MR simulator to create tailored datasets of MR Images for AI Algorithm Development.

Zaidan Bhat (B18040), Aakash Maurya (B18042)

I. INTRODUCTION

A. Background and Motivation

When it comes to detection of tumors and other medical anomalies, MRI is one of the most useful technologies available with us today. It uses magnetic properties of different tissues to create images where different types of tissues are distinguishable. In the recent years, as Machine learning and artificial intelligence continue to develop, their use in detection of tumors is a hot research topic. Using AI for detection eliminates chances of human error. One major problem is that training AI models requires large amounts of labelled data. We do not have large amounts of labelled MRI data available. Most hospitals do not have the systems and infrastructure to save and share labelled MRI data. Most of the data is lost. Patient privacy concerns further add to the difficulties in obtaining large amounts of data for training AI/ML models. Obtaining this data from MRI machines is a complex, labor-intensive and costly process. This has created a need for an MR simulator.

B. Literature Review and Present Work

One such simulator is MRiLab. The MRiLab is a numerical MRI simulation package. It has been designed and optimized to simulate MR signal formation, k-space acquisition and MR photo reconstruction. MRiLab offers numerous dedicated toolboxes to analyze RF pulse, layout MR sequence, configure transmitting and receiving coils, look at magnetic field related properties and evaluate real-time imaging approach. The primary MRiLab simulation platform blended with those toolboxes can be utilized out to customize diverse digital MR experiments which can serve as a previous stage for prototyping and trying out new MR method and software. The MRiLab capabilities include a tremendously interactive graphical user interface (GUI) for the benefit of speedy experiment design and technique prototyping. High simulation accuracy is obtained through simulating discrete spin evolution at small time interval the use of the Bloch-equation and appropriate tissue model. In order to manipulate multidimensional spin array, MRiLab employs parallel computing by incorporating state-of-the-art graphical processing unit (GPU) approach and multi-threading CPU technique. With efficient parallelization, MRiLab can accomplish multidimensional multiple spin species MR simulation at high simulation accuracy and time efficiency, and with low computing hardware cost.

II. BASICS OF MRI

MRI uses the magnetic properties of tissues to create images. Different tissues with different magnetic proprieties generate signals of varying strengths which helps us differentiate tissues. The signals are generated in response to pulses. There are different types of sequences that can be run. One basic sequence is the spoiled gradient echo sequence. In the experiments we carried out this is the sequence we have used. In this sequence, any residual magnetization left after a pulse is removed before another pulse if given.

Some important MRI parameters used in the below mentioned experiments are:

- TE: Time to Echo (TE) is the time taken to recieve the echo signal after delivery of the
- TR: Repetition Time (TR) is the amount of time between successive pulse sequences applied to the same slice
- FA: The flip angle is an MRI phenomenon by which the axis of the hydrogen proton shifts from its longitudinal plane (static magnetic field B0) Z axis to its transverse plane XY axis by excitation with the help of radiofrequency (RF) pulses

III. EXPERIMENTS USING MRILAB

Our experiments were confined to the Spoiled Gradient Echo Sequence for the Brain High Resolution Phantom and the Water Fat Phantom. For both the phantoms we did experiments varying various parameters. All the experiments were carried out for three noise levels 0, 5, 10. For Each Noise level the following experiments were carried out.

- Vary TE while keeping all other parameters constant (TE = 5 - 100 ms; steps: 10 ms (5,10,20,30,....) , TR = 500 ms, FA = 30°).
- Vary TR while keeping all other parameters constant (TR = 25 - 500 ms; steps: 25 ms) , TE = 5 ms, FA = 30°).
- Vary FA while keeping all other parameters constant (FA 5° - 90°, steps: 5°; TE = 5 ms, TR = 500ms).
- Vary FA while keeping all other parameters constant (FA 5° - 90°, steps: 5°; TE = 5 ms, TR = 40ms).

Results of the above experiments provided 80¿¡100 images. Certain pixels were selected to correspond to various tissues. In all the experiments the pixels chosen remained the same, so did the slices and the field of view. This was to ensure uniformity in the experiments.

Pixels Selected For signal and Noise computation from images obtained from the experiments

- For Noise: Four Patches(i= [4-8], j= [4-8]), (i= [72-76], j= [4-8]), (i= [4-8], j= [92-96]), (i= [72-76], j= [92-96]) (64 total Pixels)
- For CSF Signal: One patch (i= [38-42], j= [38-42]) (16 total pixels)
- For White Matter: One Patch (i= [38-42], j= [23-27]) (16 total pixels)

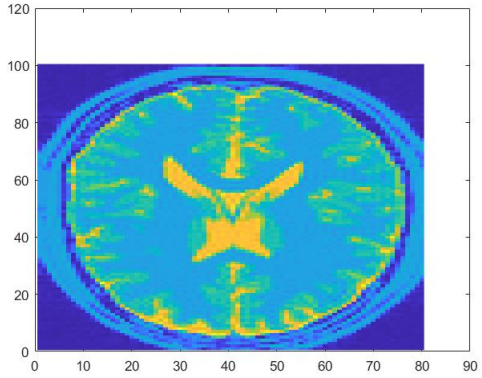

Fig. 1. A sample output image with dimensions

Results of the above experiments were loaded into python and mean values of signal for various tissues(CSF and White Matter) were computed. Expected signals were also computed using theoretical equation for computation of signal in a spoiled gradient echo sequence. Parameters that indicate magnetic proprieties(T1, T2 and T2*) for the tissues were obtained from MRiLab itself.

