instead of real crushers with an isochromat volume size equal to that of the first test case (0.5mm x 0.5mm x 1mm, total number of isochromats: 10000). After validating the use of software crushers with the aforementioned four test cases, software crushers were used for all remaining experiments described herein.

6. Multi-GPU performance

To test for scalability in multi-GPU experiments, the execution times of the computational kernel were recorded for a total of 1, 2, 3, 4, 5 and 6 GPU cards. For these tests, a GE pulse sequence with the respiratory motion model was used. The total number of time steps was 640000. The anatomical computer model consisted of 2000000 isochromats.

7. Non-kernel embedded motion model

To evaluate the performance of non-predefined motion models, the respiratory motion model was considered as a “non-predefined motion model” whose temporal evolution,

rather than being embedded in the kernel, was calculated directly on the host CPU prior to its transfer to the global memory of the GPU. This allowed for direct comparisons of execution times against the embedded model of the GE pulse sequence described earlier. This non-predefined motion model resulted in a total number of 640000 timesteps, 2000000 isochromats and required approximately 4TB of total memory space. Since the current MATLAB and GPU configuration allowed transfer of matrices with size less than 2GB, the object was divided in 36 parts and the pulse sequence in 80 parts of 8000 timesteps each. The total experiment size was thus divided 36*80 = 2880 times. Thus, the CUDA kernel was called 2880 times. For each kernel call, a dataset of 1.28GBs (figure 3.7) was transferred from the host to the GPU through a high-speed serial computer expansion bus (PCIe x16 Gen2, speed rate of 8GBs/sec). The total execution time of the kernel was recorded for different combinations of the available GPU cards on a single-node computer system.

8. Other applications

To demonstrate the enhanced capabilities of the MRISIMUL on different aspects of MR physics, which are usually of interest to programmers of MRI scanners, two more experiments were also run. In the first experiment, a 10msec inversion Silver-Hoult hyperbolic secant adiabatic RF pulse was applied to a single isochromat on resonance and a 3D vector representation of this isochromat was acquired. In the second experiment, the slice profiles of a rectangular-shaped RF pulse and a three lobes sinc-shaped RF pulse on a 3D object were explored (object size 30 cm x 10 cm x 30 cm, isochromat volume size 1 mm x 1 mm x 2.5mm, k-space matrix 256 x 256, FOV 40 cm x 40 cm, slice thickness 100 mm). For this purpose, the Gradient Echo pulse sequence was altered to accommodate slice selection along the Measurement/Frequency direction (the duration of both the RF pulses was 3 ms duration). In this simulation, a real gradient crusher pulse was applied along the slice axis at the end of each TR.

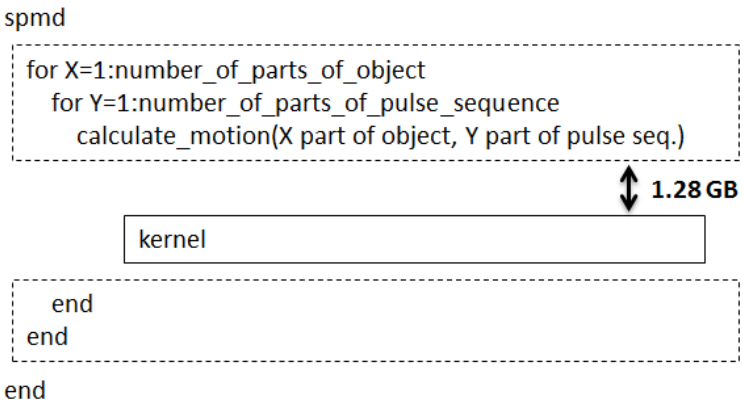

Fig. 3.7. Non-kernel embedded motion models algorithm. The object was divided in X parts and the pulse sequence in Y parts. Every kernel call was accompanied by a transfer of 1.28 of data to the GPU global memory through the high-speed serial computer expansion bus (PCIe x16 Gen2, speed rate of 8GBs/sec).

References

[1] NVIDIA. (2010, Tesla C2050 and C2070 computing processor board Available: http://www.nvidia.com/docs/IO/43395/Tesla_C2050_Board_Specification.pdf

[2] NVIDIA. (2011, Tesla C2075 computing processor board. Available: http://www.nvidia.com/docs/IO/43395/BD-05880-001_v02.pdf

[3] B. Aubert-Broche, M. Griffin, G. B. Pike, A. C. Evans, and D. L. Collins, "Twenty new digital brain phantoms for creation of validation image data bases," *IEEE Trans Med Imaging,* vol. 25, pp. 1410-6, Nov 2006.

[4] D. L. Collins, A. P. Zijdenbos, V. Kollokian, J. G. Sled, N. J. Kabani, C. J. Holmes*, et al.*, "Design and construction of a realistic digital brain phantom," *IEEE Trans Med Imaging,* vol. 17, pp. 463-8, Jun 1998.

[5] M. J. Ackerman, "The Visible Human Project: a resource for anatomical visualization," *Stud Health Technol Inform,* vol. 52 Pt 2, pp. 1030-2, 1998.

[6] A. E. Lujan, E. W. Larsen, J. M. Balter, and R. K. Ten Haken, "A method for incorporating organ motion due to breathing into 3D dose calculations," *Medical Physics,* vol. 26, pp. 715-720, 1999.

[7] S. R. Tecelao, J. J. Zwanenburg, J. P. Kuijer, and J. T. Marcus, "Extended harmonic phase tracking of myocardial motion: improved coverage of myocardium and its effect on strain results," *J Magn Reson Imaging,* vol. 23, pp. 682-90, May 2006.

[8] U. Elkayam, J. M. Gardin, R. Berkley, C. A. Hughes, and W. L. Henry, "The use of Doppler flow velocity measurement to assess the hemodynamic response to vasodilators in patients with heart failure," *Circulation,* vol. 67, pp. 377-83, Feb 1983.

[9] L. Axel and L. Dougherty, "MR imaging of motion with spatial modulation of magnetization," *Radiology,* vol. 171, pp. 841-5, Jun 1989.

[10] NVIDIA. (2013). *NVIDIA Visual Profiler*. Available: https://developer.nvidia.com/nvidia-visual-profiler

[11] C. G. Xanthis, I. E. Venetis, A. V. Chalkias, and A. H. Aletras, "MRISIMUL: A GPU-based Parallel Approach to MRI Simulations " *IEEE Transactions on Medical Imaging,* vol. PP, 2013.

4. RESULTS

4.1 Simulator performance

1. Benchmarking results

The kernel execution times, for the 24 different pairs of blocks and threads per block, are shown in Table 1, which identifies the optimum combination of blocks and threads. For the current design of the algorithm, the shorter kernel execution time was achieved for 112 blocks and 128 threads per block therefore resulting in simulating a total of 14336 isochromats per kernel call.

Threads/Block	64					
Blocks	14	28	56	112	224	448
Kernel time (sec)	154.6	82.3	47.0	33.5	33.4	37.2

Threads/Block	**128**					
Blocks	14	28	56	**112**	224	448
Kernel time (sec)	87.1	51.6	32.9	32.8	36.6	43.8

Threads/Block	256					
Blocks	14	28	56	112	224	448
Kernel time (sec)	64.0	33.4	33.5	37.4	44.7	59.1

Threads/Block	512					
Blocks	14	28	56	112	224	448
Kernel time (sec)	34.8	34.5	38.9	46.3	61.5	60.1

TABLE 1. Kernel execution times were compared among different combinations of blocks and threads/block for the application of a pulse sequence of 296000 time steps on a 3D object of 500000 isochromats. The shorter kernel execution time has been achieved for 112 blocks and 128 threads per block (shown in bold letters and gray-colored box).

2. GPU performance

To illustrate the performance of MRISIMUL, three different configurations (single thread CPU, multi-thread OpenMP CPU, GPU) were examined. As described earlier, the Gradient Echo pulse sequence run for 210944 time-steps on a 3D object of 1350000 isochromats. The execution times were recorded and are shown in Table 2.

CPU				**GPU**
	OpenMP			**1 x C2070**
1 thread	**2 threads**	**4 threads**	**8 threads**	
64421 sec	32340 sec	16582 sec	8771 sec	282 sec

TABLE 2. Execution times were compared among different approaches for the application of a pulse sequence of 210944 time-steps on a 3D object of 1350000 isochromats

It can be seen that the serially executed C-code on the CPU offered limited computing performance since computations involved multiplication/summation of large matrices whose values change in time. Using OpenMP parallel executed C-code on the CPU with multi-threading up to 8 threads produced an increased computing efficiency, though the runtimes were still long for realistic MRI simulations. Table 2 also shows that the GPU-based simulation recorded a kernel execution time of approximately 5 minutes for the same experiment. A speedup of about 228 times when compared to serially executed C-code on the CPU was observed. The speedup was between 31 to 115 times (Fig. 4.1) when compared to the OpenMP parallel executed C-code on the CPU, depending on the number of threads used in multithreading (2 to 8 threads). Figure 4.2 shows the identical simulated images of the sequential and GPU implementations along with their kernel execution times.

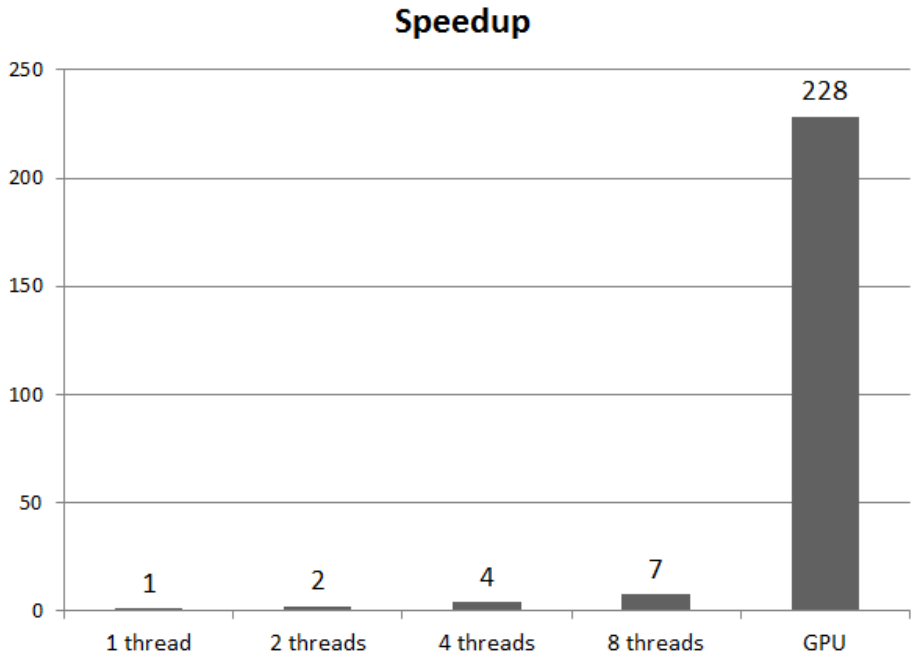

Fig. 4.1. Speedup of the OpenMP parallel executed C-code on the CPU and CUDA-C parallel executed C-code on a single GPU setup compared to serially executed C-code on the CPU.

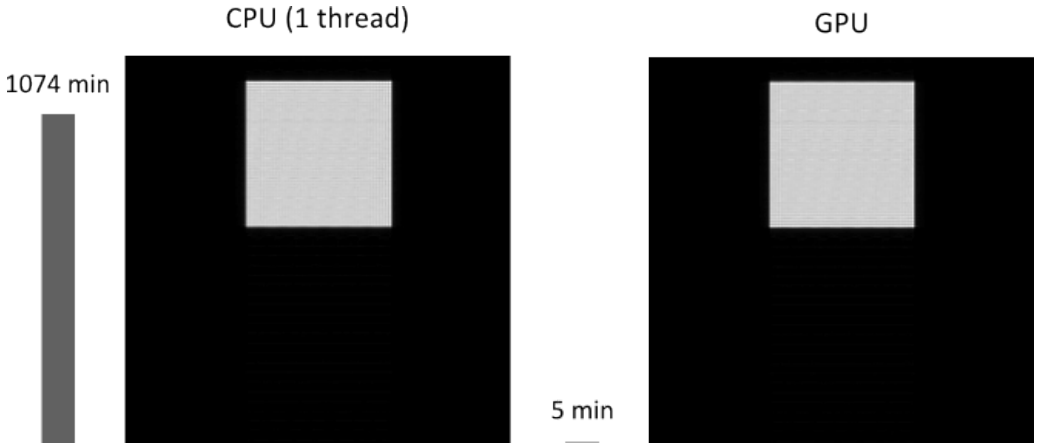

Fig. 4.2. Output simulated images of the sequential (left image) and GPU (right image) implementation along with their kernel execution times. No discernible differences were observed between the two images.

