CCP in Synergistic PET-MR Reconstruction: User Specification for Software Framework

Version 1.0, 21 September 2015

Synergistic PET-MR Reconstruction

List of contributors

- David Atkinson (UCL)
- Ron Fowler (STFC)
- Julian Matthews (UManch)
- Ross Maxwell (Newcastle)
- Andrew Reader (KCL)
- Kris Thielemans (UCL)
- Martin Turner (STFC)
- Many more for comments and some use cases

Aim of this document

This document is used to first solicit views and then agree on requirements from a user perspective for the core software development by the CCP PET-MR. It will be complemented later by a software design document stating how the software will fulfil those requirements.

General aim of the software framework

Our primary target audience consists of researchers in image reconstruction for PET-MR. We hope that the software will be useful for a wider audience, including researchers in image reconstruction for other (multi)modality systems and researchers in all other aspects of PET-MR data processing.

The software should provide a ***Framework*** for 3D and 4D reconstruction of PET-MR data. It should be ***simple enough*** for education and teaching at post-graduate level, while have ***enough functionality*** for processing of real data in a research context. It will be ***Open Source*** *(using the Apache 2.0 license).*

While the core of the software should provide the functionality described here, it is expected that users will extend (and hopefully) contribute to expansion of the capabilities of the framework. It is therefore essential that the software is ***modular, extendable and robust.*** This implies that an appropriate testing framework is included.

The framework should provide libraries that enable the incorporation of PET and MR reconstruction functionality into other applications. In addition, through these libraries, the functionality with be provided for access from both a ***MATLAB and Python*** interface. Other target languages (e.g. Octave) might be added during the course of the project.

Although we hope that our software will lead to new algorithms/implementations that produce images with higher “image quality” and/or reduced bias/variance, the core CCP software will only provide the tools for researchers to develop such algorithms. The CCP can help with distributing these new algorithms of course.

Supported environments

To simplify the installation procedure, our initial target OS will be Ubuntu 14.04 LTS (which should then work on other Linux systems with a Debian-compatible package management system. Red Hat RPMs should be supported later). MacOS and Windows will be supported later by providing precompiled libraries and executables, as well as source code and instructions. For licensing reasons, MATLAB installation will have to be performed by the user. We will investigate the availability of a cloud-setup where people can log-in and run the Python version on standard data. It is envisaged that in the future there will be need for a cloud-based computational service. Our design will not exclude this option but this will need extra resources from outside this project.

Minimum version requirements

- OS: Linux kernel equivalent to Ubuntu 12.04 LTS, MacOS 10.10 Yosemite, Windows 7
- Compiler: g++ 4.7 (although other C++11 compilers hopefully will work), later Visual Studio 2013 (note: these requirements are tied to MATLAB support)
- CMake 2.8.11
- Python 2.7.5 (but try to maintain compatibility with Python 3), MATLAB 2014a.

These minimum versions might be updated during the 5 year project.

Specific requirements

The sections below contain example code. These lines are not intended as a strict specification but more to give an idea on how we would do this. The syntax used is object oriented and would work with C++, MATLAB and Python (with occasional minor variations between languages).

Error handling

For simplicity, we will not require the user to check return values via error codes etc., but instead throw an error (‘exception’). This can be handled by the user, or will just return to the top level in MATLAB/Python.

File formats

Initial list of supported file formats, but this will be reviewed during the course of the project

- Images: DICOM (allowing access to private DICOM fields), Interfile for PET (as proposed on the STIR website)
- MR raw data: ISMRMRD (current k-space + labels. Future version might include “waveform” data.)
- PET raw data: STIR-compatible projection data, including normalisation files (after conversion to projection data). In the future, list mode data will need to be supported for e.g. motion correction. This will also need ECG info etc. A standard file format for list mode data might therefore be needed.

Other formats might be supported by convertors supplied by the manufacturers. In the first year, we will target to be able to read raw data from the Siemens mMR. We aim to support the GE and Philips PET-MR formats in year 2 (depending on manufacturer support).

High Level User Requirements

- Documentation and scanner-scripts (if necessary) for enabling the saving and exporting of raw MR and list-mode PET from a Siemens mMR Biograph for a single Series. Future additions for other vendors (including GE Signa PET/MR and Philips) and for a whole Study.
- Example datasets.
- Documentation and example script files (MATLAB/Python) for the reconstruction of an exported Series using code that runs on a CCP virtual machine. In MR, initially restricted to scan-types for which there is a convertor to ISMRMRD and an available reconstruction gadget in the Gadgetron.
- Document and example code that demonstrates how to read example DICOM files (reconstructed by the scanner) and largely using existing visualisation software. (This is primarily for developers to be able to check reconstructions, get a feel for data and specifically is NOT the CCP developing a viewer).

