

SIRF: Synergistic Image Reconstruction Framework

Evgueni Ovtchinnikov, David Atkinson, Christoph Kolbitsch, Benjamin A. Thomas, Ottavia Bertolli, *Student Member, IEEE*, Casper O. da Costa-Luis, *Student Member, IEEE*, Nikolaos Efthimiou, Ronald Fowler, Edoardo Pasca, Palak Wadhwa, *Student Member, IEEE*, Elise Emond, *Student Member, IEEE*, Julian Matthews, *Member, IEEE*, Claudia Prieto, Andrew J. Reader, Charalampos Tsoumpas, *Senior Member, IEEE*, Martin Turner, Kris Thielemans, *Senior Member, IEEE*
On behalf of *CCP PETMR*

Abstract— The combination of positron emission tomography (PET) with magnetic resonance (MR) imaging opens the way to more accurate diagnosis and improved patient management. At present, the data acquired by PET and MR scanners are essentially processed separately, and the search for ways to improve accuracy of the tomographic reconstruction via synergy of the two imaging techniques is an active area of research.

The aim of the collaborative computational project on PET and MR (CCP-PETMR), supported by the UK engineering and physical sciences research council (EPSRC), is to accelerate research in synergistic PET-MR image reconstruction by providing an open access software platform for efficient implementation and validation of novel reconstruction algorithms.

Manuscript received November 17, 2017. This work was supported by UK EPSRC under Grant EP/M022587/1.

E. Ovtchinnikov is with the Visual Analytics and Imaging Systems Group, Rutherford-Appleton Laboratory, Science and Technology Facilities Council, UK (telephone: +44 1235 446668, e-mail: evgueni.ovtchinnikov@stfc.ac.uk).

D. Atkinson is with the Centre for Medical Imaging, University College London, UK (telephone: +44 20 3549 5660, e-mail: d.atkinson@ucl.ac.uk).

C. Kolbitsch is with the Physikalisch-Technische Bundesanstalt (PTB), Braunschweig and Berlin, Germany (e-mail: christoph.kolbitsch@ptb.de).

B. A. Thomas is with the Institute of Nuclear Medicine, University College London, UK (e-mail: b.a.thomas@ucl.ac.uk).

O. Bertolli is with the Institute of Nuclear Medicine, University College London, UK (e-mail: ottavia.bertolli.13@ucl.ac.uk).

C. da Costa-Luis is with the School of Biomedical Engineering & Imaging Sciences, King's College London, UK (e-mail: casper.dcl@kcl.ac.uk).

N. Efthimiou is with the Institute of Nuclear Medicine, University College London and School of Biological, Biomedical and Environmental Sciences, University of Hull, UK (e-mail: N.Efthymiou@hull.ac.uk).

R. Fowler is with the Visual Analytics and Imaging Systems Group, Rutherford-Appleton Laboratory, Science and Technology Facilities Council, UK (telephone: +44 1235 445243, e-mail: ronald.fowler@stfc.ac.uk).

E. Pasca is with the Visual Analytics and Imaging Systems Group, Rutherford-Appleton Laboratory, Science and Technology Facilities Council, UK (telephone: +44 1235 445660, e-mail: edoardo.pasca@stfc.ac.uk).

P. Wadhwa is with the Division of Biomedical Imaging, University of Leeds and Imanova Ltd, UK (e-mail: p.wadhwa@leeds.ac.uk).

E. Emond is with the Institute of Nuclear Medicine, University College London, UK (e-mail: elise.emond.16@ucl.ac.uk).

J. Matthews is with the Division of Informatics, Imaging and Data Sciences, MAHSC University of Manchester, UK (telephone: +44 161 275 0024, e-mail: Julian.Matthews@manchester.ac.uk).

C. Prieto is with the Department of Biomedical Engineering, King's College London, UK (e-mail: claudia.prieto@kcl.ac.uk).