$$\text{Signal} = \rho \frac{\sin\alpha \cdot [1 - \exp(-\text{TR}/\text{T}_1)] \cdot \exp(-\text{TE}/\text{T}_2^*)}{1 - \cos\alpha \exp(-\text{TR}/\text{T}_1)}$$

Fig. 2. Equation for Noise Clacuation for Spoiled Gradient Echo Sequence

IV. Results and Discussion

The results of benchmarking the Spoiled Gradient Echo Sequence for MRiLab are mentioned below. During the benchmarking of the tool, we varied only four MRI parameters i.e. flip angle, repetition time (TR), echo time (TE) and hardware noise level. The results are helpful to understand whether the simulator is working correctly or not. By working correctly, we mean to check that the signal achieved from the phantom undergoing MR imaging (simulation) resembles the theoretical signal or not, and if the noise in the image resembles that of a real MR machine.

On Varying TE, at fixed FA = 30° and TR = 500 ms, the simulations worked well for TE values less than 20 ms. But for TE values greater than or equal to 20 ms, the MRiLab showed an error caused due to constants set internally. These constants were used to simply the computation process while solving the Bloch equations. However, this **inability to perform simulations for certain TE values** was not expected and is a limitation of MRiLab.

On Varying TR, at fixed FA = 30° and TE = 5 ms, the simulations follow very different patterns for different noise levels. And more importantly, none of these patterns resemble to that of the theoretical signal values. Fig 3 and Fig 4, shows how the theoretical signal varies with TR, and how the signal computed from image generated by MRiLab varies with TR for Cerebrospinal Fluid (in Fig 3) and White Matter (in Fig 4). We observe no resemblance of MRiLab signal patterns for different noise levels i.e. 0, 5, and 10 with the theoretical signal pattern for both CSF and WM. However, this **inability to resemble theoretical signal while varying TR values** was not expected and is another limitation of MRiLab. Further, it is very strange that at Noise level = 0, we observe no change in Signal while varying TR values for both CSF and WM. Even this **inability of signal to vary with different TR values** is another limitation of MRiLab.

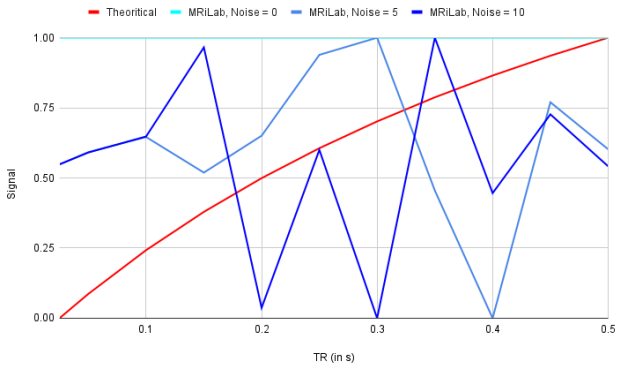

Fig. 3. Theoretical Vs Actual Normalised Signal on varying TR for CSF

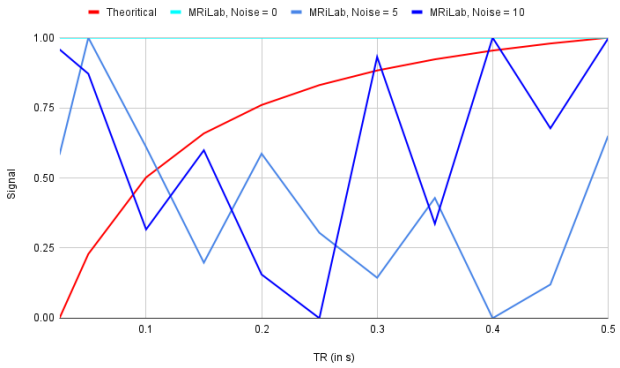

Fig. 4. Theoretical Vs Actual Normalised Signal on varying TR for WM

On varying flip angle, at fixed TE and TR values, the MRiLab simulations showed a monotonous increase in signal strength (brightness) with an increase in flip angle. Fig 5, shows three MR images of brain high-resolution phantom,

where the parameter, flip angle was varied from 30° to 60° to 90° moving in the right direction. The noise level (hardware parameter, taken by MRiLab) was kept at 5, and all the images obtained are converted into grayscale by dividing by the maximum signal value of a phantom location (pixel in the image) for any of the above MR Image. It is to be noted that signal magnitudes for the entire image (brain high-resolution phantom) improves which indicates that whether it is cerebrospinal fluid (CSF), white matter (WM), fat, muscle or any constituent of brain phantom, its signal strength will increase on increasing the flip angle. A very similar increase in signal strength when the experiments were repeated at Noise level 0 and Noise level 10. The

