NTAV/SPA 2012
27-29TH SEPTEMBER, 2012, ŁÓDŹ, POLAND
NEW TRENDS IN AUDIO AND VIDEO / SIGNAL PROCESSING ALGORITHMS, ARCHITECTURES, ARRANGEMENTS AND APPLICATIONS

Computer Simulation of Magnetic Resonance Angiography Imaging. Parallel Implementation in Heterogeneous Computing Environment

Artur Klepaczko, Piotr Szczypiński, Grzegorz Dwojakowski, Marek Kociński, Michał Strzelecki
Lodz University of Technology
Institute of Electronics
90-924 Lodz, ul. Wolczanska 211/215
Email: aklepaczko@p.lodz.pl

***Abstract* — Magnetic resonance imaging (MRI) is currently widely used in medical image diagnosis. However, MR scanners are extensively used in clinics and thus are rarely accessible for experimentation. In consequence, the number of images available for image processing methods evaluation is too low and there appears a need for a method to generate synthetic images. In their previous works, the authors studied various methods for blood vessels segmentation and tracking. Effectiveness of the designed algorithms requires objective verification which implies repetition of experiments for large number of images and comparing the results with some ground truth models. Therefore, this study aims at designing a computer system which implements numerical routines for generation of synthetic MRA images. In particular, in this paper we study the performance of various configurations of assembled computer grid and analyze their potential in angiographic image synthesis.**

I. Introduction

Magnetic resonance imaging (MRI) is currently widely used in medical image diagnosis. This non-invasive technique allows acquisition of high-resolution volumetric images which visualize even small details of biological tissues precisely positioned relative to the interior of the whole body. On the other hand, recent advances in the development of image processing algorithms facilitates their usage in quantitative anlysis of MR images. If conducted properly, such analysis can augment objectivity and correctness of the medical diagnosis. However statistically credible validation of the designed methods is hampered by the significant cost of collecting MR images solely for the research purposes. Additionally, MR scanners are extensively used in clinics and thus are rarely accessible for experimentation. In consequence, the number of images available for evaluation studies is too low and there appears a need for a method to generate synthetic images. One of the approaches that helps to overcome these impediments is computer simulation of magnetic resonance imaging [1].

In particular, we put focus on magnetic resonance angiography (MRA) directed at visualization of the human vessel system. In their previous works [2], [3], the authors studied various methods for blood vessels segmentation and tracking in 3D MR images acquired using Time-Of-Flight (ToF) or Susceptibility Weighted Imaging (SWI) sequences. Effectiveness of the designed algorithms requires objective verification which implies repetition of experiments for large number of images and comparing the results with some ground truth models.

Therefore, this study aims at designing a computer system which implements numerical routines for generation of synthetic MRA images. This general aim can be decomposed into several subtasks which include:

1) definition of the virtual object containing synthetic vessel branches and characterized not only by the magentic properties (such as spin-spin and spin-lattice relaxation time constants, hydrogen proton density, tissue magnetic susceptibility) but also by the hydrodynamic parameters of a substance flowing through virtual vessels;
2) blood flow simulation which generates information about displacement of moving voxels during MR image formation;
3) simulation of electromagnetic phenomena which take place in real MR scanner spin selective excitation, precession, relaxation, signal induction, etc.
4) image formation k-space filling, filtering in the spatial-frequency domain, application of the Fourier transform.

The above described tasks are computationally extensive, especially if the target image is a 3D high-resolution data structure. Most existing MRI simulators are implemented using various architecture computer grids. Thus, for the need of our project and as a first step towords the target system, we designed our own parallel implemention of the core MRI simulator, i.e. no flow effects are taken into account. The objective was to use the available resources and thus gain guaranteed, flexible, on-demand access to it in any future exeperiments and be independent from publicly available, but of limited access, open grid projects. As a consequence, our runtime environment is developed as a heterogeneous grid of UNIX/LINUX machines which communicate with one another over local ethernet network using the SSH protocol.