A. J. Reader is with the School of Biomedical Engineering & Imaging Sciences, King's College London, UK (e-mail: andrew.reader@kcl.ac.uk).

C. Tsoumpas is with the Division of Biomedical Imaging, University of Leeds and Imanova Ltd, UK (telephone: +44 113 343 8312, e-mail: c.tsoumpas@leeds.ac.uk).

M. Turner is with the Visual Analytics and Imaging Systems Group, Rutherford-Appleton Laboratory, Science and Technology Facilities Council, UK (e-mail: martin.turner@stfc.ac.uk).

K. Thielemans (corresponding author) is with the Institute of Nuclear Medicine, University College London, UK (e-mail: k.thielemans@ucl.ac.uk).

In this paper, we present the first release of the Synergistic Image Reconstruction Framework (SIRF) software suite for the CCP-PETMR. SIRF provides user-friendly Python and MATLAB interfaces to advanced PET and MR reconstruction packages written in C++ (currently this uses STIR, Software for Tomographic Image Reconstruction, for PET and Gadgetron for MR, but SIRF will be able to link to other reconstruction packages in the future as appropriate). The software is capable of reconstructing images from real scanner data. Both available integrated clinical PET-MR systems (Siemens and GE) are being targeted, and a suitable data format exchange is being negotiated with the manufacturers.

Index Terms— Positron Emission Tomography, Magnetic Resonance Imaging, Research Software Engineering, Software Programming

I. INTRODUCTION

MR images usually provide excellent anatomic detail and high soft tissue contrast of the investigated area, but lack sensitivity for imaging probe detection compared to PET images. The latter in turn can clearly show regions of abnormality thanks to the distribution of radiotracer throughout the body, whereas normally functioning areas may not be seen. Recognition of the complementarity of the two imaging modalities prompted the development of integrated PET-MR scanners, which as yet has not been paralleled by the development of an integrated software platform for processing data acquired by such scanners.

CCP-PETMR (<https://www.ccppetmr.ac.uk>) is a not-for-profit organisation established in April 2015 with the aim of facilitating the investigation of novel synergistic PET-MR reconstruction methods, as investigated by various authors [1][2], by providing the PET-MR research community with an open access software development platform that would be simple enough to use for research and education purposes and, at the same time, powerful enough to be able to handle (in real time) raw medical data acquired by PET, MR and PET-MR scanners. Both of the available integrated clinical PET-MR systems are being targeted (Siemens and GE) in this software platform, and a suitable data format exchange is being negotiated with the manufacturers.

This paper describes the first release of our synergistic image reconstruction framework (SIRF), an open access software platform.

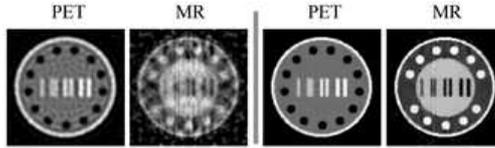


Fig. 1. Example reconstruction from simulated PET-MR data (Poisson noise for PET, Gaussian noise for MR, radial sampling with 15 lines). Left: independent reconstructions. Right: joint-reconstructions using Quadratic Parallel Level sets prior (after fig 5 in [1]).

II. SOFTWARE DEVELOPMENT STRATEGY

We strive to make our platform useful for as wide a research community as possible. An important choice is the programming language that the users of our platform will need to adopt for their own software development. Two languages and environments that are widely used by researchers with limited software development experience, are MATLAB and Python, basic knowledge of at least one of which we expect our users to have. To handle the conflict between the simplicity of development and the efficiency of the developed software, we use advanced programming languages such as C++ for the actual development and provide simple Python and MATLAB interfaces to building blocks that the users of our platform may use in their own software development. For ease of interfacing into MATLAB, Python and other prospective languages, we wrap C++ code into a C interface.