3. Problem size and MRISIMUL performance

To demonstrate the relationship between simulation size and kernel execution time of the MRISIMUL, four different test cases were examined with different combinations of the total pulse sequence time-steps and total anatomical object isochromats. Table 3 shows the execution times recorded in these test cases along with the change in the kernel execution time due to the added computational load. There was a linear relationship between kernel execution time and increasing number of isochromats. A relationship close to linear was also observed between kernel execution time and total number of time-steps. The simulator's response to problem size modifications is dependent on the computational path that the isochromats will follow during the pulse sequence execution (section 2.4) and it may slightly vary for different experiments.

		k-space size (total time-steps)		
		128x128 (72704)	256x256 (210944)	Change
iso	3600000	277 sec	749 sec	2.7x
	14400000	1106 sec	2991 sec	2.7x
	Change	4x	4x	

Table 3. Performance of the MRISIMUL kernel on different simulation sizes (total number of object isochromats and total number of pulse sequence time-steps). The increase in kernel execution time is also presented. The fast algorithm (section 2.6) was enabled for these experiments.

4. Fast algorithm

Last, the efficacy of the fast algorithm on the kernel execution time was examined. Table 4 displays the kernel execution times for the same simulation with and without the fast algorithm enabled. It should be noted that the observed speedup is highly dependent on the structure of the pulse sequence being used each time.

fast algorithm DISABLED (fast = 0)	**fast algorithm ENABLED (fast = 1)**
Total time-steps = 1280000	Total time-steps = 210944
1114 sec	281 sec

Table 4. Execution times for the same simulation of the MRISIMUL kernel with and without the fast algorithm.

4.2 Simulations on stationary anatomical computer models

Fig. 4.3 shows the results obtained with MRISIMUL with a gradient echo pulse sequence on the McGill Brain phantom. The three cardinal planes (transverse, coronal and sagittal planes) of the white matter tissue of the brain phantom are shown. Fig. 4.4 shows the results obtained with MRISIMUL with a gradient echo pulse sequence on the model of the heart developed from the Visible Human Project dataset. Fig. 4.4A to 4.4C illustrate the procedure followed for the development of the heart computer model whereas Fig. 4.4D depicts the simulated MR image that was acquired from the same slice location of the Visible Human Project.

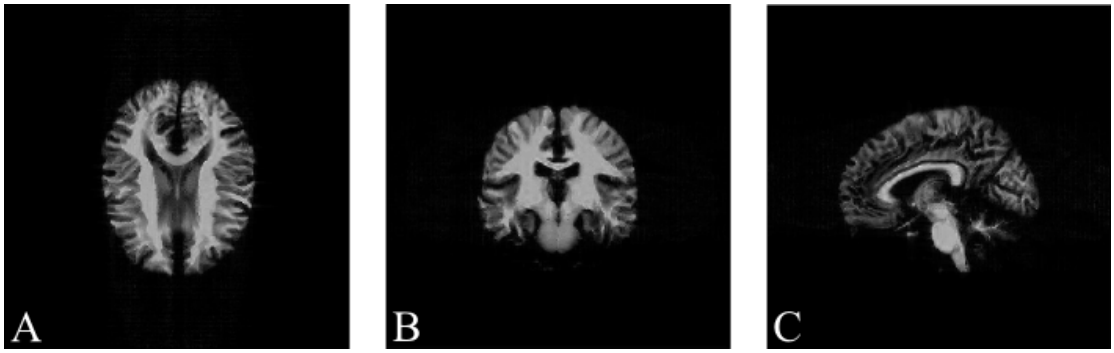

Fig. 4.3. Gradient echo images of the McGill Brain phantom. (A) Transverse, (B) coronal and (C) sagittal planes of the white matter tissue only. The CUDA kernel execution time was approximately 9 minutes for each image. (Total number of tissue isochromats 848028, total number of time-steps 685568, k-space matrix 512 x 512, FOV 30 cm x 30 cm, slice thickness 20mm). For each image the entire volume of the McGill Brain phantom was simulated (not just the slice of interest).

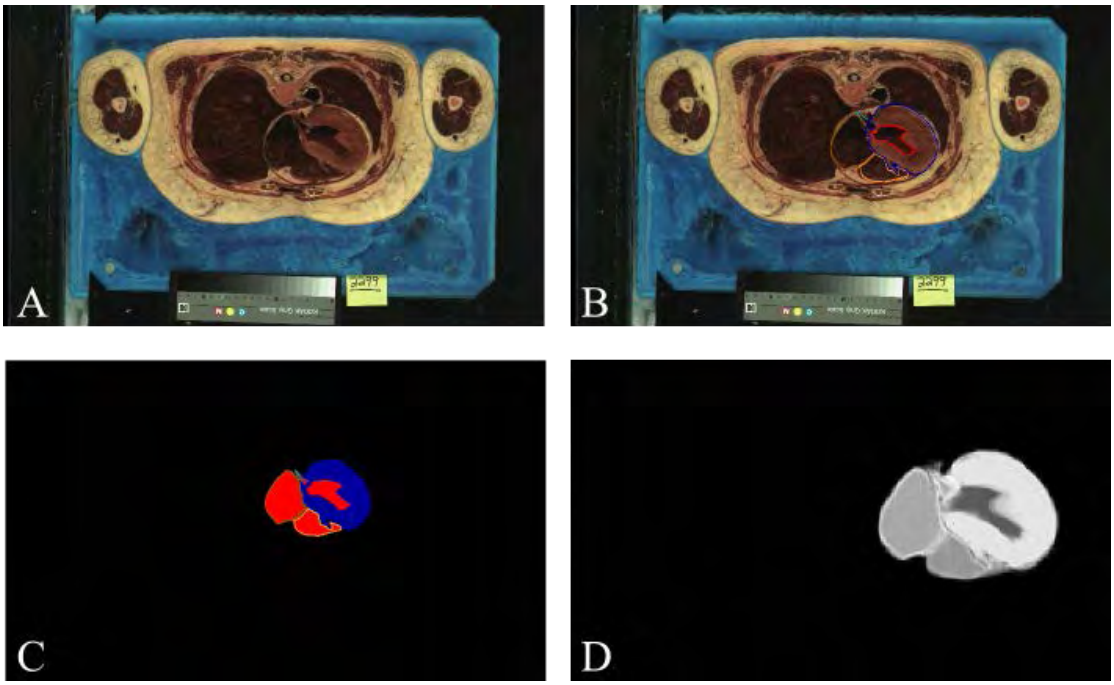

Fig. 4.4. (A) The VHP thorax image of the female cadaver (image avf1430a.png). (B) Semi-automatic segmentation of the Left Ventricle (blue), Left Atrium (light blue), Right Ventricle (orange), Right Atrium (brown) and Left Ventricular Cavity (red). (C) Four chambers of the computer model of the human heart that corresponds to image (A). (D) Simulated MR image of the human heart acquired at the same slice location as image (A). The CUDA kernel execution time was approximately 17 minutes for this simulation. (Total number of tissue isochromats 4947708, total number of time-steps 212480, k-space matrix 256 x 256, FOV 40 cm x 40 cm, slice thickness 10 mm). For each image the entire volume of the heart in the VHP phantom was simulated (not just the slice of interest.

4.3 Simulations on motion models

1. Respiratory motion

Fig. 4.5 (left column) shows GE images obtained with the simulator both without motion (A) and with the respiratory motion model (B). Since this GE pulse sequence is not triggered to the respiratory motion (B) a motion artifact appears in the phase encoding direction. Fig. 4.5 also shows CINE GE images triggered to the respiratory motion model for different combinations of phases per cycle and views per segment. Fig. 4.5C illustrates different phases during the respiratory cycle for 250 phases and 2 views per segment (1st row), 125 phases and 4 views per segment (2nd row) and 100 phases and 5 views per segment (3rd row). For a respiratory model displacement of 12 mm, the measured displacement from the CINE GE images was equal to 12.31 ± 0.03 mm.

The implementation of the variable step size "fast algorithm" with the Gradient Echo pulse sequence experiment provided a speedup of four times. The CINE pulse sequence did not benefit at all from the "fast algorithm". In terms of accuracy, the Gradient Echo pulse sequences were compared with and without the variable step size "fast algorithm". For k-space data, the difference between the two was on the order of 0.1% [1]. Finally, for educational purposes, the Gradient Echo pulse sequence experiment was limited to two dimensions (including a 2D motion model) resulting to a total number of 10000 isochromats. This educational 2D configuration demonstrated a kernel execution time of 3.9 sec (speedup of 123 times relative to the 3D configuration, which run in 8 mins) when the variable step size "fast algorithm" was off. When the variable step size "fast algorithm" was on then the 2D configuration run the kernel in 3.7 sec (speedup of 32 times relative to the 3D configuration, which run in 2 mins).

2. Heart motion

Fig. 4.6 (left column) shows the results obtained with MRISIMUL for the application of the GE pulse sequence on the heart motion model. GE images were acquired in the absence of ECG triggering from the heart anatomical model both without motion (A) and with motion (B). The motion artifact caused by the application of non-triggered GE pulse sequence is seen in (B) along the phase encoding direction. Fig. 4.6 (right table) also shows CINE GE images triggered to the motion of the anatomical heart model for a variety of combinations of phases per cycle and views per segment, as mentioned in Methods. Fig. 4.6C illustrates different phases during the cardiac cycle for 100 phases and 1 view per segment (1st row), 50 phases and 2 views per segment (2nd row) and 20 phases and 5 views per segment (3rd row). As expected, a small ghosting artifact is present in phases where the object is moving and multiple k-space lines are acquired per segment (such as fig. 4.6C, phase 12, 3rd row). The inner radius of the heart model (24 mm undeformed and 10 mm deformed) was measured from the CINE GE images to be equal to 24.4±0.52 mm undeformed and equal to 10±0.13 mm deformed [1].

Last, the ability of MRISIMUL to perform myocardial tagging is displayed in Fig. 4.7. This demonstrates the temporal evolution of the contractility of the heart model based on the

tagging that has been applied along the readout direction. Contrast reduction due to T1 relaxation can also be observed throughout the cardiac cycle.

3. Flow motion

Fig. 4.8 shows results obtained from the MRISIMUL simulator by applying a Phase Contrast (PC) velocity encoded gradient echo pulse sequence on the flow model described earlier. Fig. 4.8A depicts the magnitude image obtained from one of the velocity encoded experiments whereas fig 4.8B depicts the phase contrast velocity image. Stationary tissue is depicted in mid-grey whereas flowing spins are depicted in shades of mid-gray to white. The maximum velocity was measured from the phase contrast velocity images and found to be in good agreement with the heart model (model maximum velocity 5cm/s, measured velocity 5.08±0.14 cm/s) [1].