In this paper, we study the performance of various configurations of our computer grid and analyze its potential in angiographic image synthesis. In the reminder of this paper

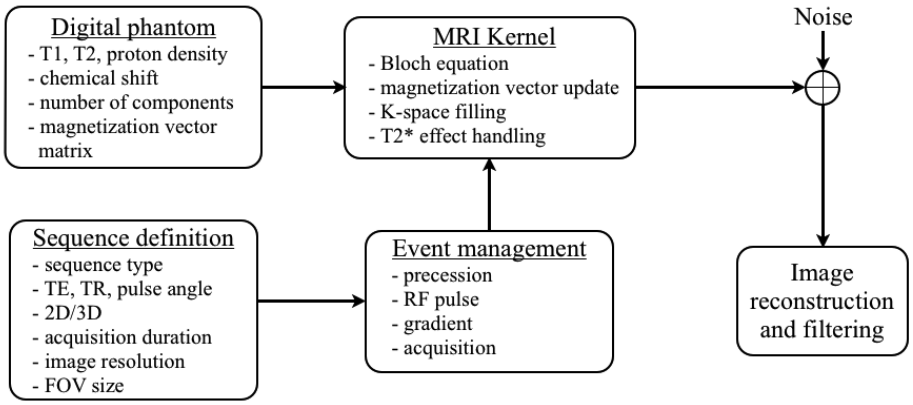

Fig. 1. Main software modules of the MRI simulator

we describe the overall concept of the MRA simulator and main parts of the system (Sect. II). Then in Sect. III, we provide here the description of how we plan to deal with the flow phenomena, so that angiographic images can be simulated. In Sect. IV we give technical specification of the computer grid used in the experiments, as well as the details of simulator parallel implementation based on the MPI standard. The performance tests are reported in Sect. V and finally Sect. VI concludes.

II. MRI SIMULATION

Difficulties in acquistion of large volumes of real MRI images solely for research purposes constitute the main reason why over the last two decades there appeared a number of attempts to build computer MRI simulators. The proposed solutions [4], [5], [6], [7] differ from one another by the approach to modelling tissue of imaged objects, the MR image synthesis routines, and the degree to which different artifactual or undesired phenomena (like noise, chemical shift, ringing or Gibbs phenomenon, partial volume effect etc.) are taken into account.

In the approach proposed in [8], the imaged objects are defined as parameter maps. These maps visualize distribution of T1 (spin-lattice relaxation time constant), T2 (spin-spin relaxation) and rho (proton density) values within the object. These values are determined based on real images. Formation of new ones is performed using different repetition and echo times.

On the other hand, in paper [9], the virtual organ object is modelled using the concept of spin density. By the use of inverse Fourier Transform, the simulator constructs the image representation in the spatial-frequency domain (k-space formalism). Simulation of the signal sampling phase and specific imaging sequence is accomplished through appropriate selection of the k-space elements and their amplitude and phase alteration. It is raised that this approach requires separate simulation phases for each of the modelled tissues. This introduces significant impediment if a voxel contains more than one tissue (i.e. partial volume effect).

In comparison with other approaches, more realistic simulation effects are achieved by solutions proposed in [5], [6], [7]. Models of the imaged objects used therein require for each object voxel and every tissue component definition of T1, T2, and rho parameters, as well as value of the frequency offset linked to chemical shift artefact. In order to deal with the partial volume effect it is sufficient to determine how much of particular tissue material is present in a voxel.

The simulator used in our study is based on the SIMRI (fr. *SIMulateur de MRI*) project described in [5], due its versatality and proven effectiveness in producing realistic MR images. Moreover, it seems to be best suited – after necessary modifications – to our proposed angiography simulator. The main software modules of the originally proposed system – reimplemented by our team – are presented in Fig. 1.

A. Digital phantom description

The input to the simulator is the virtual object composed of 3D rasterized components, each of which corresponds to a different tissue type. Tissues are described by their spin-lattice (T1) and spin-spin (T2) relaxation times, as well as proton density (ρ) values. Every component is also described by the water-relative frequency shift which allows simulation of the chemical shift artifact. This value is determined assuming that the main magnetic field B_0 = 1T. Moreover, each object voxel is associated a 3D magneitzation vector, whose state at subsequent stages of image formation is controlled by the MRI simulation kernel. In addition to raster size (number of voxels in x, y and z coordinates), the object is characterized by its physical dimensions. Thus, every object voxel accomodates information about portion of the tissue component it contains, (T1,T2,ρ)-tuples for each component, and current magnetization vector state being a sum of magnetization vectors calculated for every tissue.