SIRF is the product of a joint effort of a number of developers from various UK and European Universities. To manage SIRF development efficiently, we use GitHub with its issue tracking, milestones and release management facilities (visit <https://github.com/CCPPETMR/SIRF>). We also employ the Travis Continuous Integration system: every GitHub commit is compiled and tested automatically, the build/test status is reported in an e-mail and can be checked online at any time.

III. CURRENT SOFTWARE STRUCTURE

SIRF is built upon existing open source software. At present, we employ two well-known libraries: STIR (software for tomographic image reconstruction) [3] for PET and Gadgetron [4] for MR, and we refer to particular reconstruction software packages as 'engines'. Current software structure is reflected by the SIRF file structure, as follows:

```

SIRF      : root
data      : raw data
examples  : Matlab and Python demo scripts
src       : sources
common    : utilities common to all engines
iutilities : C-interface utilities
xgadgetron : Gadgetron extensions
cgadgetron : C/C++ interface to Gadgetron
mgadgetron : Matlab interface to cgadgetron
pgadgetron : Python interface to cgadgetron
xstir     : STIR extensions
cstir     : C/C++ interface to STIR
mstir     : Matlab interface to STIR
pstir     : Python interface to STIR

```

IV. USAGE

SIRF follows object-oriented principles, which means most functions callable by a user's code are member functions (or methods) of a platform's objects.

As an example, a Python script using ordered statistics maximum *a posteriori* one step late (OSMAPOS) algorithm for PET reconstruction would involve the following (among others, not shown for brevity):

```

# create reconstructor object
recon = OSMAPOSReconstruction()
# set reconstructor parameters
recon.set_num_subsets(num_subsets)
# prepare reconstructor for use
recon.set_up(image)
# perform reconstruction
recon.reconstruct(image)

```

The above exemplifies the simplest level of user's involvement in PET reconstruction. While it is the most performant and efficient way of using the STIR library for reconstruction, some users may prefer to have more control over computation. For example, given an existing iterative algorithm, the user may wish to run a loop containing the following lines:

```

# perform one iteration
recon.update_current_estimate()
# get current image as Python array
img_arr=recon.get_current_estimate().as_array()
# process image using own processor
img_arr=my_image_processor(img_arr)
# make processed image a new image estimate
recon.set_current_estimate(image.fill(img_arr))

```

On a still higher level of involvement, the user may employ their own or a third-party optimization algorithm. For example, in an instance, a simple gradient ascent could use a loop:

```

# compute the gradient of objective function
grad=obj_fun.get_subset_gradient(image,subset)
# get gradient as Python array
g=grad.as_array()
# get current image as array
x=image.as_array()
# define a line search function
f=lambda t: -obj_fun.get_value(image.fill(x+t*g))
# maximize f
t=scipy.optimize.fminbound(f, 0, maxstep)

```

Similarly, for MR reconstruction, one can use the following script that requires no user's input (and no knowledge of the reconstruction engine):

```

# get access to input data
input_data=AcquisitionData(input_file)
# remove oversampling
processed_data=preprocess_acquisitions(input_data)
# create reconstructor
recon=SimpleReconstruction()
# reconstruct
image=recon.reconstruct(my_processed_data)

```

A user familiar with Gadgetron's chains is likely to use an efficient, explicit two-gadget chain instead:

```

input_data=AcquisitionData(input_file)
# create a chain to remove oversampling and reconstruct
recon=Reconstructor(['RemoveROoversamplingGadget','SimpleReconstruction'])
# do all processing
image=recon.reconstruct(input_data)

```

Again, while this is the most performance-efficient way of using Gadgetron, some users may prefer to try their own acquisition processor in addition to the removal of oversampling, in which case the above chain can be split into two, with the user's processing in between:

```

input_data=AcquisitionData(input_file)

```

```

# create a chain that removes oversampling
ap=AcquisitionProcessor(['RemoverOOversamplingGadget'])
# remove oversampling
processed_data = ap.process(input_data)
# apply your own data processing
my_processed_data = my_processor(processed_data)
# create a reconstruction chain
recon = Reconstructor(['SimpleReconGadgetSet'])
# reconstruct
image = recon.reconstruct(my_processed_data)