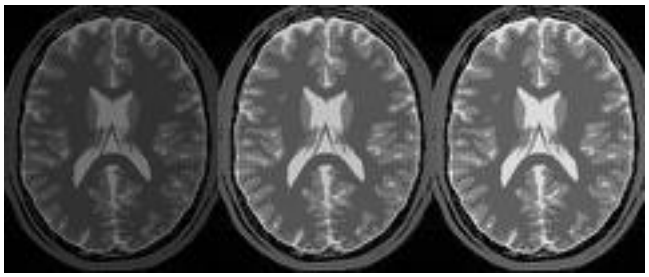

Fig. 5. MR images on increasing flip angle (left to right)

On Varying flip angle for two different TR, we observe no difference in the signal obtained at any noise level. Fig 6 and Fig 7, shows how the theoretical signal varies with flip angle, and how the signal computed from image generated by MRiLab varies with flip angle for Cerebrospinal Fluid (in Fig 6) and White Matter (in Fig 7) at TR = 40 ms and TR = 500 ms. We observe no resemblance of MRiLab signal patterns with the theoretical signal pattern for both CSF and WM. However, this **inability to resemble theoretical signal while varying FA values** was not expected and is another limitation of MRiLab. Further, it is very strange that at two different TR values, we observe absolutely similar signal while varying flip angle values for both CSF and WM, at any noise level. Even this **inability of signal to vary with flip angle for different TR values** is another limitation of MRiLab.

Theoretically, on increasing flip angle we observe an increase in signal, after some time the signal decreases with increasing flip angle. The point of maxima occurs, when the flip angle is equal to the Ernst angle. This is a basic concept in MR imaging, which is violated in the MRiLab simulations, as seen in Fig 5 and Fig 6. We can clearly see the monotonous increase in signal on increasing flip angle in Fig 3. However, this **inability to consider the basic MRI concept maximum signal at Ernst angle while varying flip angle** was not expected and is another limitation of MRiLab.

We also looked into the noise computation model of MRiLab, where we understood that the simulator adds Gaussian noise with zero mean and user-defined standard deviation (referred to as Noise Level) to acquired k-space data. Fig 7, shows the formula used for calculation of noise by MRiLab. It would have been interesting to check the correctness

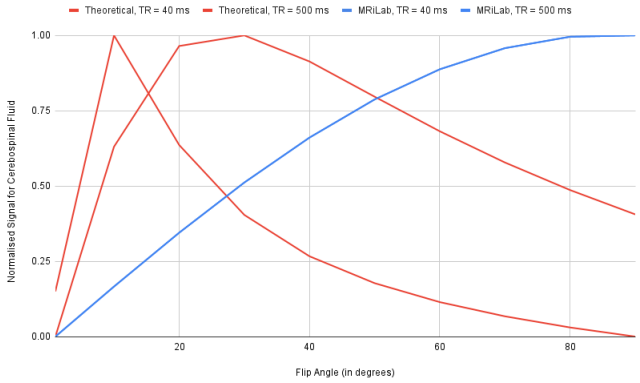

Fig. 6. Theoretical Vs Actual Normalised Signal on varying flip angle for CSF

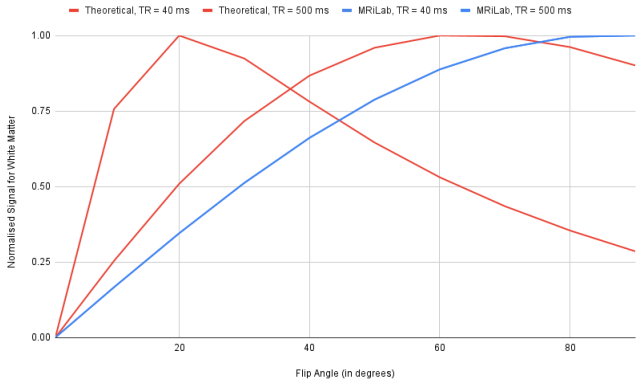

Fig. 7. Theoretical Vs Actual Normalised Signal on varying flip angle for WM

of the added noise, by comparing it to real MR images. However, there are **multiple flaws in calculation of Signal, which inhibits our progress in benchmarking the noise computation model of MRiLab.**

$$Noise = \frac{\frac{NoiseLevel}{NoiseRef} \times \sqrt{\frac{BandWidth}{BWRef}}}{\frac{B0}{B0Ref} \times \frac{RFreq \times RPhase \times RSlice}{VolRef} \times \sqrt{\frac{NEX}{NEXRef}} \times \frac{ResFreq \times ResPhase \times SliceNum}{ADCRef}}$$

Fig. 8. Equation for calculating noise

V. Need for another Simulator

The Experiments performed so far show that MRiLab does not calculate the signal as it should. Focusing mainly on the Spoiled Gradient Echo Sequence, we observed very little resemblance of MRiLab produced images from theoretical variations. MRiLab provides an user-friendly interface with various real-world like functionalities like matter diffusion, multiple MR sequences, etc. However, the results of our experiments put light on some underlying faults.