B. Sequence definition

The image synthesis procedure is controlled by the group of paramaters which set up sequence type and output image contents. The latter refers both to the image resolution which can be lower or equal to the input virtuual object raster size and to the position and size of the field of view (FOV). By defualt FOV embraces whole object, but it can be reduced to only a part of it and its center can be moved to any object point. This is important to keep the record of the FOV center offset and its location relative to gradients isocenter, since it affects the gradient value experienced by each voxel.

The sequence definition is mainly determined by the sequence type, which cen be either spin echo (SE) or gradient echo (GE) imaging. In both cases, the program requires specification of the echo and repetition times (TE and TR accordingly). In the case of GE, it is also necessary to define the flip angle (α). The sequence type also include information whether the target image is 2- or 3-dimensional. For the 2D image, one has to specify the slice position (x coordinate). Eventually, the signal sampling window is determined by the acquisition time which must ensure that the sampling frequency meets the Nyquist theorem.

C. Event management

The event management module is responsible for invoking subsequent steps of computations based on the current simulation stage. It begins with establishing the initial magnetization of the object by allowing all of its spins to freely precess in the presence of the main external magnetic field B_0. Then, depending on the sequence type, the object magnetization state is altered through the application of an RF pulse, free precession, refocussing pulse (SE imaging only) and application of phase encoding gradients (in y and z directions in the case of volumetric imaging or only in y direction for the 2D case). Eventually, the frequency encoding gradient is applied and the signal acquisition step is executed. At each step the MRI kernel is invoked with the appropriate parameters.

D. MRI kernel

The heart of the whole system is the MRI kernel which implements the discrete time solution to the Bloch equation [10]. It uses the rotation matrices and exponential scaling to modify the voxel magnetization vectors in accordance to the specified sequence events. For the details of kernel implementation we refer the reader to the original paper [5] – here we recall only the main bits of this system module.

First of all, the magnetization vector $\vec{M}$ at location $\vec{r} = [x, y, z]$ is given by

$$\vec{M}(\vec{r}, t + \Delta t) = Rot_z(\theta_g) Rot_z(\theta_i) R_{\text{relax}} R_{\text{RF}} \vec{M}(\vec{r}, t), \quad (1)$$

where $Rot_z(\theta)$ rotates the magnetization vector around the z-axis in reply to the phase encoding gradient (θ_g) and as a consequence of the main magnetic field inhomogeneity (θ_i). The R_{relax} rotation matrix is responsible for the relaxation effects and is given as

$$R_{\text{relax}} = \begin{bmatrix} e^{-\frac{\Delta t}{T_2(\vec{r})}} & 0 & 0 \\ 0 & e^{-\frac{\Delta t}{T_2(\vec{r})}} & 0 \\ 0 & 0 & 1 - e^{-\frac{\Delta t}{T_1(\vec{r})}} \end{bmatrix}. \quad (2)$$

Eventually, the R_{RF} encapsulates the effect of the RF excitation pulse which flips the magnetization vector around an axis belonging to the xy-plane by the angle in time Δt. In the presence of gradient, the effective flip angle is α' and R_{RF} is calculated as

$$R_{\text{RF}} = Rot_z(\phi) Rot_y(\beta) Rot_x(\alpha') Rot_y(-\beta) Rot_z(-\phi), \quad (3)$$

where

$$\alpha' = -\Delta t \sqrt{(\Delta\omega)^2 + \left(\frac{\alpha}{\tau}\right)^2},$$

$$\beta = \tan^{-1}\left(\frac{\Delta\omega}{\alpha/\Delta t}\right),$$

and $\Delta\omega$ is the local frequency offset from the main static magnetic field induced by gradients, RF pulse and field inhomogeneities.