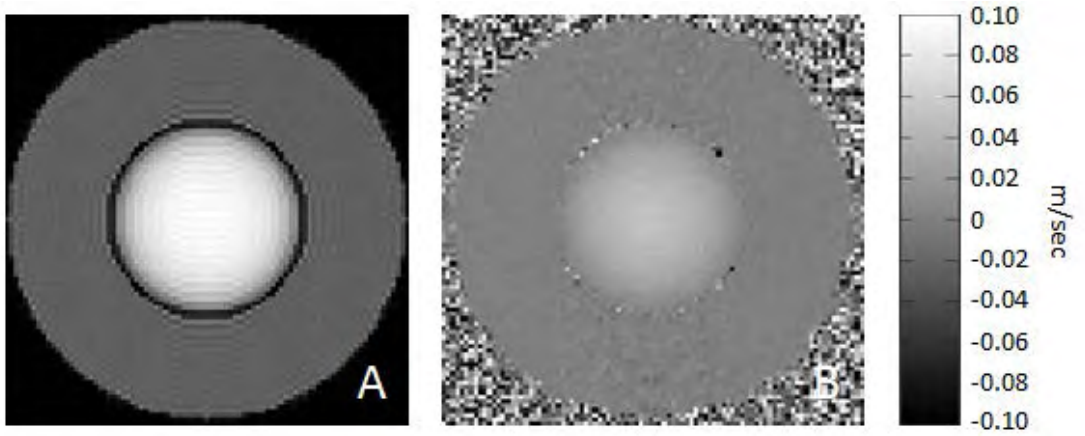

Fig. 4.8. Images obtained by applying a PC velocity encoded GE pulse sequence on the flow model. (A) Magnitude image obtained from the flow model. (B) Corresponding phase image. The entire 3D object was simulated; not just the slice of interest. (Total number of tissue isochromats 30779693, total number of time-steps 1380000, k-space matrix 256x256, slice thickness 10 mm, Kernel execution time on the 4xC2075 system: 111 min for each experiment).

4.4 Spurious echoes formation and image quality

The importance of running simulation experiments with a high isochromat density is illustrated in Fig. 4.9. It can be seen that a lower isochromat density induced a phase difference of more than 180 degrees among neighboring isochromats, which in turn led to spurious echoes formation and image quality degradation.

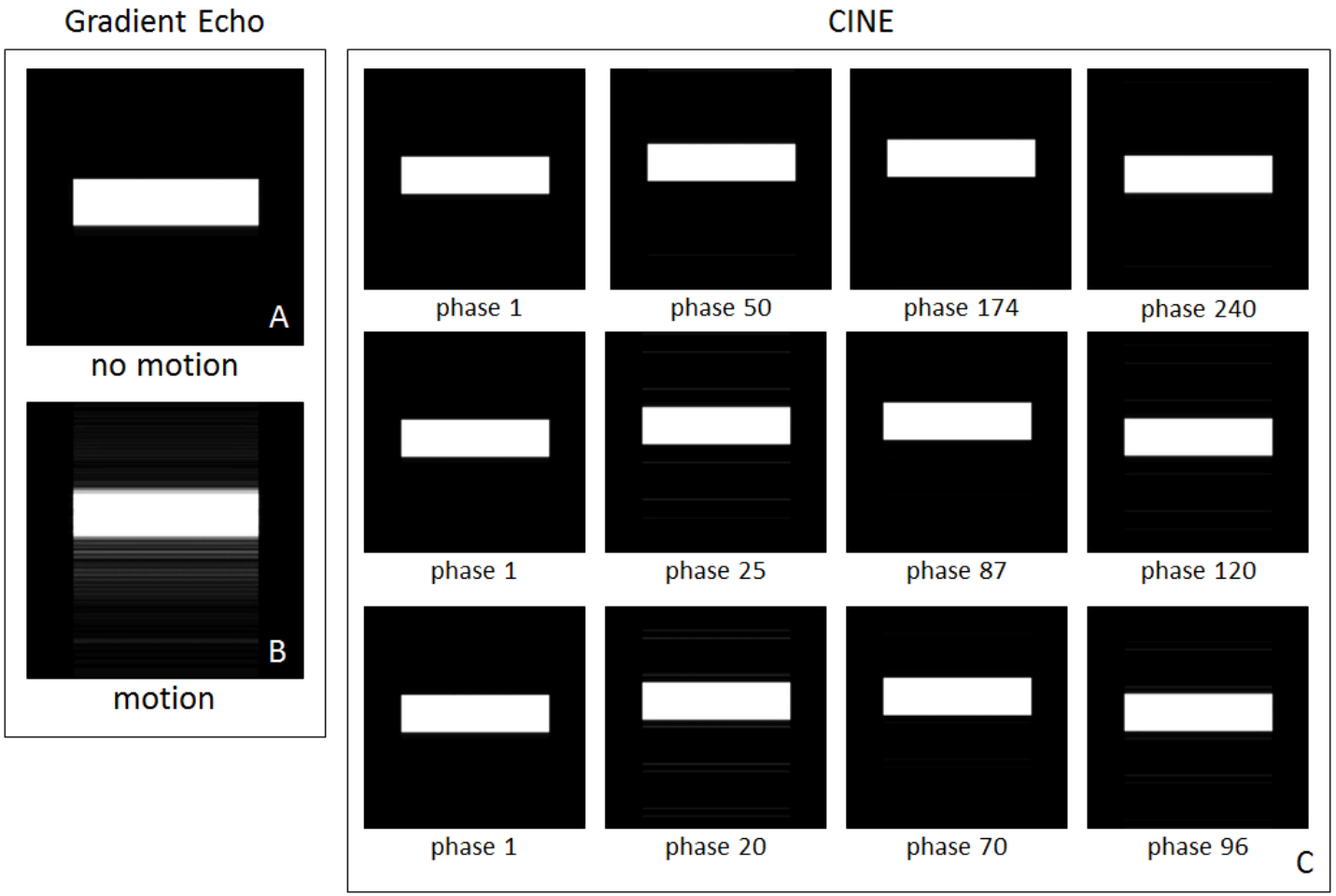

Fig. 4.5. GE images of the respiratory motion model. (A) Application of the gradient echo pulse sequence under no motion. (B) Application of the gradient echo pulse sequence on the respiratory motion model without triggering. (C) Application of the CINE GE pulse sequence with triggering on the respiratory motion model for different combinations of phases per cycle and views per segment: 250 phases and 2 views per segment (1st row), 125 phases and 4 views per segment (2nd row) and 100 phases and 5 views per segment (3rd row). (Total number of tissue isochromats 2000000, total number of time-steps for GE: 1280000, for CINE 1st row: 55200000, for CINE 2nd row: 27600000, for CINE 3rd row: 22400000, k-space matrix 256x256, FOV 15cm x 15cm, slice thickness 10 mm, kernel execution time on the 4xC2075 system for GE: 8 min, for CINE 1st row: 518 min, for CINE 2nd row: 258 min, for CINE 3rd row: 213 min). The entire 3D object was simulated; not just the slice of interest. Contrast has been exaggerated to visualize the residual artifact in the noise floor.

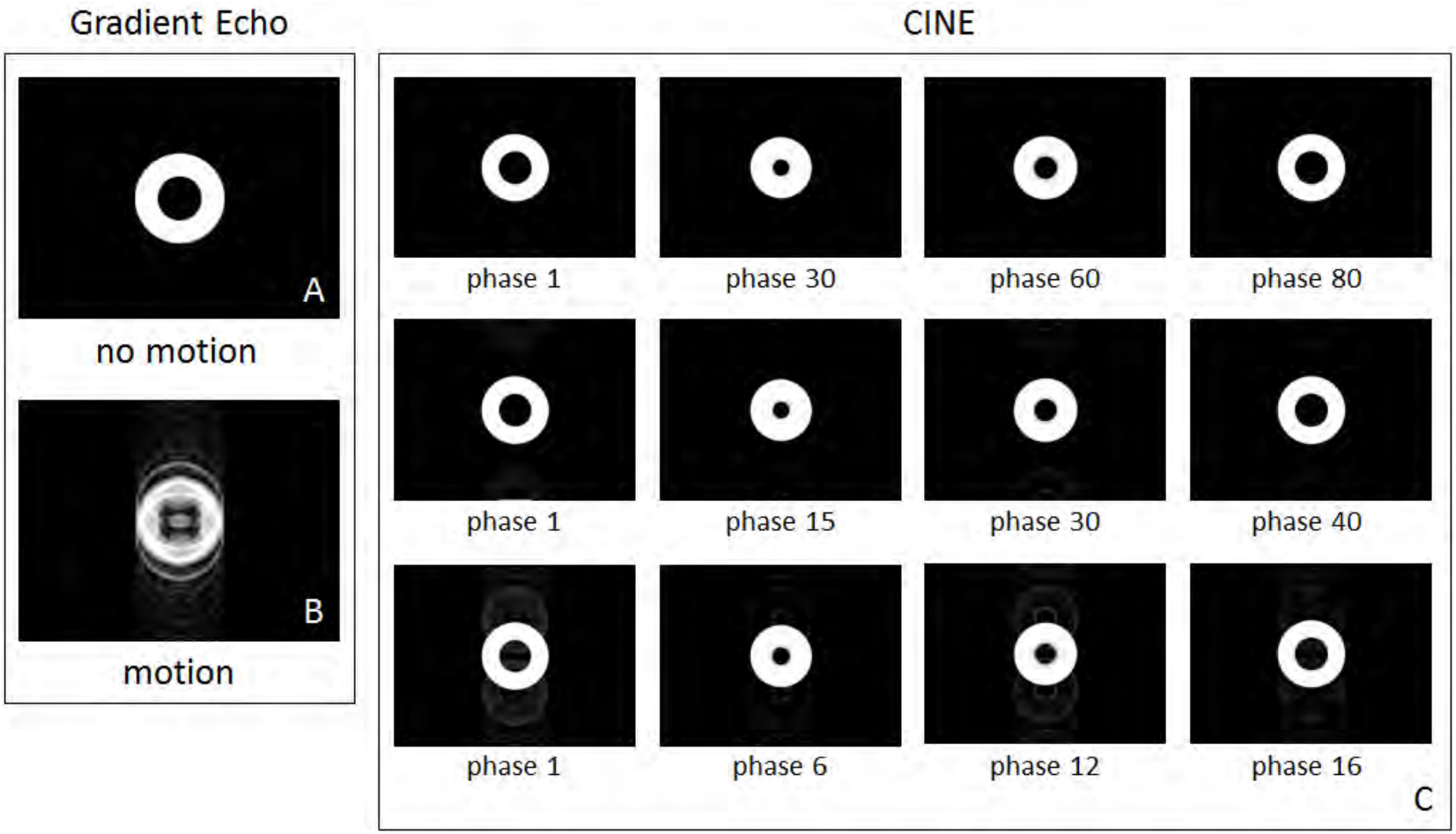

Fig. 4.6. GE images of the heart motion model. (A) Application of the gradient echo pulse sequence on the heart model under no motion. No artifacts observed. (B) Application of the gradient echo pulse sequence on the heart motion model with no triggering. Phase direction artifacts are observed. (C) Application of the CINE GE pulse sequence triggered to the heart motion model for different combinations of phases per cycle and views per segment: 100 phases and 1 view per segment (1st row), 50 phases and 2 views per segment (2nd row) and 20 phases and 5 views per segment (3rd row). (Total number of tissue isochromats 7632792, total number of time-steps for GE: 1280000, for CINE 1st row: 22080000, for CINE 2nd row: 11040000, for CINE 3rd row: 4480000, k-space matrix 256x256, FOV 36cm x 27cm, slice thickness 10 mm, kernel execution time on the 4xC2075 system for GE: 36 min, for CINE 1st row: 887 min, for CINE 2nd row: 434 min, for CINE 3rd row: 175 min). The entire 3D object was simulated; not just the slice of interest. Contrast has been exaggerated to visualize the residual artifact in the noise floor.