While going through the implementation of MRiLab software (open-source code), we noticed that the code was written in a non-modular fashion in a low-level language. On delving deeper into where the theoretical blocks of equations

related to MR Simulation were solved, we found that the code used super-simplified forms of differential equations which were derived through an unclear way. The scale of the code spanned over its numerous functionalities which made the error fixing task, a huge and time-taking task.

Therefore, largely due to the complexity, size and non-modularity of the MRiLab code, we agreed upon building our own limited use MR Simulator rather than fixing the flaws of the existing tool. As a part of the new plan, we were made a completely modular MR Simulator written in high level language, Python. For now, our simulator (initial progress) works for the spoiled gradient echo MR sequence and can later be extended to other MR sequences since the code is made modular and by this point has already implemented other MR sequence independent blocks and functions. Essentially, the simulator takes a phantom and finally produce it's MR Image, working with multiple theoretical equations in the back-end.

VI. Detailed Implementation of Our Self-made MR Simulator

A. How our MR simulator works

Fig 9 gives an overview/explains the basic work flow diagram of our MR Simulator.

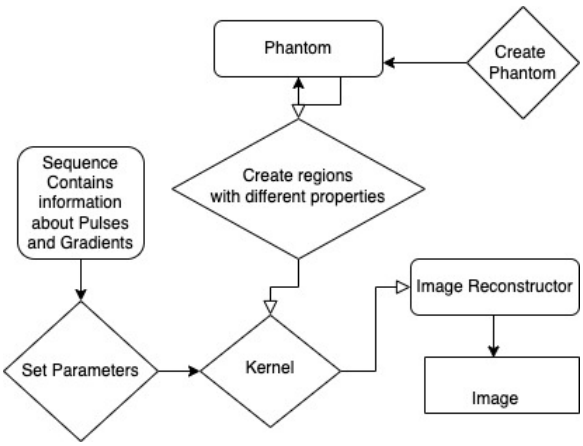

Fig. 9. Flow diagram of our MR Simulator

B. Phantoms

We have functions that can create 3d phantoms that mimic real objects. For simplicity we are starting with cube shaped objects with smaller dimensions. We also have functions that can be used to set values for material intrinsic parameters like T1, T2, proton density for voxels.

A phantom in our implementation can be considered as a python numpy 4-Dimensional array where the first 3 dimensions are to represent the 3D location the voxel and the fourth dimension of this array contains proton density and T1, T2 relaxation times of the material in that voxel.

The phantoms can be customized by adding cubical/spherical regions of different materials, for example, white matter, cerebro-spinal fluid, grey matter, etc.

Fig 10 shows 2 types of phantom that we create for the preliminary analysis of our MR Simulator.

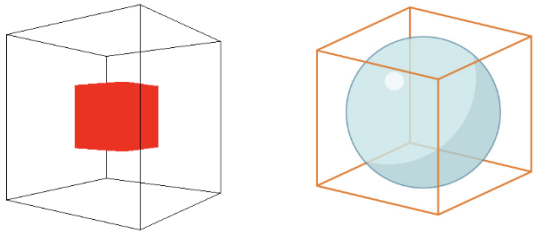

Fig. 10. Phantoms made by us

Fig 11 shows how slice a slice can be selected in such phantom, an MR Image is 2D representation of such a section of the phantom.

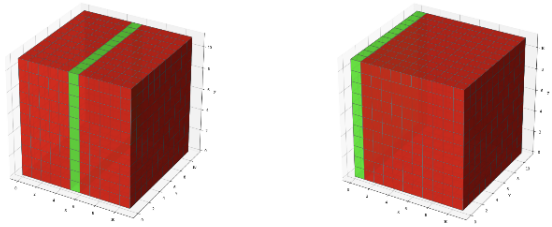

Fig. 11. Slice Selection for final MR Imaging

Another way to build phantom is load a pre-build phantom (say a brain phantom as available in MRiLab) using numpy library.

C. Parameters

For the initial implementation, the MR Simulator works only on the gradient echo sequence. The properties and size of a phantom can be varied, namely the parameters like B0, T1, T2, proton density, etc.

D. Bloch Simulator/Kernel

We have a function that simulation rotation of the magnetization under a magnetic field and the relaxation of transverse and longitudinal magnetization. The Bloch equations are simulated in the rotating frame of reference. We use Bloch equations in their matrix form.

A way to benchmark the working of our simulations till this point would be to study the phenomena of transverse and longitudinal relaxations by using the half-build MR Simulator (whatever was build till this point. Fig12 and Fig13 shows the above-mentioned simulations.