Signal acquistion is performed by aggregating the transverse magnetization components over all object voxels and this summation is repeated the number of times needed to fill one K-space line. Between each two acquistions, the voxel magnetization states are updated due to relaxation effects that take place during the sampling period δt. Thus, one point $s(t)$ of the K-space line is calculated as follows

$$s(t) = \sum_{\vec{r}} \vec{M}(\vec{r}, t)\vec{x} + j \sum_{\vec{r}} \vec{M}(\vec{r}, t)\vec{y}. \quad (4)$$

The MRI kernel is actually much more versatile than described here. It offers T2* effect handling, variety of pulse wave forms, gradient crushing, numerical transverse magnetization spoiling, can take into account various artefatcs linked to static field inhomogeneities, field default intensity values or tissue susceptibility. Apart from standard SE and GE sequences, there are more sophisticated sequences implemented, such as e.g. True-FISP technique (Fast Imaging with Steady-state Precession). It also allows adding noise to the K-space data, although in the current project phase, this feature is restricted to white Gaussian noise only, which is a simplification of the true MR noise characteristics. As indicated in [11], the better model here would incorporate the Rice function.

E. Image reconstruction and filtering

The last stage of image formation procedure consists in transformation of the raw data in the spatial-frequency domain (K-space) into image intensity domain. This is accomplished by the fast Fourier transform, followed by calculation of the magnitude of – in general – complex tranformation output. Before application of the FFT routine, it may be necessary to filter the raw data to reduce the Gibbs artifact in the case of small resolution images. We decided to port the filtering routine (as welll as FFT-based image reconstruction) to the Matlab environment and thus we can use any kind of low-pass digital filter available in the Signal Processing Toolbox [12].

III. Extension to angiography imaging

The major modification we need to introduce to the SIMRI simulator concerns the virtual object, which now must include information about replacement of the flowing blood. We decided to determine this information using a separate program dedicated to flow simulation. For that purpose we employed the COMSOL Multiphysics 4.2a software [13]. In our initial experiments we used a simple digital flow phantom, composed of a single cylindrical tube of impermeable boundaries placed in porous material, able to absorb arbitrary volume of liquid. Since under the framework of this paper we are mostly interested in MR image formation process, the flow phenomena are not covered here and will be discussed in a separate work.

It must be noted, that the flowing blood particles move at different speed, depending on their position within a vessel and the vessel size. Replacement of the slower particles during a single simulation time step (such as a sampling period e.g.) may be less than a voxel. This breeds a need for even a deeper remodelling of the digital phantom than simple addition of per-voxel flow information. The two feasible solutions for this issue is to perform computations at sub-voxel level or redefinition of the object (at least for the moving component) so that instead of voxels arranged in a regular lattice it is composed of particles, each storing its current position within the object volume. Because the prior startegy does not solve the problem entirely but only reduces it to the limit of the voxel quantization scale, we decided to focus on the second approach.

The model of object constructed as a set of particles requires that each particle is associated a portion of media – call it a cell – it represents. Contribution to the output image coming from each separate cell should be proportional to its size. Therefore, after each time step and magnetization vectors update, our system is designed to recalculate new mesh of simplices (cells) linked to the moving particles. We chose the 3D Delaunay triangulation [14] method to perform this operation. The triangulation is invoked on the points corresponding to current particle locations and static points placed all over the vessel walls.

The signal acquistion procedure for the moving liquid compnent is executed similarily to stationary tissues, but now the sum in Eq. (4) is taken over cells and not over regularly distributed voxels. Moreover, each element of the sum is weighted by the factor which relates volume of a cell to the volume of an output image voxel. Note, that the K-space structure remains a regular grid of voxels, as the signal sampling procedure is independent from the object internal definition. Eventually, the collected K-space data are simply added to complex signal volumes built for the stationary tissues.

Extension to angiography imaging sequences is still under development. Thus, the rest of the paper presents the performance of the designed cluster environment emplyed for the synthesis of basic MRI images.

IV. Cluster environment

As noted in the introduction, we aimed at constructing our own computer cluster built on available hardware resources instead of utilizing any of the exisiting and publicly avaialble remote grid projects. In the latter case, the access to grid execution units is subject to specific rules which define job submission scheduling process, software interface for parallel programming, total amount of working hours granted to a user, etc. In the current phase of our project, which is still under intensive development, it is preferable to have on-demand access to effiicient computing enviroment allowing for instant testing of any of the incremental and major system modifications.