```

V. INSTALLATION

SIRF is available (under the Apache 2.0 license) at <https://github.com/CCPPETMR> and the user has the following options for obtaining and installing SIRF (see the SIRF Wiki for detailed instructions):

1. Download a Ubuntu Virtual Machine with all the necessary software (except MATLAB) preinstalled from <https://www.ccppetmr.ac.uk/downloads>.
2. Linux/MacOS: Download the sources of SIRF, STIR and Gadgetron and build using instructions on the [SIRF Wiki](#).
3. Windows: as above but use Gadgetron installed on VM.
4. Linux/MacOS: Use SIRF-SuperBuild project, based on CMake. The major benefit of this approach is that specific versions of the required dependencies are built (to ensure consistency) without user intervention.

VI. EXAMPLE RESULTS

In this section, we show some preliminary results of reconstructed data of clinical scanners. Support for PET and MR data is mostly complete for the Siemens mMR, while for the GE Signa, only the PET data are handled. PET-TOF support is currently under development [5][6].

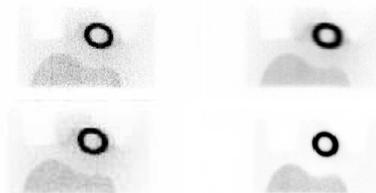


Fig 2. Reconstructed images from an anthropomorphic torso phantom experiment for GE SIGNA PET/MR scanner with attenuation (conducted with MR-based Dixon segmentation) and normalisation correction, and without scatter correction using STIR: (top left) OSEM, 28 subsets and 3 iterations, (top right) OSMAPOSL with Quadratic Prior, penalty factor 10, (bottom left) OSSPS-PSF, (bottom right) ToF-OSEM acquired from console.

Fig 3. Example images of cardiac patient data acquired on Siemens mMR reconstructed with SIRF. Data from [7].

VII. CONCLUSION AND FUTURE WORK

This software suite will offer a simpler tool for established researchers to actively and quickly participate in the research of PET and MR image reconstruction and members of a larger consortium. This should lead to efficient and robust translation of novel ideas into clinical practice.

ACKNOWLEDGMENT

We thank GE Healthcare and Siemens Healthineers for their support and for the supplied information.

REFERENCES

- [1] Ehrhardt M, Thielemans K et al. "Joint reconstruction of PET-MR exploiting structural similarity", *Inverse Problems*, Vol 31, No 1, 2015.
- [2] F. Knoll, M. Holler, et al., "Joint MR-PET reconstruction using channel image regularizer," *IEEE Transactions on Medical Imaging*, p. 1, 2016.
- [3] Thielemans K, Tsoumpas C et al., "STIR: Software for Tomographic Image Reconstruction Release 2", *Physics in Medicine and Biology* (4), 2012 pp.867-883
- [4] Hansen MS, Sørensen TS. Gadgetron: "An Open Source Framework for Medical Image Reconstruction". *Magn Reson Med*. Jun;69(6):1768-76.
- [5] Efthymiou N, Thielemans K, Tsoumpas H, "Initial Validation of Time-Of-Flight List-Mode MLEM and OSEM Reconstruction Algorithms using the STIR Framework Using Monte Carlo Simulated Data", *IEEE Transactions on Medical Imaging*, 2016
- [6] N. Efthymiou, E. C. Emond, C. Cawthorne, C. Tsoumpas, Thielemans, Reconstruction of Time-Of-Flight Projection Data using the STIR reconstruction framework, *Proc. IEEE NSSMIC 2017*
- [7] Kolbitsch C, Neji R et al., "Fully integrated 3D High-Resolution Multicontrast Abdominal PET-MR with High Scan Efficiency", *Res. Med.* 2017 May 14. doi: 10.1002/mrm.26757.