E. Gradient Simulator

A function simulators gradients in various direction to help perform the phase encoding and the frequency encoding steps. The value of the gradients is calculated in the sequence

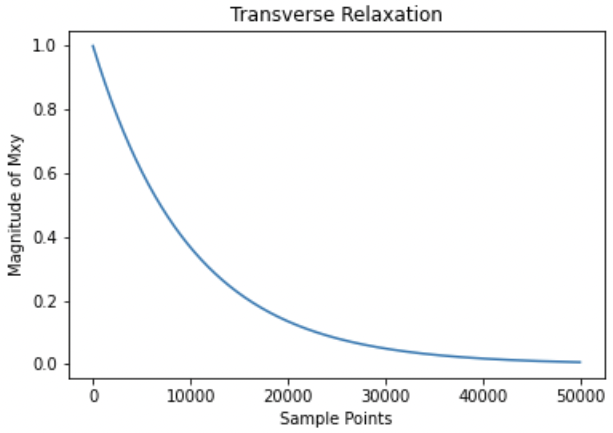

Fig. 12. Simulation of Transverse Relaxation

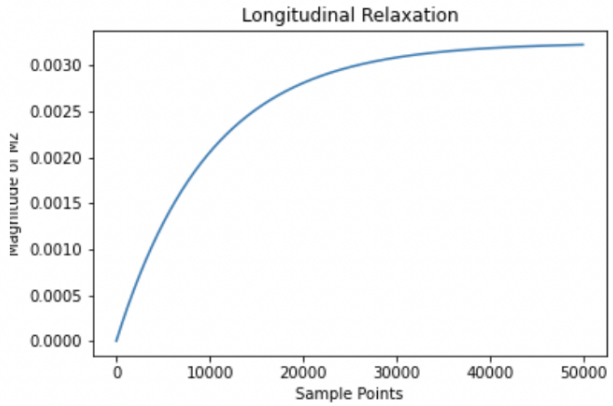

Fig. 13. Simulation of Longitudinal Relaxation

function. The gradients are calculated using equations in the given in Fig 14.

The gradient simulator function can also be used to simulate relaxation in absence of any gradients. In that case the value of the gradient is set to zero.

F. Sequence simulator/K Space Construction

A function that simulates the sequence by calling the gradient simulator function multiple times. After application of the two phase encoding gradients the frequency encoding gradient is applied in steps so that signal acquisition can be done after each step. The gradients for phase encoding keep changing to move across various axis in the K Space.This function populates the K Space.

G. Receiver Coil Induced Voltages And Noise

The K-Space we have till now has 3 axis magnetization in the frequency domain. In a real MRI, there are receiver coils that record the induced voltage corresponding to each voxel. Fig 15 shows the equation used to calculate the induced voltage from magnetization in frequency domain. Each K Space voxel is appplied with this formula during the process.

Resolution and field-of-view maths

The maximum resolution is given by the pixel size

$$\Delta x = \frac{1}{\gamma G_{FE} M \Delta t} \qquad \Delta y = \frac{1}{\gamma \Delta G N \tau}$$

The size of the image or the field of view (FOV) is given by the inverse of the minimum spatial frequency step

$$FOV_{\mathrm{FE}} = \frac{1}{\gamma G_{FE} \Delta t} \qquad FOV_{\mathrm{PE}} = \frac{1}{\gamma \Delta G_0 \tau}$$

Notice that the Fourier transform principle of 'less is more' applies: it is the maximum size of the gradient which controls the pixel size, while the time between samples or phase-encode step size controls the FOV.

Fig. 14. Equations used in Gradient simulation

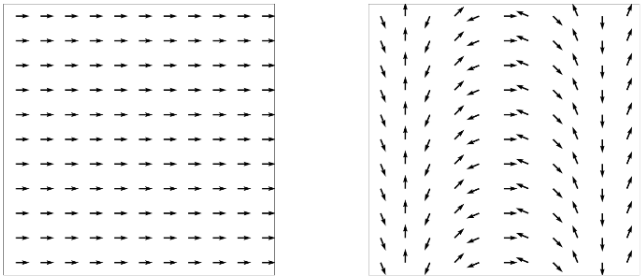

Fig. 15. Magnetic Spins Before and After Applying Gradient

$$\xi = K\omega_0 (B_1)_{xy} M_0 V_s \cos \omega_0 t,$$

Fig. 16. Equation used to calculate induced voltage

A normal-distribution based white noise due to the machine is hereby added to the signal. Fig 16 shows the equation used to calculate standard deviation of this white noise, whereas its mean shall be zero.

$$V = (4kT_c \Delta f R)^{1/2}.$$

Fig. 17. Equation used to calculate standard deviation of white noise

H. Image Construction

To construct an image it is necessary to do an Inverse Fourier Transform for the frequency domain encoding to convert into Spatial domain information (magnitude considered for the complex information generated). We get a 3D image space as output. Then, we select the slice of the 3D

image for which we are interested to see the MR Image. The image we generate are gray-scale image whose values are bounded between 0 and maximum value of the 3D image space, adjusted to 0 to 255.

I. Parallelization

The simulation, especially the simulation of gradients/bloch equations over each voxel over each time step, is highly computationally intensive process. As the first approach, we have utilized pool processing to parallelize the process and reduce the execution time. This simulator can utilize all available CPU cores on a machine. The number of times we run through the gradients is NOP*NOP*NOP (Number Of Pixels) and divided by the number of available processors, we shall get the time complexity of the code. It took around 20 seconds to simulate a 11*11*11 phantom using 2 cores.