A. Cluster configuration

The network of computers used in our simulations is composed of UNIX and LINUX machines of various architectures. The grid is composed of two server units (both possesing a pair of Intel quad-core Xeon CPUs), one 20 iMac (with double-core Intel Core Duo), one 27 iMac (quad-core Intel Core i7) and 6 MacMini computers, each equipped with an Intel Core i5 processor. Enabled with the hyperthreading feature, the i7 CPU is actually able to run 8 threads in parallel, while double-core i5 can run 4 threads simultaneously. This gives effectively 50 cores of 134,4 GHz total computing power and potentially 537.6 GFlops of peak performance. Note, that both server computers belong to the i386 architecture, while other units to the x86_64 platform. This variation in operating systems and hardware configuration makes the designed cluster a heterogeneous computing environment. As a consequence, flexibile communication protocol, transparent for both Linux and Mac OS X machines, is needed to ensure efficient and reliable parallel job execution. In conjuction with the Open MPI library that we used to parallelize the simulator, it occurred that the SSH protocol provides the easiest way to establish the grid.

B. Parallelization using Open MPI library

The Message Passing Interface (MPI) is the well-recognized standard in the domain of high performance computing. It is specifically designed for use in cluster/grid systems with distributed memory model, i.e. where each processing unit has its own memory addressing space. The MRI simulator developed under the SIMRI project was based on the MPICH-G2 [15] distribution. However, we decided to build our implementation around the Open MPI library [16] due to its better integration with Mac OS X systems and documented support for the SSH protocol.

Paralellization of the MRI simulator slightly differs for motionless (stationary tissue) and moving (blood) object components. In the first case imaged 3D volume is devided into slices along the x dimension. Each node of the grid performs signal acquisition for the whole 3D k-space, but it takes into account only the contribution coming from the object voxels belonging to this nodes slice. Then, the root node collects the k-space volumes from all the nodes and they are aggregated.

After completing calculations for one component it moves on to the remaining ones.

Because, the object component corresponding to blood flowing through vessels is designed differently, it eequires separate parallelization model. Instead of deviding the object into slices, every node receives a portion of cells to handle. A node is thus responsible for updating cell position and recalculation of its corresponding magnetization vector. During acquistion step, the k-space is filled by summing up signals coming from object cells managed by a particular node.

It is worth to stress that this strightforward parallelization model further substantiates the assumed approach to moving object component defintion. Each object cell constitutes an autonomous agent and all information connected with its replacement and magnetic vector state can be derived within a grid node. If it were the alternative approach based on subvoxel analysis, information about magnetization of flowing liquid would have to be somehow transmitted to neighboring voxels belonging to different nodes slices. This would involve additional communication between working nodes and inevitable transmission synchronization issues leading to downgraded performace.

C. Runtime parameters

In order to launch jobs on SSH-based grid using Open MPI software, it is important to properly setup user accounts on all machines. First of all, a user of the same login name should be registered on every node. Preferably, this user should be granted administrative priviliges. User logging between computers should proceed without password authentication, as described in the Open MPI documentation [16]. To achieve this, one has to generate RSA key pair on the root node, copy the public key to the *authorized_key* file, and eventually distribute *./ssh* directory to the specified user home directories on all remote nodes.

Moreover, the Open MPI distribtion – the same on all nodes at least up to the major subversion number – should be installed at the identical location. Similarily, the MPI program has to placed under the same absolute path on every computer. The progam itself, though uniformly named too, should be architecture and OS specific. Thus, in our grid the same executable is shared between both iMacs and Mac Mini units, but there are separate compilations for Xserve and the Linux server.

To simplify file exchange between nodes, it is possible to establish common file system for them. For this purpose, we utilized the NFS (Network File System) protocol. On the root node one has to create a shared folder and place its path in the */etc/exports* file with appropriately set network address space. Then, on the remote hosts, this shared resource needs to be mounted using *mount -t nfs* shell command.