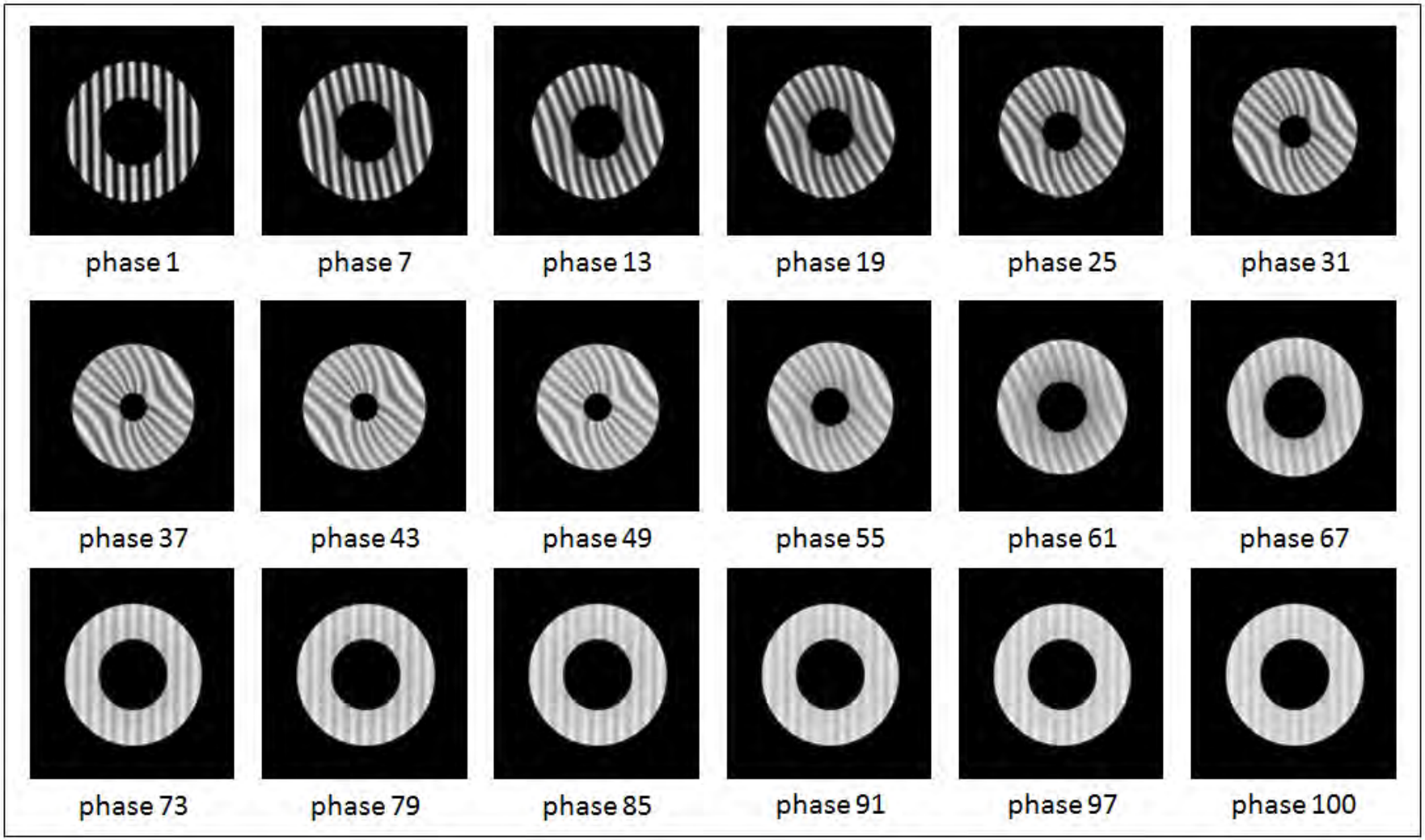

Fig. 4.7. Series of images of the heart model during motion, obtained after a 1-1 SPAMM preparation, producing saturation bands. Note the gradual loss of contrast between the bands and myocardium throughout the cardiac cycle due to T1. Kernel execution time on the 4xC2075 system: approximately 7 hours. The entire 3D object was simulated; not just the slice of interest. (Total number of tissue isochromats 7 632 792, total number of time-steps 11 943 600, k-space matrix 128x128, FOV 15cm x 15cm, slice thickness 10 mm, 100 phases per cardiac cycle and 1 view per segment.)

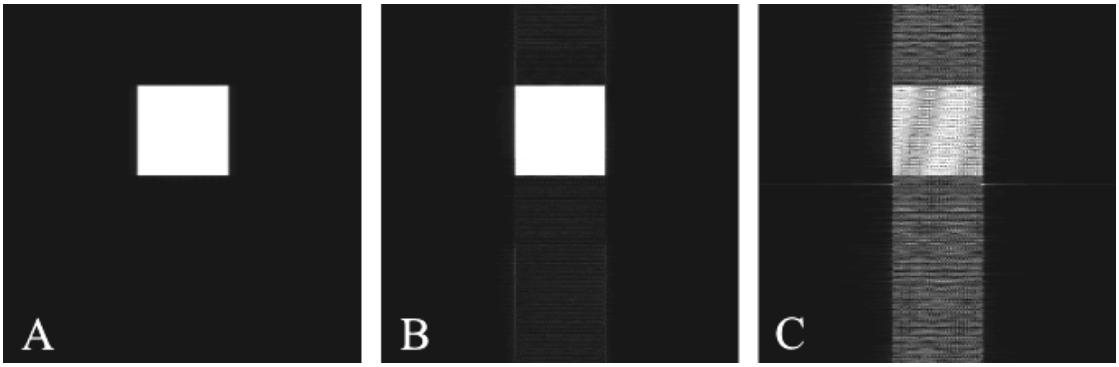

Fig. 4.9. Degradation of image quality due to pulse sequence design and isochromat density within the anatomical object. Gradient Echo images (k-space matrix 256 x 256, FOV 40 cm x 40 cm) were produced from an object with size 10 cm x 10 cm x 30 cm and voxel size 1 mm x 1 mm x 1 mm (Total number of isochromats) (A) 10000000, (B) 3000000, (C) 600000. (CUDA kernel execution approximately 34 mins, 10 mins, 2 mins respectively).

4.5 Software crushers utilization

MRISIMUL allows the introduction of two forms of crushers, either in the form of gradient pulses or as software crushers. Fig. 4.10 shows how software crushers may result in shorter execution times and may better utilize GPU resources. With CINE GE, it can be seen that a lower isochromat density (figure 4.10A) might induce a phase difference of more than 180^{o} among neighboring isochromats. Notice the resulting stripe pattern artifact. This artifact was not present when software crushers were implemented and the spatial density remained unchanged (figure 4.10D).

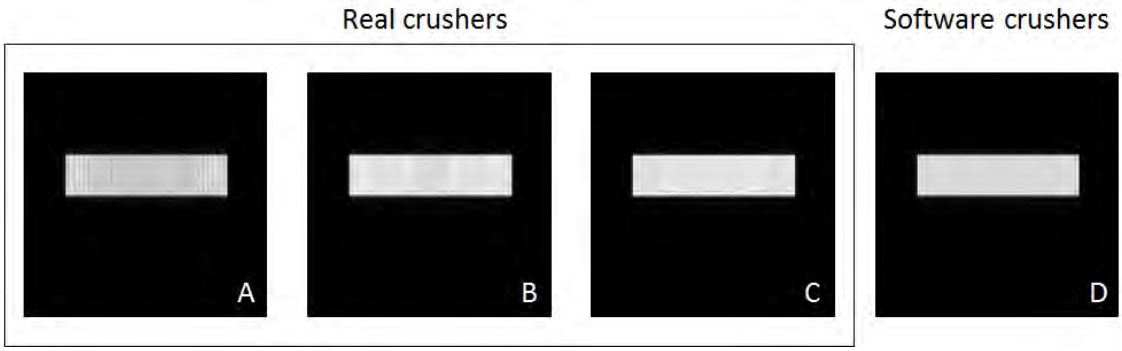

Fig. 4.10. Gradient crusher induced artifacts can be eliminated with software crushers. In images A-C real crushers have been used. (A) shows an image with a fast execution time kernel (6 min) with a stripe pattern artifact related to low isochromat density (volume size: 0.5mm x 0.5mm x 1mm, total number of tissue isochromats: 10000) (B) increasing medium isochromat density reduces the stripe artifact (volume size: 0.25mm x 0.25mm x 1mm, total number of tissue isochromats: 40000) (C) Using a high isochromats density eliminates the artifact at the expense of prolonged (30 min) execution times (isochromat volume size: 0.1mm x 0.1mm x 1mm, total number of tissue isochromats: 250000) (D) With software gradient crushers (all other parameters as in A) and low isochromats density the artifact is not visible. All experiments were run on a 4xC2075 system.

4.6 Multi-GPU performance

The execution times of MRISIMUL's kernel were recorded on both single-node computer systems and on a two node multi-GPU computer system. The performance of the simulator was evaluated with the GE pulse sequence on the respiratory motion model (Total number of tissue isochromats 500000, total number of time-steps: 640000) as mentioned earlier. Fig. 4.11 demonstrates an almost linear scalable performance along with the increasing number of the available GPU cards on both single-node and two-node multi-GPU systems.

4.7 Non-kernel embedded motion model

Non-kernel embedded motion models were also investigated. While requiring longer execution times, they allow for more complex motion patterns that are not necessarily described analytically. The test case was a GE pulse sequence of 640 000 timesteps on the respiratory motion model with 2000000 tissue isochromats. Figure 4.12, blue line, shows the total execution time of the spmd statement of the non-kernel embedded motion model as recorded for different combinations of the available GPU cards on a single-node computer system. As a reference, the execution of the kernel embedded motion model is in red.

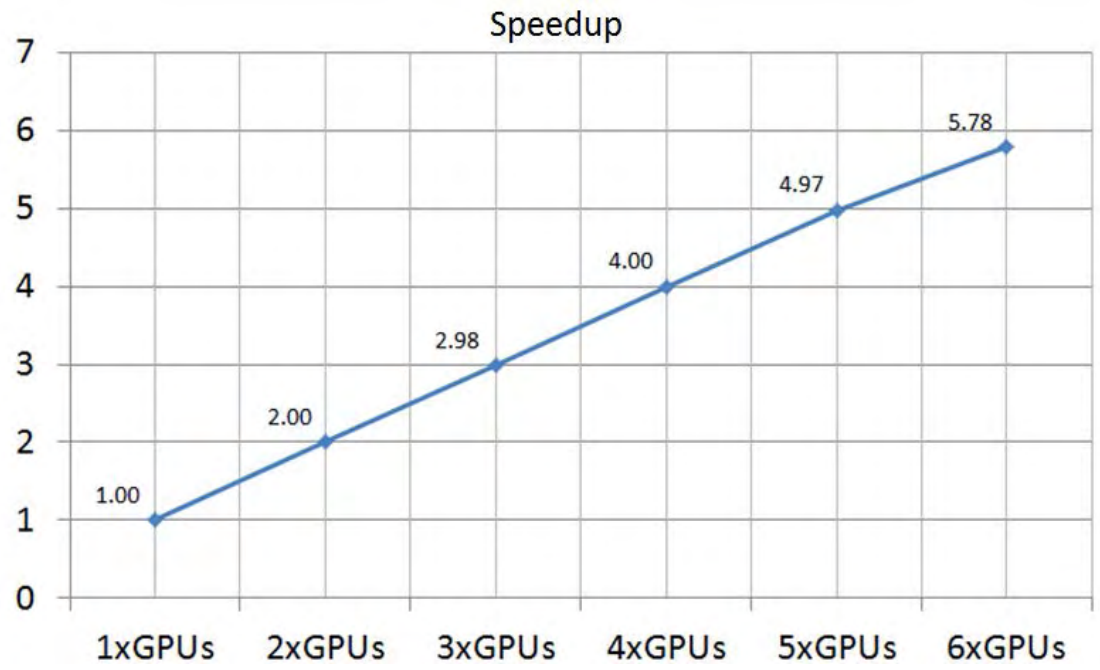

Fig. 4.11. Speedup observed with increasing number of available GPU cards on a multi-node, multi-GPU computer system. Performance is normalized to the single GPU board system, which is the reference standard for all comparisons. An almost linear dependence is observed.