J. Overall Program Flow

The overall program flow can be split into two parts. In the first part we consider the creation of a customised phantom. In this part we simulate the initial magnetization, pulse and gradient magnetic fields over the phantom. We then fill the K-Space by summing over the magnetization for the entire sampling time using 2 phase encoding and one frequency encoding parts. This is demonstrated in Fig 18.

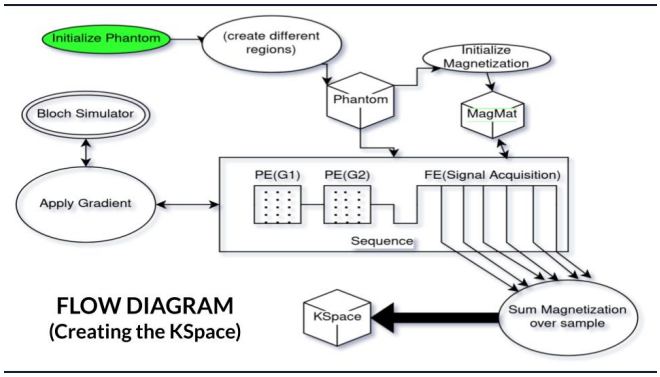

Fig. 18. Program flow - PART 1

In the second part, we consider the conversion of K Space of magnetization into K Space of induced voltages as seen in the receiver coil. This frequency domain value can be thought of as the actual signal, to which we add a white noise. To further create the required 3D image space, we do an 3D Inverse Fourier Transform. At last, we select the slice of interest and plot the values into a gray-scale image. This is demonstrated in Fig 19.

K. Summing up with Demonstrations

Overall, our MR Simulator worked considerably well on the limited scope but had some artifacts while MR Imaging. Fig 20 shows some slice of phantom, its corresponding K Space and MR Image. Visible artifacts can be seen in the case when the phantom is completely filled or filled with more than one voxels in a cube-like fashion around the center.

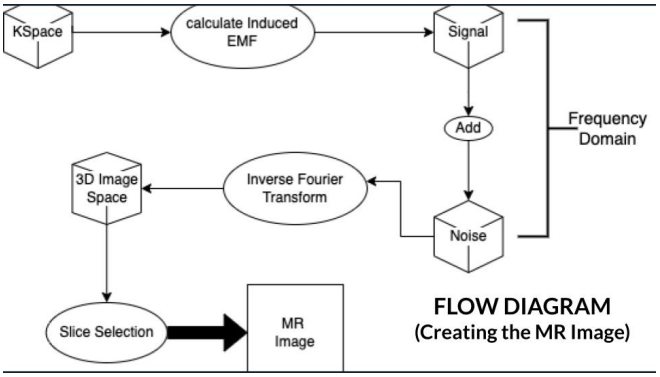

Fig. 19. Program flow - PART 2

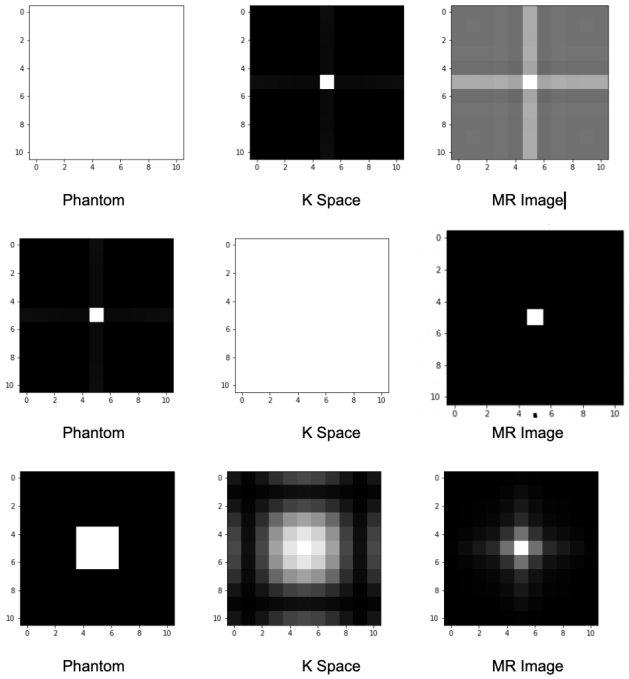

Fig. 20. Demonstration of full simulator

The noise addition worked well being an independent block of the complete modular code. We could expected variations in noise on varying the proton density and initial magnetic field. This confirmed the proper working of the noise simulator (adder).

Fig 21 and 22 show the variation in noise with proton density for a single-voxeled phantom's first and center-most slice respectively. There was an expected decrease with increase in proton density, varied as 1, 50 and 500 in E-10 protons per cubic meter (at fixed initial magnetic field 0.01 Tesla, scale could have error due to some constant offset).