Eventually, the command which invokes a parallel job using MPI and on the SSH-based grid takes the following form:

```
sh$ $MPI_HOME/mpirun
    --mca plm_rsh_agent ssh
    --hostfile $JOB_HOME/hosts
    --np 50
    $JOB_HOME/simra_mpi
    $NFS_COMMON/job_paramaters_file
```

where `$MPI_HOME` indicates the installation directory of the MPI distribution, shared `$NFS_COMMON` directory contains any addditional files required by the executed program (object definition, flow information, MRI sequence parameters) and `$JOB_HOME` is the path to the parallelized executable. The `--mca` switch states that it is the SSH agent that controls the grid management, the hosts file contains the list of computers in the grid, and the `--np` switch specifies the number of processes to run in parallel.

V. Evaluation of grid performance

In order to assess efficiency of the designed computer grid, we have conducted a series of simulations for one digital phantom at different scales of the grid. For every scale, there was a part of CPU cores constantly present across various configurations (Xserve, Linux server, and iMacs) and the rest (Mac Mini computers) were accumulatively added in subsequent simulations.

The phantom used in the experiments is composed of two virtual tissues whose voxels are spherically distributed arround common origin. The resolution of the object is 100 voxels in each dimension, while the physical size were set to $(0.2m)^3$.The intesity levels of the object components range from 0 to 125. These limits results from the assumed method of phantom synthesis. First, one chooses the minimum and maximum radii of each component boundary spheres. Secondly, a voxel is devided into subvoxels, 5 in each direction, giving 125 subvoxels in total. If a center of a subvoxels lies within the assumed limits, then it is added to the object component. The number of subvoxels included determines intensity of its corresponding voxel.

Table I summerizes the details of the object and its components. Note, that T_1, T_2 and ρ parameters are set to values typical to fat and water ((components no. 1 and 2 respectively). Average simulation times achieved to synthesize MR images are shown in Table II, while synthesized images (GE, T_2*-weighted) are presented in Fig. 2. Note that relatively strong T_2* weighting of the image lead to almost entire supression of the signal related to the fat component. On the contrary, water with high T_2 constant appears bright on the image.

VI. Conclusion

Analysis of the obtained time measurements (see Table II) proves efficiency of the assembled computer grid. Also, grid management and inter-process communication controlled by Unix/Linux machines guarantees reliable and stable operation. Computational speedup scales (almost) linearly with the number of CPU cores. Time needed to synthesize high-resolution volumetric images ranges on average from 1.5 up to 2 hours per component, depending on the object size. This appears to be acceptable, especially if compared to the results achieved by other simulators. This leads to conclusion that our

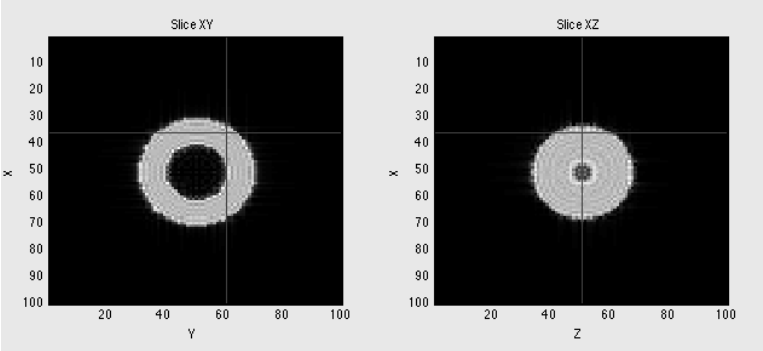

Fig. 2. Synthesized MR image. Gradient echo sequence, TE=15 ms, TR=120 ms, flip angle = 20°

TABLE I
DIGITAL PHANTOM DESCRIPTION

Component No.	$r_{\min}$[1]	$r_{\max}$[2]	ρ[3]	T_1	T_2
1	2 cm	4 cm	1	350	70
2	4 cm	6 cm	1	2569	329

[1, 2] minimal and maximal sphere radii
[3] water relative proton density

TABLE II
GRID PERFORMANCE EVALUATION

Number of cores	26	30	34	38	42	46	48
Time $\times 10^3[s]$	12,9	12,3	12,0	11,4	11,1	10,7	10,2

gird can be expected to produce angiographic MR images in reasonable times. However, if the number of available cores occurs insufficient, the archicture of the grid allows its further expansion on additional units.

ACKNOWLEDGMENT

This work was supported by the Polish National Science Centre grant no. 6509/B/T02/2011/40.