- Provide MATLAB/Python “toolbox” for researchers to create/test own reconstruction algorithms on real data.
- Computational performance has to be sufficient for research purposes. We will aim later to add high(er) performance versions as options (e.g. GPU).
- Provide facilities for multi-centre concurrent development via Github/CCPForge.

Image and raw data structures

For images, we will have 2 levels (as in DICOM): **Series** (e.g. a number of 3D volumes for different time frames or gates, together with modality, timing and spatial info, tracer or sequence etc) and **Study** (a number of different series, together with information on what is contained in each series), see https://www.medicalconnections.co.uk/kb/DICOM_Hierarchy for the DICOM hierarchy.

We will not provide visualisation functions as part of the core software, but rely on existing (possibly external) functionality in MATLAB/Python.

IO

Initially we will provide functions to read/write specific data. Later on, we will have “discovery” functions that read data and classify it accordingly.

Basic reconstruction

We will have a reconstruction hierarchy, with different algorithms as “nodes”. Input will be a Study object, parameters can be set via a text file (suggested to be in YAML https://en.wikipedia.org/wiki/YAML), a parameter file supported by the underlying reconstruction engine, or inline.

For PET, we would like to have list mode reconstruction capability. This could however depend on the underlying “reconstruction engine” (i.e. not all engines will support this).

Advanced reconstruction using objective functions

Some optimisation algorithms only need function value and gradient, e.g. in MATLAB this is ok for many functions in the Optimization Toolbox. We need an interface to this.

Access to underlying forward model etc

Some users will want to be able to use projectors, noise models, etc.

For PET, we need to be able to generate (from ‘raw’ data, or after conversion by the manufacturer) a normalisation sinogram (including dead-time), a scatter sinogram, and have access to forward and back-projector suitable for the geometry of the supported PET-MR scanners. Projectors will have to be able to work on a subset of the data.

For MR, the forward modelused depends on the level of complexity required. It may incorporate, spatially varying static fields (B0 effects), time variations of these fields, the coil sensitivities, noise, small timing delay differences and the k-space sampling pattern.

Example Use Cases

- Read in reconstructed images, understand geometry and timing, reinterpolate to different sampling, display, relying on existing tools in MATLAB/Python
- Read in raw PET and MR data and do basic QA and display
- Reconstruct undersampled multi-coil MR T1, i.e. parallel imaging (framework needs to provide this). Will aim for non-cartesian (e.g. radial) sampling later.

- Incorporation of TOF-based PET reconstruction and other use of this time resolution in joint data analysis.
    - Needs standardisation of TOF data
    - Needs TOF projectors, scatter etc
- Attenuation correction of PET data with control over which MR data is used.
    - Needs alignment of PET and MR data
    - Needs metadata information in sequence type.
    - Might need incorporation of motion correction information (e.g. external monitor, or info on any gating used for MR sequence).
- Reconstruct PET data with MR image as anatomical prior
    - Might need understanding deformations in MR (depends on sequence)
- Use numerical optimisation library as reconstruction tool
- Raw MR data processed to detect "glitches" (e.g. sudden movement), timings of "glitches" fed back to PET recon (e.g. high motion detected from MR)
    - Needs access to raw MR and timing data. No MR recon required.
    - Time sync with PET needs to have been determined, e.g. via MR time stamps in PET list mode data.
    - We need to provide metadata about clock info.
- Processing of MR navigators
    - Extraction of navigator readouts from MR raw data.
- Correct head motion during PET given real-time motion information, frame-wise or event-wise.
- Motion information from PET and MR used to correct MR raw data (to improve sharpness)
    - Needs a standard to describe coordinate system, directions etc, standard for deformation fields
    - Requires timing information.
    - Need to modify a conjugate gradient MR recon to accept motion and deal with the Hermitian transpose operation.
- 4D reconstruction:
    - Simple example: 4D recon of radial MR data using sliding windows (i.e. decide how many "spokes" to use).
    - Direct reconstruction of DCE-MRI parametric maps (with integrated motion correction and tracer-kinetic modelling).
    - include joint modelling of MR contrast agent and PET tracer kinetics. Could eventually develop to full pharmacokinetic-pharmacodynamic modelling e.g. drug kinetics from uptake of PET tracer and effects (pharmacodynamics) from fMRI.
- MR Fingerprinting (advanced topic!)
    - Uses "difficult" MR sequences with strange k-space trajectory.
    - Needs models for specific MR sequences.
    - will do dictionary-based fits (might need simulation based on tissue types)
- Provision of data suitable for radiotherapy planning (advanced topic!)
    - This is likely to require confidence on absolute spatial locations and quantitative image data (e.g. equivalent to Hounsfield units).
    - Requirements would need to be defined by experts in that domain. (Example: handling MR spatial distortions)