Fig 23 and 24 show the variation in noise with initial magnetic field (B0), for a single-voxeled phantom's first and center-most slice respectively. There was an expected decrease with increase in initial magnetic field, varied as 0.01, 1 Tesla (at fixed proton density 50 E-10 protons per cubic meter, scale could have error due to some constant

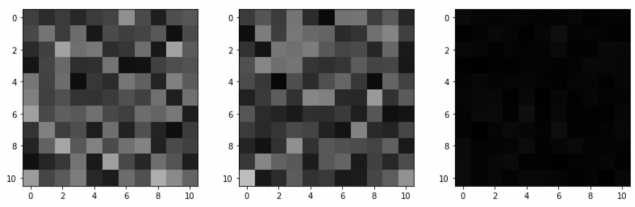

Fig. 21. Demonstration of Noise: Varying in MR Images of single-voxeled phantom's first slice with increase in Proton Density towards right

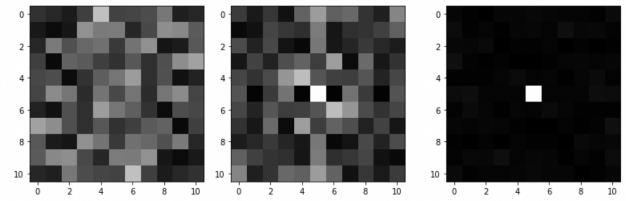

Fig. 22. Demonstration of Noise: Varying in MR Images of single-voxeled phantom's central slice with increase in Proton Density towards right

offset).

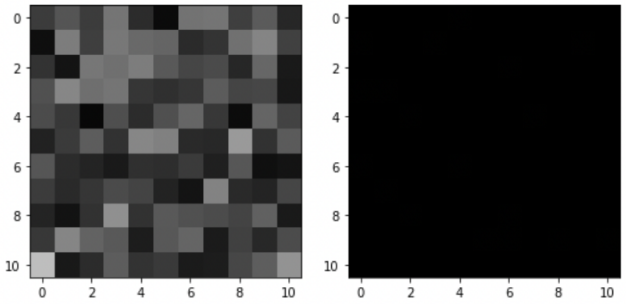

Fig. 23. Demonstration of Noise: Varying in MR Images of single-voxeled phantom's first slice with increase in Initial Magnetic Field towards right

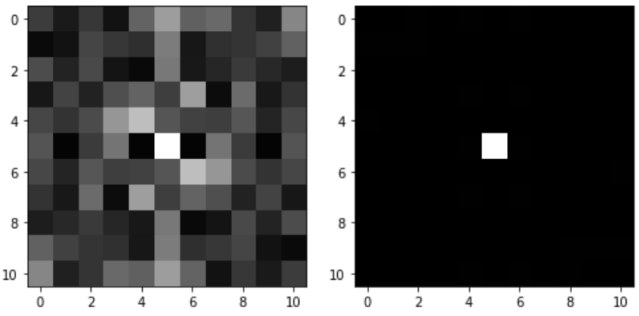

Fig. 24. Demonstration of Noise: Varying in MR Images of single-voxeled phantom's central slice with increase in Initial Magnetic Field towards right

VII. Conclusion

The MRiLab despite being a user-friendly software did not meet the expectations during our benchmarking progress. We see that the signal does not vary as it should. The results from the experiments show that the noise computation model is not accurate. The benchmarking process for the noise computation model would be difficult, considering the multiple flaws in the signal computation. This software is written in a very low level language and is not modular. So adding new functionalities or even changing the kernel (the heart of the simulator) will require a lot of efforts and going through thousands of lines of code.

Therefore, we agreed upon to make a modular basic MRI simulator which can further be developed to simulate real MR more and more closely. At the end of our Major Technical Project AY 2021-22, our MR Simulator could build customised phantoms with spherical/cubical regions of different materials, for example, white matter, cerebro-spinal fluid, grey matter, etc. The Spoiled Gradient Echo MR Sequence shall be applied to this phantom in order to generate a K-space (spatial-frequency information) generated in each voxel of this phantom.

This is generated after simulation of initial magnetic pulse also known as excitation pulse, the gradient magnetic fields on the phantom, relaxation of magnetizations (T1 and T2 relaxations). Going through multiple theoretical equations we calculate the 3D magnetization of each voxel and the induced voltage in the MR receiver coil. Subsequently, noise (as a property of the MR machine) is added, and an appropriate inverse Fourier transform leads to the desired (keeping in mind the FOV) MR image.

The MRiLab took several years of theoretical reading, implementation and debugging to become what it is. What we build, was a product took just 3-4 months and is a simplified version of a software of the scale of MRiLab. Hence, the scope of future work is immense and building a reliable MR Simulator is the need of the hour to create tailored MR datasets. Some of the notable future actions other than scaling to newer MR sequences, adding matter diffusion, etc, for scholars are to parallelize the software and benchmark the tool with real-world phantoms like the brain phantom, or simply a water-fat phantom. In the near future, as a first step a scholar with adequate resources shall also try to run the MR Simulator on big-size phantom, this computation-heavy the task shall provide us an image which shall have lesser artifacts if those were caused due to limitations of the Fourier Transform.

Acknowledgments

We would like to thank our MTP mentor Dr. Erwin Fuhrer and Dr. Arnav Bhavsar for giving us the opportunity to work on this project and constantly allocating their valuable time and efforts for the same. Dr. Erwin helped us a lot while we were trying to grasp a range of information from the basics of MRI to complicated concepts involved in building the MR simulator by provided us apt resources, doubt solving and knowledge transfer sessions. Dr. Arnav guided us to work systematically by pacing our understanding and implementations.