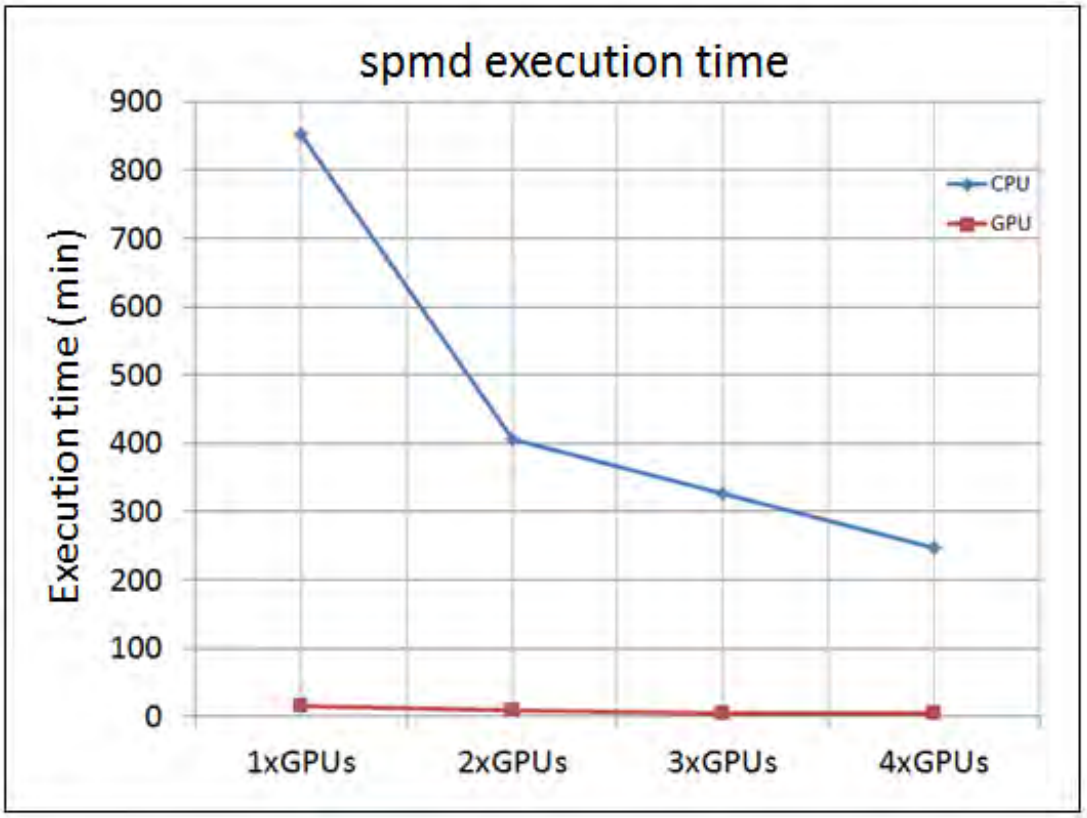

Fig. 4.12. Simulation of non-kernel embedded motion models. The blue line shows execution time of the spmd statement for simulating a non-kernel embedded motion model on the host CPU for different numbers of GPUs on a single-node computer system. The red line shows the execution times for simulating of the same motion model within the GPUs as a kernel embedded motion model. While non-kernel embedded models take longer to execute, they allow for using non-analytically described motion patterns. (Total number of tissue isochromats 2000000, total number of time-steps 640000)

4.8 Other applications

1. Adiabatic RF pulse

The capabilities of the MRISIMUL in terms of supporting frequency modulated RF pulses, e.g. adiabatic pulses, are shown in Fig. 4.13. This is an example of a 10msec inversion Silver-Hoult hyperbolic secant pulse [2] and its effect on a single on-resonance isochromat. Fig. 4.13B shows a 3D vector representation of this isochromat. The rotating frame of reference x'y'z' precesses at the instantaneous frequency of the RF pulse, thus the B1 orientation has been randomly chosen static along the x' axis.

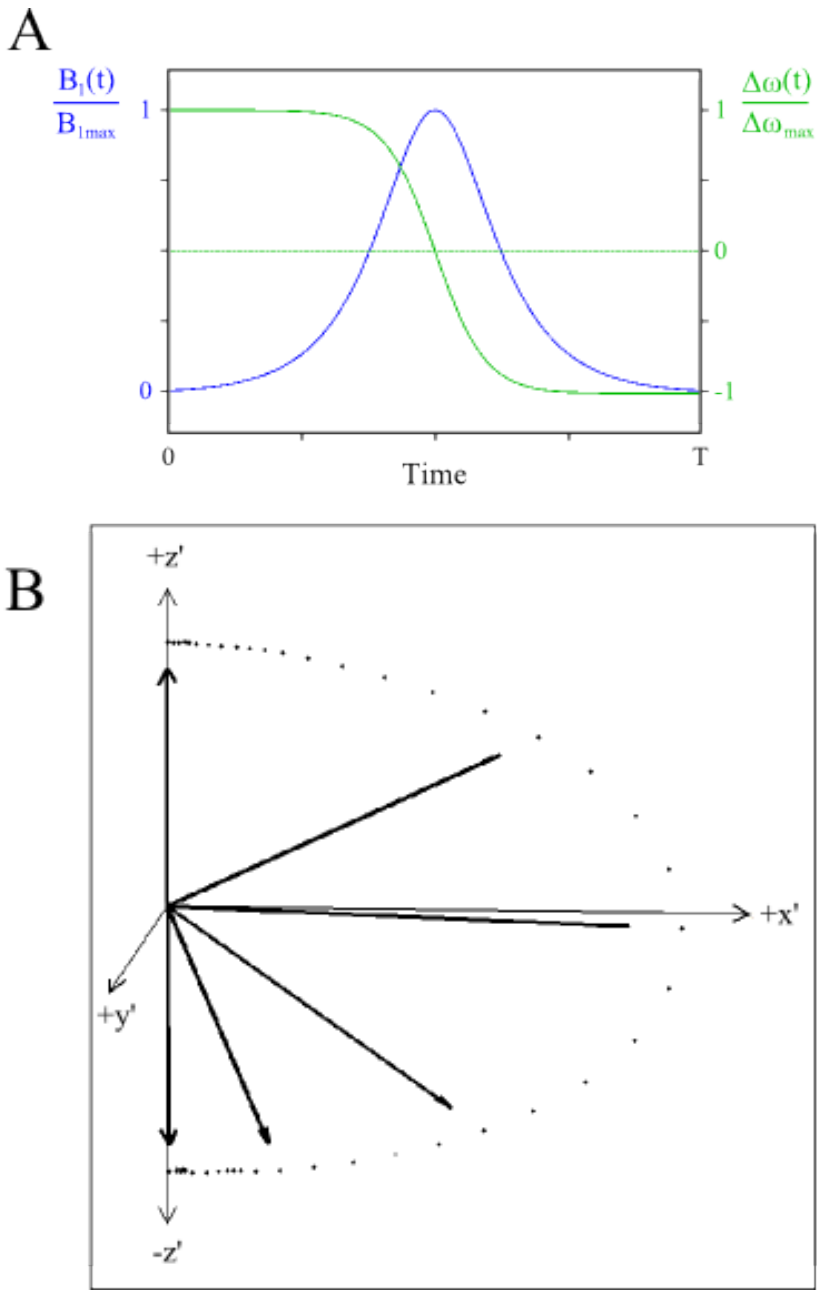

Fig. 4.13. (A) RF amplitude B1(t) and frequency Δω(t) modulation of a Silver-Hoult hyperbolic secant pulse of duration T=10msec. (B) 3D vector representation of the spin trajectory during the application of the Silver-Hoult hyperbolic secant pulse.

2. RF excitation profile

The capability of MRISIMUL to display RF excitation profiles is seen in Fig. 4.14 for two different RF pulse shapes: a rectangular-shaped pulse and a 3-lobe sinc-shaped pulse. It can be seen that, as expected, the frequency profile produced by a sinc-shaped RF pulse has better spectral selectivity and is more uniform compared to the respective frequency profile of the rectangular-shaped RF pulse.

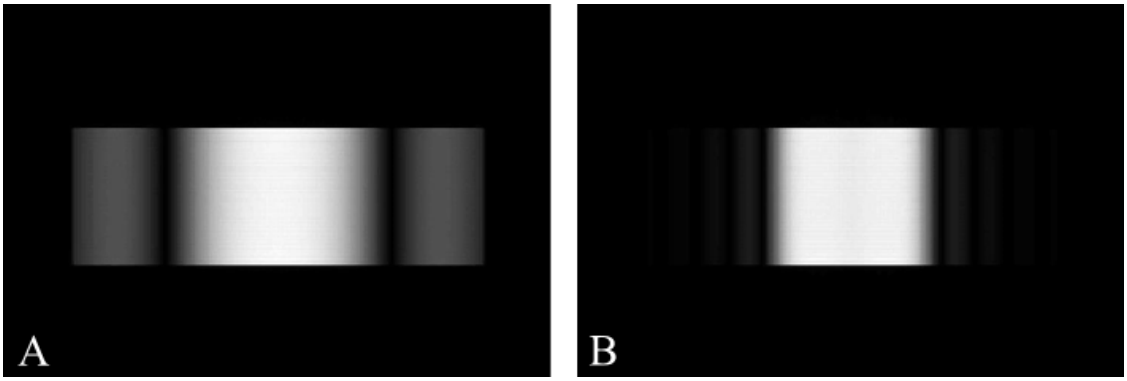

Fig. 4.14. RF excitation profiles of (A) a rectangular-shaped pulse and (B) a three lobe sinc-shaped pulse. The CUDA kernel execution time was approximately 7 minutes for simulation A and approximately 12 minutes for simulation B. (Total number of tissue isochromats 3600000, total number of time-steps 210944, k-space matrix 256 x 256, FOV 40 cm x 40 cm, slice thickness 100 mm).

References

[1] J. M. Bland, and D. G. Altman, "Statistical methods for assessing agreement between two methods of clinical measurement," Lancet, vol. 1, pp. 307 - 310, 1986.

[2] M. S. Silver, R. I. Joseph, and D. I. Hoult, "Highly selective π/2 and π pulse generation," *Journal of magnetic resonance,* vol. 59, pp. 347 - 351, 1984.

5. Discussion

A step-by-step comprehensive Bloch equation simulation platform of MR physics was presented in this study. The MRISIMUL wrapper was developed in MATLAB whereas the main computational kernel was written in CUDA-C and was executed in parallel within the graphic processing unit (GPU) environment. The high performance of MRISIMUL allowed its application in large-scale analysis without model simplifications and can bring the computational power of a supercomputer or a large computer cluster to a single GPU personal computer.

More advanced MR experiments that involved general-purpose, realistic, motion-related simulations were also investigated in multi-GPU systems. This is the first MR physics simulator to have implemented motion with a 3D large computational load on a single computer multi-GPU configuration. The use of both single-node and multi-node multi-GPU systems allowed for an almost linear speedup with increasing number of GPU boards. Non-kernel embedded motion models, even though they take longer to execute, were successfully demonstrated as a solution to using motion models that cannot be described analytically. The use of software crushers was presented as an alternative to real crushers for allowing for further simulation speedup. The variable time step “fast algorithm” was also examined in this study so as to measure the potential speedup gain.

MRISIMUL was applied to stationary anatomical computer models and motion models. The stationary computer models included 3D models of the anatomy of the human brain and heart. The brain anatomical model used for simulations was a brain phantom available online by the McConnell Brain Imaging Center at McGill University whereas the heart anatomical model was developed based on the dataset of high resolution images provided by the Visible Human Project (VHP) of the United States National Library of Medicine (Bethesda, MD). Motion models included a respiratory motion model, a heart motion model and a simple flow model. The respiratory motion model simulated one-directional motion of the diaphragm whereas the heart motion model simulated myocardial deformation due to longitudinal and radial contraction, axial torsion and rigid body displacement. Last, laminar flow of a homogeneous fluid within a straight tube was also simulated.

Gradient Echo pulse sequences were applied to the stationary anatomical computer models whereas Gradient Echo and CINE Gradient Echo were applied to motion models so as to demonstrate motion-related artifacts and to explore motion-related MR applications.

The design of MRISIMUL supported the implementation of frequency modulated RF pulses and took into account multi-component voxels. It also allowed the introduction of a time varying inhomogeneity field map and RF pulse sensitivity maps. MRISIMUL utilized controls to prevent spurious echo formation due to acquired phase difference of more than 180 degrees among neighboring isochromats within the anatomical object. These controls set the lowest spatial resolution that the anatomical object must have so that image quality degradation could be avoided. Nevertheless, special attention should be paid by the user with respect to the choice of isochromat density and temporal resolution in those cases where the experiment involves the application of an additional inhomogeneity map and

non-uniform distribution of the magnetization within the volume of the object. However, the introduction of strong gradients (crushers) into the pulse sequence may result in artificial spurious echoes and degradation of the final image quality, especially in cases that involve temporal deformation of the anatomical model. For non-deforming moving tissue or laminar flow the relative distances between neighboring isochromats did not change and therefore the aforementioned scheme performed well. For deforming tissue the local strain was taken into account and the minimum acceptable distance between isochromats was further decreased according to the strain; then the aforementioned algorithm was applied. For more complex types of motion, such as turbulent flow, this solution no longer works. In this case, the temporal evolution of the motion model in 3D would have to be examined so as to limit the maximum phase to less than 180 degrees between neighboring moving isochromats. Also, certain restrictions would have to be imposed on the motion model so that isochromats cannot occupy the same spatial location. The current implementation of MRISIMUL does not address this issue.

