Pulse Sequence Programming with Shared Components: An Open Source Approach

J. F. Magland[1], and W. R. Witschey II[2]

[1]Laboratory for Structural NMR Imaging, Department of Radiology, University of Pennsylvania Medical Center, Philadelphia, PA, United States, [2]Biochemistry & Molecular Biophysics Graduate Group, University of Pennsylvania, Philadelphia, PA, United States

Introduction

Improvements in pulse sequence methodology as well as advances in scanner hardware continue to open new possibilities for magnetic resonance imaging. However, with this innovation comes increased complexity in the task of pulse sequence programming. Researchers must grapple with combining all kinds of complex methods such as parallel imaging schemes, view reordering, contrast weighting (just to name a few), into their imaging protocols. From the number of novel adaptations of pulse sequences presented each year, it is evident that a scanner manufacturer cannot provide anywhere near a comprehensive collection of pulse sequence options to satisfy the current demand, thus requiring researchers to program their own pulse sequences, hire specialized pulse programmers, or to obtain pulse sequence programs from other laboratories or institutions.

Even in the current age of free information exchange, the activity of sharing pulse sequences for MRI is not optimal, due to a number of problems. Besides legal/liability issues, sharing of sequences across institutions is hampered by incompatibility between scanner systems. Even if two institutions use the same brand of scanner, there is often incompatibility between software release versions. Furthermore, as pulse sequences continue to increase in complexity, it is becoming more difficult to modify or customize existing sequences to meet the needs of individual researchers. This leads to a situation where pulse sequences have been designed and implemented and shown to work, but due to technical constraints are not in the hands of the researchers who could benefit, or have not been appropriately modified to function in the desired applications.

Here we present a pulse sequence development paradigm that seeks to solve some of the problems mentioned above by simplifying the pulse program implementation procedure as well as the act of sharing sequences between laboratories and institutions. The strategy relies on the fact that a pulse sequence is simply a collection of basic events to be sent to the scanner (e.g. RF pulses, gradient pulses, phase adjustment and readout events), and should therefore not be specific to any particular brand of scanner. That is, a pulse sequence simply is a scanner-independent program that generates a sequence of timed events. The paradigm has been realized on a small scale in the authors' institution using SequenceTree [1], a scanner-independent, open source, modular pulse sequence programming environment. The large number of research sequences successfully developed and implemented in a clinical setting over the past year using this software demonstrates the efficiency of the method, and the feasibility of a scanner-independent and open source solution.

Methods and Results

Approximately twelve researchers, across three laboratories at the authors' institution have used the SequenceTree (ST) pulse sequence programming environment to implement novel research sequences, sharing code components with one another. As shown in the list of pulse sequences below, the fields of research varied widely. However, since a few basic sequence building blocks are found in virtually all imaging methods, researchers were able to greatly benefit from the shared components. Pulse sequences were built up from scratch using the graphical programming environment of ST with little need for computer programming expertise. The modular and self-contained framework allowed basic sequences, such as gradient-echo or spin-echo sequences (with Cartesian or radial sampling) to be developed in a matter of minutes without any code. More advanced sequences, such as the 3D T1ρ prepared balanced SSFP sequence shown in Figure 1, were also programmed completely within the ST user interface, using custom C++ code incorporated into the sequence nodes (left panel of screenshot in Figure 1). The sequences have been incorporated into at least four research studies, and have contributed to several peer reviewed publications and conference abstracts.

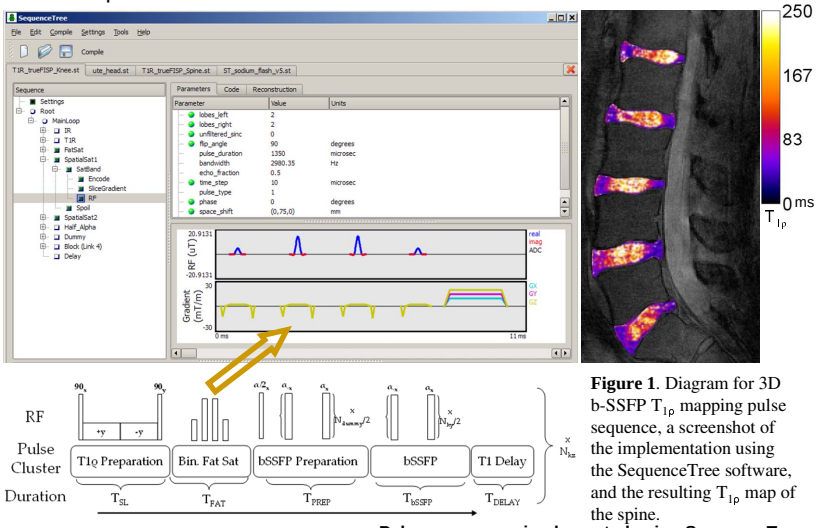

Figure 1. Diagram for 3D b-SSFP $T_{1\rho}$ mapping pulse sequence, a screenshot of the implementation using the SequenceTree software, and the resulting $T_{1\rho}$ map of the spine.

Pulse sequences implemented using SequenceTree

- Ultra-short TE imaging of the knee and head
- T1rho prepared balanced SSFP of the knee and spine
- Fast gradient echo sodium imaging
- Hybrid radial imaging with golden angle increment
- Full 3D radial imaging of the brain
- Trabecular bone micro-MRI with 3D FLASE
- Spectroscopic imaging sequences
- Interleaved multi-gradient echo chemical shift imaging
- Static field mapping using multiple gradient delays
- Susceptibility mapping for blood oxymetry in the leg
- Inversion recovery with adiabatic inversion pulse
- ……

Since ST pulse sequences are scanner-independent (i.e. they contain generic pulse events with no scanner-specific code), the same sequences were able to be implemented seamlessly across different versions of scanner software. For example, sequences could be run without modification on the 1.5T Siemens Sonata and the 3T Siemens TIM Trio. The software does not bypass the scanner safety checks because it generates Siemens-compatible source code which is then compiled and run just like any standard research sequence.

Conclusion

The SequenceTree software has been used efficiently in a shared environment to implement advanced pulse sequences which have run successfully on a clinical scanner, demonstrating the feasibility of an open source scanner-independent solution to pulse sequence programming in a research environment.

References: [1] Magland et al, ISMRM 2006; [2] Jochimsen et al, J Magn Reson, 170(1):67-78 (2004).

Acknowledgement: NIH R01AR053156, NIH F31 EB006299-01A2