We would also thank the MTP coordinator for SCEE Dr. Varunkumar Jaypaul for their smooth handling of the course and his great vision and efforts in achieving the same. Finally,

all good work require criticism and appreciation to withstand the heat of practicality and proper research methodologies, which would not have been possible for us without our MTP Panel members Dr. A D Dileep and Dr. Aditya Nigam, for which we express our sincere gratitude towards them. We also acknowledge the resources we accessed in the next section.

Fig 25 shows the actual Gantt Chart of progress of the MTP work.

REFERENCES

[1] Liu F, Velikina JV, Block WF, Kijowski R, Samsonov AA. Fast Realistic MRI Simulations Based on Generalized Multi-Pool Exchange Tissue Model. IEEE Trans Med Imaging. 2017;36(2):527-537. doi:10.1109/TMI.2016.2620961

[2] Willemink MJ, Koszek WA, Hardell C, et al. Preparing Medical Imaging Data for Machine Learning. Radiology. 2020;295(1):4-15. doi:10.1148/radiol.2020192224

[3] McRobbie DW, Moore EA, Graves MJ and Prince MR. MRI from Picture to Proton (2003), Cambridge University Press. Retrieved from https://books.google.co.in/books?id=gfuO6NK_InkC.

[4] Hoult DI, Richards RE. The signal-to-noise ratio of the nuclear magnetic resonance experiment. Journal of Magnetic Resonance (1969). 1976 Oct 1;24(1):71-85.

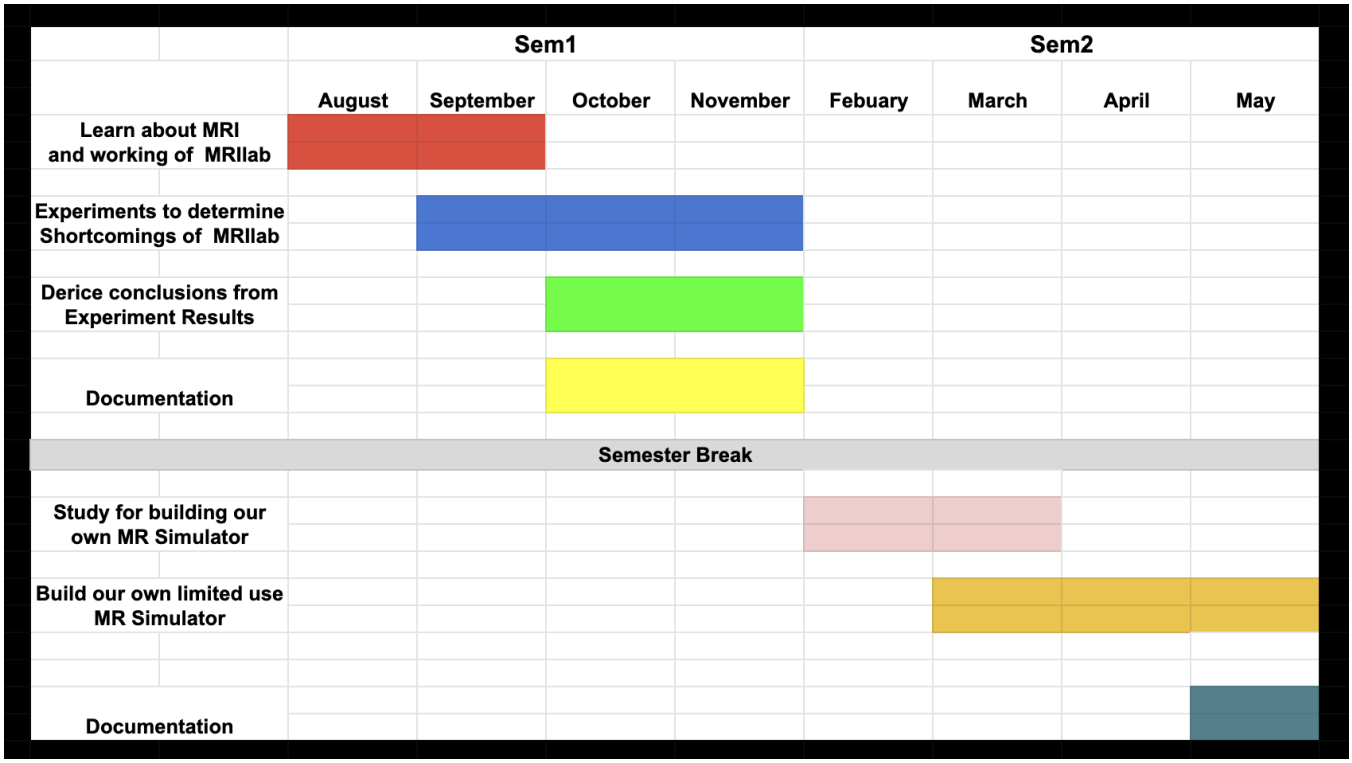

Fig. 25. Gantt Chart