In pulse sequence design, MRISIMUL introduced a second form of crushers, namely software crushers. These crushers induced nullification of the transverse components of every isochromat within the anatomical object. The replacement of real gradient crushers by software crushers allowed for not being forced to increase isochromat spatial density. This resulted in faster execution times without spurious echo formation. In simulations that involve real crushers and anatomical model deformations (such as stimulated echoes applied to the myocardium), the required spatial density of the model may not always be obvious due to tissue strain. Special considerations have to be made in such cases as well.

In this work, external Bo and B1 maps with time dependence were not studied. Also, spurious echoes, as described above, may arise when ΔBo is time dependent and unknown ahead of time thus resulting in suboptimal modeling of T2* when a low number of isochromats per voxel is used. Also, with MRISIMUL all fields and positions are assumed constant within each time step of the simulator thus setting a limit on the permissible length of the time step when considering fast motion. The maximum time step length is dependent on the spectral content of the motion. At the expense of execution time, the aforementioned problems can be mitigated by shorter time steps and increased number of isochromats.

Moreover, the simulator's performance was evaluated on different computer configurations and datasets. MRISIMUL demonstrated a linear relationship between kernel execution time and increasing number of isochromats and a relationship close to linear between kernel execution time and total number of time-steps. Additionally, MRISIMUL implemented an algorithm (fast algorithm) that introduced further speedup of the experiment by temporarily prolonging the pulse sequence time-step in cases where no temporal changes occur within the pulse sequence and/or the anatomical model (no motion) and the acquisition window was off. However, in order to accommodate the high computational load introduced by simulation of motion experiments, the kernel distributed in a balanced manner the computational load to multiple GPUs. MRISIMUL demonstrated an almost linear scalable performance with the increasing number of available GPU cards, in both single-node and multi-node multi-GPU computer systems.

Last, it should be noted that the simulation took place in the entire 3D object and not just in the imaging slice of interest. This allows the observation of the slice profile effects on the resulting image and in the future it may help in simulating flow artifacts originating from out of slice.

In the past, three other advanced MR physics simulators have been presented in the literature; namely SIMRI, POSSUM and JEMRIS [1-3]. Compared to MRISIMUL, these environments and installation procedures require advanced technical knowledge. Also, their full application in large-scale analysis requires an advanced setup of a computer cluster of multiple nodes, knowledge of implementation and the ability to make several modifications in the source code. A direct comparison among different simulation platforms is not easy due to design and implementation differences but also due to different experiment sizes, such as the number of isochromats and the total time-steps of the pulse sequence. Moreover, from these three MR physics simulators, only POSSUM and JEMRIS have presented in the literature incorporation of motion in their simulations. However, both of them presented the following limitations/assumptions: 1. they allowed only simulation of simple translational and/or rotational motion models, 2. simulations were usually limited within the slice of interest and did not include the entire 3D anatomical model. Although the latter assumption is not related to any specific design limitation of these simulation platforms, the main reason for imposing it was the high computational load introduced by the simulation of an entire motion model during the entire course of the pulse sequence. Compared to these simulators, MRISIMUL exploited the high computational power of GPU technology and allowed for simulating realistic motion models in the entire 3D anatomical object during the entire course of the MR pulse sequence. Moreover, the high computational power of MRISIMUL further enhanced the utilization of non-kernel embedded motion models that could not be simulated within the GPU kernel in an analytical manner.

In conclusion, MRISIMUL is a high-performance simulation platform of MRI physics that allows its application in large-scale analysis by utilizing GPU-technology. Moreover, MRISIMUL is an MR physics simulator that allows for computationally intense simulations of general-purpose, realistic, 3D motion-related MR experiments on single-node computer systems by taking advantage of the high computational power of multi-GPU configurations. The incorporation of realistic motion models, such as heart motion and flow models may benefit the design and optimization of existing or new MR pulse sequences, protocols and algorithms that examine motion related MR applications. In the future, MRISIMUL may benefit not only research aimed at optimizing imaging protocols but also in terms of low-cost training of technologists, physicians and students. Moreover, the future reduction in cost of GPU based technology suggests that such an MRI simulator has the potential to become an indispensable tool for medical imaging centers and may change the optimization methods for MRI protocols.

References

[1] H. Benoit-Cattin, G. Collewet, B. Belaroussi, H. Saint-Jalmes, and C. Odet, "The SIMRI project: a versatile and interactive MRI simulator," *J Magn Reson,* vol. 173, pp. 97-115, Mar 2005.

[2] I. Drobnjak, D. Gavaghan, E. Suli, J. Pitt-Francis, and M. Jenkinson, "Development of a functional magnetic resonance imaging simulator for modeling realistic rigid-body motion artifacts," *Magn Reson Med,* vol. 56, pp. 364-80, Aug 2006.

[3] T. Stocker, K. Vahedipour, D. Pflugfelder, and N. J. Shah, "High-performance computing MRI simulations," *Magn Reson Med,* vol. 64, pp. 186-93, Jul 2010.

6. Future Work

MRISIMUL is a step-by-step comprehensive Bloch equation simulation platform of MR physics that allows its application in large-scale analysis without model simplifications and can bring the computational power of a supercomputer or a large computer cluster to a single GPU personal computer. More advanced MR experiments that involve general-purpose, realistic, motion-related simulations are now feasible on multi-GPU computer systems.

Although single node, multi-GPU computer systems demonstrated high performance in large-scale MR experiments, the simulation of realistic complex experiments that involve motion of the anatomical computer models, such as myocardial motion and strain or blood flow, seem to require even better hardware solutions to accommodate the increased processing power needed. Moreover, the multi-purpose use of MRISIMUL - from optimization of complex imaging protocols down to training of technologists, physicians and students on simple MR physics concepts - points out the need of an on-demand, scalable computer system that will be adjusted on the processing power required each time.

Currently, MRISIMUL is being used as an in-house simulation platform whereas further development of more advanced motion models is under way and validation of them against well-established clinical protocols is warranted in the future. A web-based application that gives web access to our CUDA-capable T7500 computer installation has already been developed. In this application, a user can make simple protocol changes to preconfigured MRI pulse sequences and we envision that, in the future, this interface will be expanded to allow the user to enter his/her own more complex pulse sequence experiments in a variety of anatomical models.

Appendix 1

Submitted & Published Articles

The work presented in this dissertation has been published in the following peer-review papers:

[1] C. G. Xanthis, I. E. Venetis, A. V. Chalkias, and A. H. Aletras, "MRISIMUL: A GPU-based Parallel Approach to MRI Simulations", IEEE Transactions on Medical Imaging, vol. 33, Issue 3, March 2014

[2] C. G. Xanthis, I. E. Venetis and A. H. Aletras, "High performance MRI simulations of motion on multi-GPU systems", Journal of Cardiovascular Magnetic Resonance, vol. 16, Issue 48, 2014

Other scientific contribution that is not included in this dissertation:

[3] Kanski et al., “Whole-heart 4D flow magnetic resonance without respiratory gating reduces scan time without affecting quantitative intracardiac flow parameters” (in preparation)

[4] Hedström et al.,”A fully automatic algorithm for assessing T2* and its certainty value for accurate cardiac and liver iron load determination” (in preparation)

Appendix 2

Abstracts Presented in Conferences

[1] C. G. Xanthis, I. E. Venetis, and A. H. Aletras, "A high performance parallelizable MRI physics simulator with graphic processing unit technology," Journal of Cardiovascular Magnetic Resonance, vol.15 (Suppl 1), 2013

[2] C. G. Xanthis, I. E. Venetis, and A. H. Aletras, "Accelerating MRI physics simulations using GPU computing: comparison with other computer configurations," 5th Panhellenic Conference on Biomedical Technology, Athens, Greece, 4-6 April 2013

[3] C. G. Xanthis, I. E. Venetis, and A. H. Aletras, "Accelerated MR Physics Simulations on multi-GPU systems", IEEE 13th International Conference on Bioinformatics and Bioengineering (BIBE 2013), Chania, Greece, 10-13 November 2013

[4] C. G. Xanthis, I. E. Venetis, and A. H. Aletras, "Accelerated MR simulations of heart motion model using graphic processing unit technology", 16th Cardiovascular Spring meeting, Malmo, Sweden, 7-9 May 2014

Other scientific contribution that is not included in this dissertation:

[5] N. K. Ververis, C. G. Xanthis, S. L. Bidhult, A. H. Aletras and E.K Heiberg, "Segmentation and visualization of cardiovascular tree using high performance graphics processing cards", 16th Cardiovascular Spring meeting, Malmo, Sweden, 7-9 May 2014

Xanthis et al. Journal of Cardiovascular Magnetic Resonance 2013, 15(Suppl 1):E45
http://www.jcmr-online.com/content/15/S1/E45

Journal of Cardiovascular Magnetic Resonance

POSTER PRESENTATION Open Access

A high performance parallelizable MRI physics simulator with graphic processing unit technology

Christos G Xanthis[1*], Ioannis E Venetis[2], Anthony H Aletras[1]

From 16th Annual SCMR Scientific Sessions
San Francisco, CA, USA. 31 January - 3 February 2013

Background

The acquisition of high quality Magnetic Resonance (MR) images requires exploring a parameter space for image quality improvement. This iterative process can benefit from simulations of MRI pulse sequences and imaging protocols.

Current MRI physics simulators are confined to few pulse sequences and compromises are made due to the high computational power needed. A step-by-step comprehensive Bloch equation simulation from signal acquisition to image formation, without the aforementioned compromises, may help in evaluating cardiovascular MRI protocols and pulse sequences.

Methods

In this study, a comprehensive computer simulation platform of MR physics was designed and developed based on work by Nazarova et al (2004). The simulation platform makes no assumptions of the underlying pulse sequence and was developed in MATLAB. The computationally demanding core was executed in parallel within the graphic processing unit (GPU) environment by employing CUDA technology.

The simulation platform allowed the development of custom MRI pulse sequences and their application on computer models, including a detailed 3D model of the anatomy of the human heart and torso. The simulator accounted for the main static field and also allowed for the introduction of a time varying inhomogeneity field map. The time evolution of the magnetization vector and its components were displayed in 3D.

The efficacy of GPU implementation was evaluated against other computer configurations. Execution times were recorded for the application of the simulator on three different computer configurations a) one single CPU computer (Intel Pentium D, 3.40GHz) b) a 23 core computer cluster and c) one single CPU computer with one GPU Tesla c2070 of 448 GPU cores. These simulations were based on a Gradient Echo pulse sequence applied on a cubic object of 21600 voxels (k-space 256x256).

Finally, the GPU-based simulation was also applied on a detailed 3D model of the anatomy of the human heart. However, due to RAM memory size limitations on the PC side (4 GB), it was applied only to the right atrium tissue and its blood cavity.

Results

Speedup of almost three orders of magnitude (x908) was recorded when compared to CPU-based and more than two orders of magnitude (x240) when compared to cluster-based systems.

A simulated short-axis tomographic image of the right atrium and its blood cavity is shown in Figure 1.

Conclusions

The MRI simulator proposed in this study allows its application in large-scale analysis without model simplifications by employing the GPU technology. The almost three orders of magnitude speedup obtained in MRI simulations with a single video card of 448 GPU cores is promising. The future reduction in cost of GPU based technology suggests that such an MRI simulator has the potential to become an indispensable tool for medical imaging centers and should change the manner by which MRI protocols are optimized.