REFERENCES

[1] R. Kwan, A. Evans, and G. Pike, "An extensible MRI simulator for post-processing evaluation," in *Visualization in Biomedical Computing*, ser. Lecture Notes in Computer Science, K. Hoehne and R. Kikinis, Eds. Springer Berlin / Heidelberg, 1996, vol. 1131, pp. 135–140. [Online]. Available: http://dx.doi.org/10.1007/BFb0046947

[2] A. Materka, M. Strzelecki, P. Szczypinski, M. Kocinski, A. Deistung, and J. Reichenbach, "Arteries tracking in simultaneous TOF-SWI MR images: image characteristics and preliminary results," in *Image and Signal Processing and Analysis, 2009. ISPA 2009. Proceedings of 6th International Symposium on*, 2009, pp. 748 –753.

[3] M. Strzelecki, P. Szczypiński, A. Materka, M. Kociński, and A. Sankowski, "Level-set segmentation of noisy 3D images of numerically simulated blood vessels and vascular trees," in *Image and Signal Processing and Analysis, 2009. ISPA 2009. Proceedings of 6th International Symposium on*, 2009, pp. 742 –747.

[4] D. A. Yoder, Y. Zhao, C. B. Paschal, and J. M. Fitzpatric, "MRI simulator with object-specific field map calculations," *Magnetic Resonance Imaging*, vol. 22, pp. 315–328, 2004.

[5] H. Benoit-Cattin, G. Collowet, B. Belaroussi, H. Saint-Jalmes, and C. Odet, "The SIMRI project: a versatile and interactive MRI simulator," *Journal of Magnetic Resonance*, vol. 173, pp. 97–115, 2005.

[6] T. Stoecker, K. Vahedipour, and N. J. Shah, *HPC Simulation of Magnetic Resonance Imaging*, ser. NIC Series. Juelich: John von Neumann Institute for Computing, 2007, vol. 38, pp. 155–164.

[7] K. Jurczuk and M. Kretowski, "Virtual magnetic resonance imaging – parallel implementation in a cluster computing environment," *Biocybernetics and Biomedical Engineering*, vol. 29, no. 3, pp. 31–46, 2009.

[8] A. Simmons, S. Arridge, G. Barker, and S. Williams, "Simulation of MRI cluster plots and ap," *Magnetic Resonance Imaging*, vol. 14, no. 1, pp. 73–92, 1996.

[9] J. S. Petersson, J. O. Christoffersson, and K. Golman, "MRI simulation using the k-space formalism," *Magnetic Resonance Imaging*, vol. 11, no. 4, pp. 557–568, 1993. [Online]. Available: http://www.sciencedirect.com/science/article/pii/0730725X9390475S

[10] Z.-P. Liang and P. C. Lauterbur, *Principles of Magnetic Resonance Imaging*, ser. Series in Biomedical Engineering. IEEE Press, 2000.

[11] P. Tofts, *Quantitative MRI of the Brain: measuring changes caused by desease*. Chichester: John Wiley & Sons, 2003.

[12] MATLAB, *version 7.13.0 (R2011b)*. Natick, Massachusetts: The MathWorks Inc., 2011.

[13] C. Multiphysics, *version 4.2.1 (4.2a)*. Los Angeles, CA: COMSOL, Inc., 2012.

[14] M. de Berg, O. Cheong, M. van Kreveld, and M. Overmars, *Computational Geometry: Algorithms and Applications*. Springer-Verlag, 2008.

[15] N. Karonis, B. Toonen, and I. Foster, "MPICH-G2: A grid-enabled implementation of the Message Passing Interface," *Journal of Parallel and Distributed Computing*, vol. 63, no. 5, pp. 551–563, 2003.

[16] E. Gabriel, G. E. Fagg, G. Bosilca, T. Angskun, J. J. Dongarra, J. M. Squyres, V. Sahay, P. Kambadur, B. Barrett, A. Lumsdaine, R. H. Castain, D. J. Daniel, R. L., and T. S. Woodall, "Open MPI: Goals, concept, and design of a next generation MPI implementation," in *Proceedings, 11th European PVM/MPI Users' Group Meeting*, Budapest, Hungary, 2004, pp. 97–104.