[1]Department of Computer Science and Biomedical Informatics, University of Central Greece, Lamia, Greece
Full list of author information is available at the end of the article

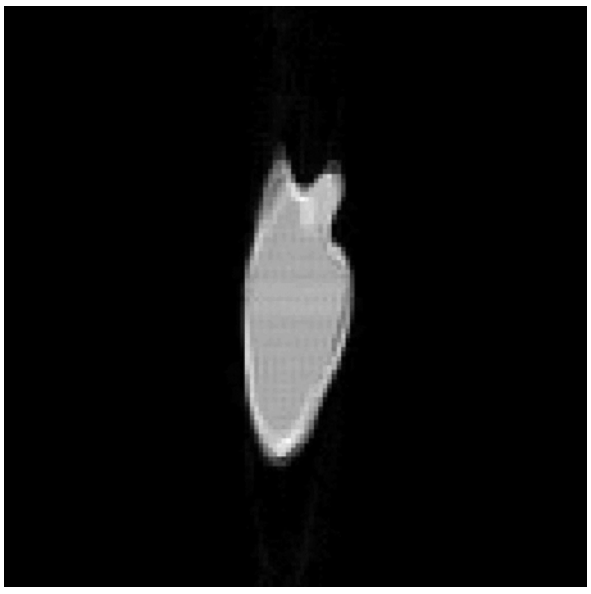

Figure 1 Simulated short-axis tomographic image of the right atrium and its blood cavity.

Funding

Funding provided by the European Research Council via a four year Marie Curie Actions grant (FP7 - PEOPLE - 2009 - IRG call) and by the 'Alexander S. Onassis public benefit foundation'.

Author details
[1]Department of Computer Science and Biomedical Informatics, University of Central Greece, Lamia, Greece. [2]Department of Computer Engineering and Informatics, University of Patras, Patra, Greece.

Published: 30 January 2013

doi:10.1186/1532-429X-15-S1-E45
Cite this article as: Xanthis *et al.*: A high performance parallelizable MRI physics simulator with graphic processing unit technology. *Journal of Cardiovascular Magnetic Resonance* 2013 **15**(Suppl 1):E45.

Accelerating MRI physics simulations using GPU computing: comparison with other computer configurations

C. G. Xanthis[1,2], I. E. Venetis[3] and A. H. Aletras[1,2]

[1]Department of Computer Science and Biomedical Informatics, University of Central Greece
[2]Department if Clinical Physiology, Skåne University Hospital Lund, Lund University, Sweden
[3]Department of Computer Engineering and Informatics, University of Patras, Greece

Introduction

MRI physics simulators may be used both for optimizing imaging protocols and for training purposes. Since the introduction of the first nuclear magnetic resonance physics simulator by Summers, Axel and Israel in 1986, further research has been conducted on the development of various simulators serving different purposes. However, current MRI physics simulators are confined to few pulse sequences and compromises are made due to the high computational power needed.

A step-by-step comprehensive Bloch equation simulator, which allows for high performance parallelizable computations, without the aforementioned compromises, was designed and developed employing the CUDA-technology [1]. In this study, the efficacy of GPU implementation was evaluated against other computer configurations.

Methods

A comprehensive computer simulation platform of MR physics was developed that integrates all the stages of MRI physics from signal generation to image reconstruction in a realistic manner [1]. This simulation platform allowed the development of custom MRI pulse sequences and their application on computer models, including custom phantoms, a brain model [2] and a detailed 3D model of the anatomy of the human heart and torso, previously developed in our lab.

The simulation platform was developed in MATLAB while the main computationally demanding services were written in CUDA-C and were executed in parallel within the GPU environment.

Execution times were recorded for the application of the simulator on three different computer configurations: 1) one single CPU computer (Intel Pentium D, 3.40GHz) 2) a 23 core computer cluster using MATLAB spmd constructs 3) one single CPU computer with one GPU Tesla c2070 of 448 GPU cores. These simulations were based on a Gradient Echo pulse sequence applied on a cubic object of 21600 voxels (k-space 256x256).

Results

Conventional CPU-based calculations offered limited computing performance since the simulator involves multiplication/summation of large matrices whose values change in time and cannot be addressed by using sparse matrices. In addition to that, sometimes the size of the object being simulated prohibited the initialization of the simulator due to memory limits.

The setup of a computer cluster of 23 cores produced an increased computing efficiency, though the runtimes were still long for realistic MRI simulations.

On the other hand, the GPU-based simulation recorded a speedup of almost three orders of magnitude (x908) when compared to CPU-based and more than two orders of magnitude (x240) when compared to the cluster-based system (Table 1).

Table 1: Execution times for the application of a pulse sequence of 480000 time-steps on a cubic object of 21600 isochromats. Speedups are compared to the CPU-based setup.

Computer Setup	Execution Time (sec)	Speedup
CPU-based	91950	1
Cluster (23-cores)	15843	≈ 4
GPU c2070	66	908

Discussion

The CUDA technology employed by this MRI simulator allows its application in large-scale analysis without model simplifications. The almost three orders of magnitude speedup obtained in MRI simulations with a single video card of 448 GPU cores is promising and reveals the potentials of such an MRI simulator in optimizing MRI protocols and for educational purposes.

References

1. C.G.Xanthis, I. E. Venetis and A. H. Aletras (2013) A high performance parallelizable MRI physics simulator with graphic processing unit technology, Journal of Cardiovascular Magnetic Resonance 15(Suppl 1):E45
2. R.K.S. Kwan, A.C. Evans, G.B. Pike (1999) MRI simulation-based evaluation of image-processing and classification methods, IEEE Transaction on Medical Imaging, 18:1085–1097

Keywords: GPU, CUDA, MRI, simulations, physics

Funding: Funding provided by the European Research Council via a four year Marie Curie Actions grant (FP7 – PEOPLE – 2009 – IRG call) and by the ‘Alexander S. Onassis public benefit foundation’. Two NVIDIA Tesla C2070 GPU computing cards have been donated from NVIDIA through the “Professor Partnership” program.

Accelerated MR Physics Simulations on multi-GPU systems

Christos G. Xanthis, *Member, IEEE*, Ioannis E. Venetis, and Anthony H. Aletras

***Abstract* — A multi-GPU approach of MRISIMUL, a recently developed step-by-step comprehensive MR physics simulator of the Bloch equation, is presented in this study. The specific aim was to apply MRISIMUL on multi-GPU systems so as to achieve even shorter execution times. We hypothesized that such a simulation platform could achieve a scalable performance with the increasing number of available GPU cards on single node, multi-GPU computer systems. A parallelization strategy was employed using the MATLAB single-program-multiple-data (spmd) statement and an almost linear speedup was observed with the increasing number of available GPU cards on two separate systems: a single computer of 2 quad-core processors and two Tesla C2070 GPU cards and a single computer of 2 hexa-core processors and four Tesla C2075 GPU cards.**

I. Introduction

The acquisition and the interpretation of Magnetic Resonance (MR) images usually requires a higher level of understanding of the underlying physics. Towards that direction, MRI physics simulators have been used in the past for training purposes but also for tracking artifact sources and answering methodological problems during pulse sequence or imaging protocol development. However, current MRI simulators are usually confined to a handful of pulse sequences whereas important compromises are made due to the high computational power needed. Two previously developed MRI simulation platforms [1, 2] have been shown to address the long execution times of more complicated large-scale experiments by utilizing parallel processing on computer clusters based on the Message Passing Interface (MPI) communications protocol [2, 3]. Nevertheless, both systems require an advanced cluster setup of multiple nodes along with advanced technical knowledge.

Recently, a new MR physics simulation platform, by the name MRISIMUL, was developed and presented [4]. MRISIMUL is a step-by-step comprehensive Bloch equation simulator from signal acquisition to image formation which allows its application in large-scale analysis without model simplifications by employing GPU technology. Previous results [4] have already demonstrated high computational speedup of GPU-based simulations on a single board GPU personal computer compared to serially executed code on the CPU and OpenMP parallel executed code on the CPU with multi-threading (up to 8 threads).

In this study, the implementation of MRISIMUL on single node, multi-GPU computer systems is presented. The specific aim was to apply MRISIMUL on multi-GPU systems so as to achieve even shorter execution times. We hypothesized that such a simulation platform could be adapted and exploited on single node, multi-GPU computer systems achieving a scalable speedup compared to that of a single GPU system.

II. Methods

MRISIMUL is a step-by-step comprehensive Bloch equation simulator [4]. This simulation platform allowed the development of custom MRI pulse sequences and their application on 3D computer models of realistic objects. MRISIMUL was developed in MATLAB (The Mathworks Inc., Natick, MA) while the computationally demanding core services (kernel) were developed in CUDA-C (NVIDIA, Santa Clara, CA) and executed in parallel within the graphic processing unit (GPU) environment. MATLAB handled the basic MR imaging processes, such as pulse sequence programming, anatomical object development and reconstruction whereas CUDA-C handled the multiplication and summation of large matrices for the calculation of the object's magnetization vector.

For multi-GPU parallelism, the MATLAB single-program-multiple-data (spmd) statement was employed. The number of the on-system available GPU cards defined the size of the pool of processes for parallel computation. Each process was assigned a certain part of the object being analyzed and a GPU card that would be used for the computationally demanding CUDA-C kernel. The assignments were designed to preserve a balanced load for all GPUs. At the end, the raw datasets of each part of the object were summed so as to form the k-space of the object (figure 1).

To demonstrate the efficacy of the multi-GPU implementation of MRISIMUL, its performance was evaluated on two different single computers with multiple GPUs. The first system was a single computer of 2 quad-core processors and 48GB RAM with two Tesla C2070 GPU cards of 448 GPU cores and 6GB global memory each.

Manuscript received July 1, 2013. This project is funded by the European Research Council (PIRG06-GA-2009-256569 FP7 MARIE CURIE IRG) and by the "Alexander S. Onassis public benefit foundation". Two NVIDIA Tesla C2070 GPU computing cards were donated from NVIDIA through the "Professor Partnership" program. The University of Thessaly funded hardware for this project. The United States National Library of Medicine (Bethesda, MD) for the Visible Human Project (VHP) dataset.

C. G. Xanthis and A. H. Aletras are with the Department of Computer Science and Biomedical Informatics, University of Thessaly, Greece and with the Department of Clinical Physiology, Skåne University Hospital Lund, Lund University, Sweden (e-mail: cxanthis@gmail.com; aletras@hotmail.com).

I. E. Venetis is with the Department of Computer Engineering and Informatics, University of Patras, Greece (e-mail: venetis@capsl.udel.edu).

The second system was a single computer of 2 hexa-core processors and 32 GB RAM with four Tesla C2075 GPU cards of 448 GPU cores and 6GB global memory each.

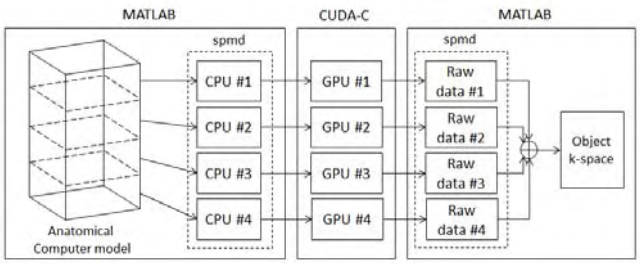

Fig. 1. Each CPU core is assigned a certain part of the anatomical computer model and a certain GPU card. A balanced load among the GPUs has been considered. At the end of the kernel execution, the resulting signals were summed and the final k-space of the object was formed.

The execution times were recorded for all the possible combinations of the available GPUs in both computers for the application of a Gradient Echo (GE) pulse sequence on a 3D computer model of the anatomy of the human heart [4]. The total number of timesteps was 98816 while the anatomical computer model consisted of a total of 4947708 tissue isochromats. The GE pulse sequence was applied with TE = 4.5msec and TR = 50msec, the RF pulse was a three lobe sinc-shaped pulse with a flip angle of 15^{o} whereas a gradient crusher pulse was applied along the slice selection direction at the end of each TR. The temporal resolution of the pulse sequence was 10μsec, the bandwidth of the receiver was 50 kHz, the k-space matrix was of size 128x128 and the main magnetic field was set to 1.5 Tesla. The heart computer model had a voxel size of 0.33mm x 0.33mm x 0.99mm and it was based on the segmentation of a dataset of high resolution images provided by the Visible Human Project (VHP) of the United States National Library of Medicine (Bethesda, MD) [5].

TABLE I
KERNEL EXECUTION TIMES

Single Computer with 2 Tesla C2070 GPUs

No. of GPUs	Execution time (hours)	Speedup
1	3.10	-
2	1.56	1.99

Single Computer with 4 Tesla C2075 GPUs

No. of GPUs	Execution time (hours)	Speedup
1	3.15	-
2	1.58	1.99
3	1.09	2.89
4	0.82	3.84

Execution times of the kernel were recorded for the application of a pulse sequence of 98816 timesteps on a computer model of the human heart of 4947708 tissue isochromats.

III. RESULTS

The execution times of MRISIMUL's kernel on different single node, multi-GPU computer configurations were recorded and are shown in Table I. It can be seen that a speedup of about 2 times when compared to a single Tesla C2070 GPU was observed on the first computer system with the parallel use of two Tesla C2070 GPUs whereas a speedup of almost 4 times when compared to a single Tesla C2075 GPU was observed on the second computer system with the parallel use of four Tesla C2075 GPUs. To illustrate this better, figure 2 depicts an almost linear scalable performance along with the increasing number of available GPU cards on both single node, multi-GPU computers. The resulting simulated MR image of the above-mentioned simulations, which was acquired from the same slice location of the Visible Human Project (figure 3A), is illustrated in figure 3B.

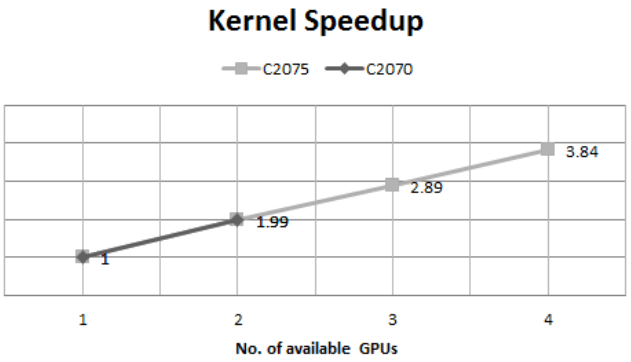

Fig. 2. Speedup observed with increasing number of available GPU cards on the single node, multi-GPU computer systems.

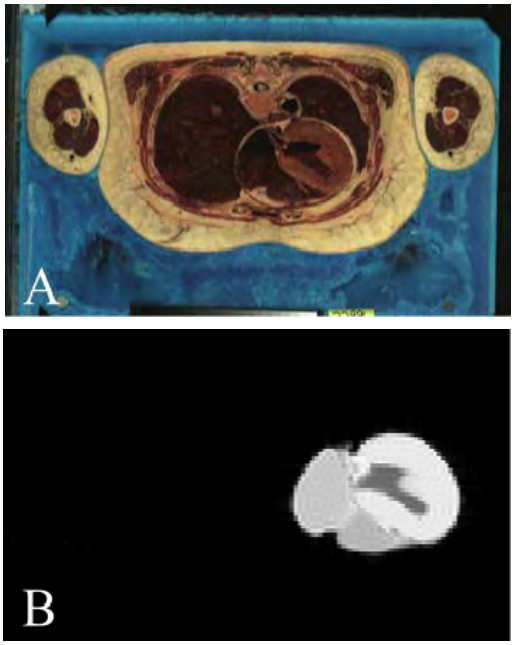

Fig. 3. (A) Thorax image of the Visible Human Project dataset. (B) Simulated MR image of the human heart resulted from the application of the pulse sequence on the computer model of the anatomy of the human heart at the same slice location as image (A).

IV. DISCUSSION

A multi-GPU approach of MRISIMUL was presented in this study. The almost linear scalable performance that MRISIMUL demonstrated with the increasing number of available GPU cards on single-node, multi-GPU computer systems enhances its application in large-scale complicated experiments without model simplifications. Moreover, it suggests that an even better performance may be achieved in multiple nodes, multi-GPU computer systems. However, for those cases, a further investigation of the inter-node communication protocol shall be performed in the future. A web-based version of MRISIMUL, along with its latest releases, is available online under mri.dib.uth.gr.

REFERENCES

[1] H. Benoit-Cattin, G. Collewet, B. Belaroussi, H. Saint-Jalmes, and C. Odet, "The SIMRI project: a versatile and interactive MRI simulator," *J Magn Reson*, vol. 173, pp. 97-115, Mar 2005.

[2] T. Stocker, K. Vahedipour, D. Pflugfelder, and N. J. Shah, "High-performance computing MRI simulations," *Magn Reson Med*, vol. 64, pp. 186-93, Jul 2010.

[3] K. G. Baum, G. Menezes, and M. Helguera, "Simulation of High-Resolution Magnetic Resonance Images on the IBM Blue Gene/L Supercomputer Using SIMRI," *International Journal of Biomedical Imaging*, vol. 2011, 2011.

[4] C. G. Xanthis, I. E. Venetis, and A. H. Aletras, "MRISIMUL: A GPU-based Parallel Approach to MRI Simulations " *Manuscript submitted for publication*, 2013.

[5] M. J. Ackerman, "The Visible Human Project: a resource for anatomical visualization," *Stud Health Technol Inform*, vol. 52 Pt 2, pp. 1030-2, 1998.

Accelerated MR simulations of heart motion model using graphic processing unit technology

Introduction

Magnetic Resonance Imaging (MRI) physics simulators have been used in the past for educational and research purposes. Current MRI physics simulators make several compromises and only few of them incorporate motion in their simulations due to the added computational power needed. Moreover, study of motion is usually confined to simple translational and/or rotational motion models whereas motion is being activated only during the read-out phase of the pulse sequence.

A MR physics simulator that supports simulations of realistic, motion-related cardiac MR experiments may help in evaluating cardiovascular MRI protocols and pulse sequences.

Methods

In this study, a previously developed MR physics simulation platform, by the name MRISIMUL, adapted so as to allow the temporal evolution of a heart motion model in space during the entire course of the MR pulse sequence. MRISIMUL allows its application in large-scale analysis without model simplifications by employing Graphics Processing Unit [GPU] technology.

For the simulation of heart motion, a mathematical model of myocardial deformation was applied on a cylinder based on work of Tecelao et al (2006). 1D myocardial tagging was introduced in a simple segmented CINE pulse sequence of 100 phases and 1 view per segment (total timesteps: 123456) and applied on the heart motion model (total voxels: 123456). The simulation was performed on a computer system with 4 Tesla C2075 GPU cards (448 GPU cores, 6GB global memory) and the kernel execution time was recorded.

Results

Myocardial tagging of the heart computer model on different phases of the cardiac cycle is displayed in Figure 1. The kernel execution time was recorded equal to approximately 419 minutes.

Conclusion

MR simulation of realistic heart motion-related experiment on multi-GPU system was presented in this study revealing new future capabilities on investigating motion artifact sources, developing new motion compensation techniques but also for training purposes.

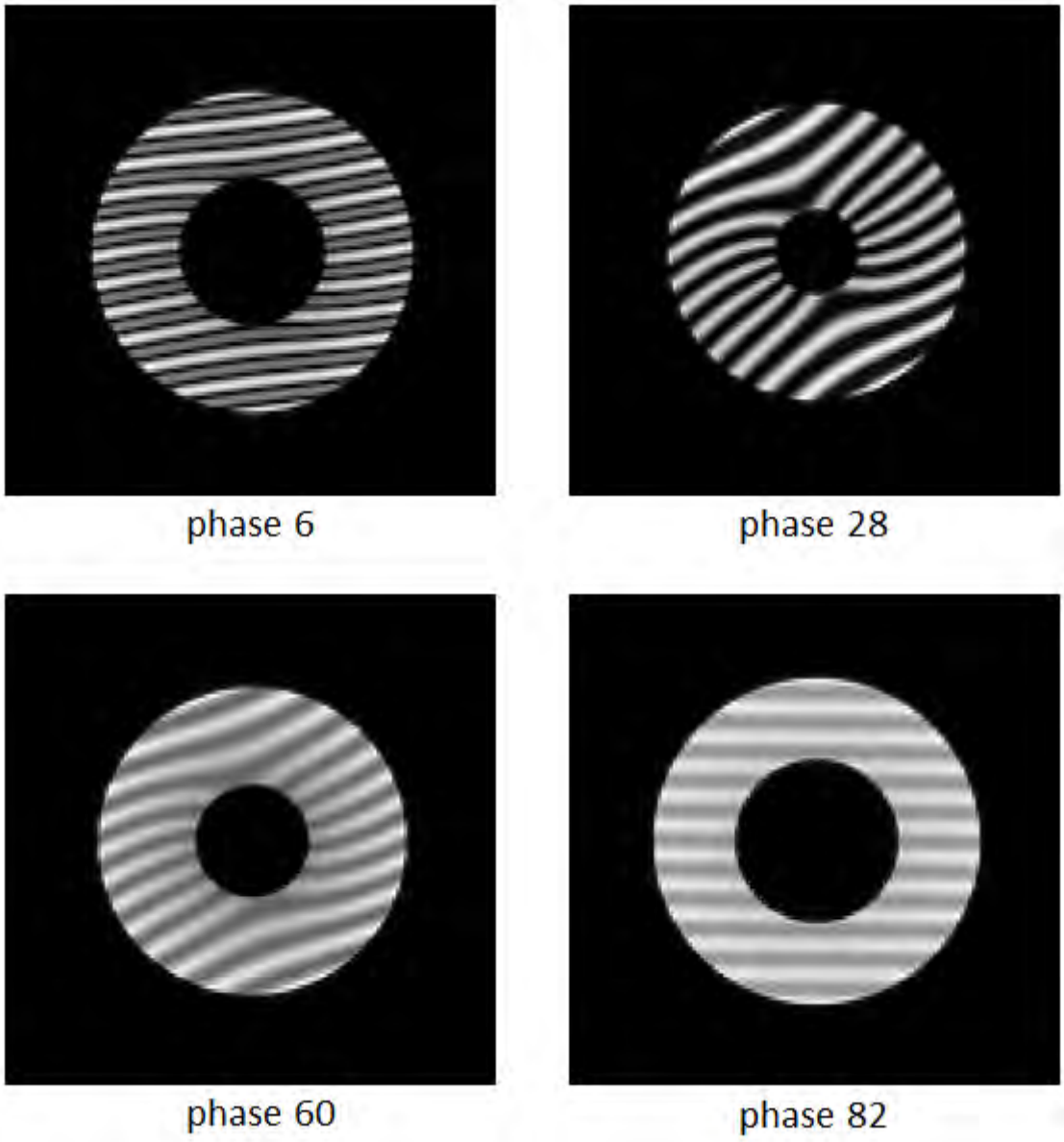

Figure 1. Series of images of the heart motion model throughout the cardiac cycle. SPAMM preparation with 90o RF flip angle has been introduced in a simple segmented CINE pulse sequence of 100 phases and 1 view per segment.

Segmentation and visualization of cardiovascular tree using high performance graphics processing cards

Introduction

Time resolved three dimensional phase contrast MRI (4D-PCMRI) is a versatile tool for understanding complex intracardiac blood flow. 4D-PCMRI may be used without contrast agents to segment and visualize the cardiovascular tree, adding clinical utility. However, data processing is currently time consuming and requires high computational power.

Therefore, the purpose of this study was to develop a high performance, fully automatic vessel segmentation algorithm that exploits the properties of 4D-PCMRI flow methods in the absence of gadolinium administration.

Methods

A previously developed method from Van Pelt et al (2012) was adapted and optimized for performance by employing the Graphics Processing Unit [GPU] technology. The method was implemented in MATLAB whereas the computationally demanding core was executed in parallel within the GPU by exploiting CUDA technology. The GPU implementation involved calculation of the probability map for vessel visualization based on 4D PC-MRI data. Segment and Fourflow were used for data handling and visualization (figure 1) respectively.

The efficacy of GPU implementation was evaluated on three different GPU configurations and compared against the serially executed code on the CPU. Execution times were recorded for a) a GeForceGT430 (96 GPU cores, 1GB global memory) b) a Quadro K5000M (1366 GPU cores, 4GB global memory) and c) a Tesla C2075 (448 GPU cores, 6GB global memory).

Results

Speedup ranged from 24 times faster (GeForceGT430) up to 229 times faster (Tesla C2075) when compared to CPU-based calculations. Total execution time was decreased from >20min to few seconds (<1min), and the speedup depended on computer architecture.

Conclusion

A GPU-based algorithm for segmentation of the cardiovascular tree was described in this study. The utilization of GPU cards demonstrated significant speedup, which in turn, may enhance the applicability of this algorithm as a significant clinical tool in the future.