

# J-PET Framework: Software platform for PET tomography data reconstruction and analysis

W. Krzemien

*High Energy Physics Division, National Centre for Nuclear Research,  
05-400 Otwock-Świerk, Poland  
wojciech.krzemien@ncbj.gov.pl*

A. Gajos, K. Kacprzak, K. Rakoczy, G. Korcyl

*Faculty of Physics, Astronomy and Applied Computer Science  
Jagiellonian University  
S. Łojasiewicza 11, 30-348 Kraków, Poland*

---

## Abstract

J-PET Framework is an open-source software platform for data analysis, written in C++ and based on the ROOT package. It provides a common environment for implementation of reconstruction, calibration and filtering procedures, as well as for user-level analyses of Positron Emission Tomography data. The library contains a set of building blocks that can be combined by users with even little programming experience, into chains of processing tasks through a convenient, simple and well-documented API. The generic input-output interface allows processing the data from various sources: low-level data from the tomography acquisition system or from diagnostic setups such as digital oscilloscopes, as well as high-level tomography structures e.g. sinograms or a list of lines-of-response. Moreover, the environment can be interfaced with Monte Carlo simulation packages such as GEANT and GATE, which are commonly used in the medical scientific community.

*Keywords:* PET, data processing, software analysis framework

---

## Code metadata

---

Current code version	8.1
Permanent link to the repository	[1]
Legal Code License	Apache License 2.0
Code versioning system used	git
Software code languages, tools and services used	C++, Python, ROOT, Boost
Compilation requirements & dependencies	Linux; g++ compiler supporting the c++14 standard; CMake 3.1.5 or later; Boost libraries 1.64 or later, ROOT version 6.X
Developer documentation/manual	[2]
User and developer support, bug tracker	[3]

---

## 1. Motivation and significance

J-PET Framework is an open-source C++ library for the analysis of Positron Emission Tomography (PET) data providing an environment for implementation of reconstruction, calibration and filtering algorithms, as well as for user-level data analyses.

Positron Emission Tomography is one of the most popular methods for tomographic imaging used in nuclear medicine. In contrast to other techniques such as Magnetic Resonance Imaging or Computed

Tomography that can mostly detect structural changes, PET provides information about metabolic processes in the patient's body even at the cell level. This allows detection of pathological symptoms that usually precede the anatomical changes. PET tomography has a wide range of research and clinical applications e.g. it is commonly used for diagnosis of cancer, neurological disorders, heart diseases and many others.

Although the PET technique is well established for clinical usage, there are ongoing efforts in the scientific community to develop novel modalities that would overcome the limits of the commercial scanners and improve the quality of the image or even enrich the available information by introducing new diagnostic methods. Whole-body or total-body PET scanner projects [4, 5, 6] propose tomographs that improve the sensitivity of the measurement in order to shorten the time of a scan or alternatively require a smaller radiation exposure for the patients [7].

The transformation of the data acquired by a PET scanner from the *raw* binary level till the final patient image analysed by physicians is a complex, multi-stage process involving low- and high-level reconstruction algorithms. The associated data handling and reconstruction is an especially hard task in case of the whole-body scanners due to large data volume [8].

The J-PET collaboration aims at providing a low-cost, modular, whole-body PET scanner based on detection of photon interactions in plastic scintillators [9, 10, 11] with a view to its application in both medical diagnostics [11, 12] and in proton therapy monitoring [13]. The J-PET prototype is a research device which not only demonstrates the new operating principle for its use in standard PET tomography but also explores new imaging modalities such as spatially-resolved determination of properties of positronium atoms produced in a patient's body [14, 15, 16, 17].

The exploratory nature of the J-PET device results in its operation with much more flexible data registration conditions than used in commercial PET solutions. In order to allow for classical PET imaging without discrimination of signals, which may be used in the novel diagnostic methods, J-PET operates in a trigger-less data acquisition mode [18], resulting in a volume of recorded data unprecedented in medical imaging technologies.

From the software point of view, development and testing of novel PET modalities and tomography methods become challenging as the standard approaches must be either extended or entirely replaced by new algorithms. Moreover, at the prototyping stage, multiple elements of the detector, its geometrical setup and the data acquisition chain are subject to change and various reconstruction procedures may be tested in parallel. The software framework used to analyze data from evolving prototypes and to implement and test new reconstruction algorithms must follow these changes dynamically. At the same time, however, the need to efficiently process the data stream from trigger-less acquisition requires that the performance may not be compromised when asserting flexibility.

The J-PET Framework package has been developed as an answer to the aforementioned challenges, providing a dynamically adjustable environment for development and efficient implementation of new algorithms. The basic idea is to provide a set of generic building blocks, allowing a quick implementation of data processing chains to be used by analysts with even little programming experience through a convenient and simple API. The J-PET Framework is used for analysis of data recorded by the tomograph prototype from the level of raw data saved by its data acquisition system, through assembly of higher-level data structures representing the logic needed for reconstruction of the physical properties of electron-positron annihilation into photons, up to the level of medical image reconstruction and statistical analysis of the data.

Notably, the data acquisition system of J-PET is based on the TRB3 hardware platform [19, 20] which is widely used by experimental setups both in the fields of medical imaging and particle physics experiments [21]. Consequently, the J-PET Framework can be easily adopted for data analysis in other TRB3-based experiments.

While an early version of the J-PET Framework is described in Ref. [22], this article is intended to present its architecture and functionality available in its current mature form, which allows to extend the scope of its usage beyond the J-PET project. Therefore, we focus on the properties of the core J-PET Framework library [1] rather than on the particular J-PET-specific reconstruction algorithms developed using the framework which are available in a separate repository [23].

## 2. Software description

The design of the J-PET Framework originated from the necessity of performing reconstruction and analysis of PET data from a prototype tomography scanner. It has become a more generalized environment for execution of tasks which could be adapted to multi-step analysis of various kinds of data. All

the features naturally followed the implementation: managing input/output, incorporating palette of configurations, adapting data and parameter structures, user interface and task handling.

The core of the J-PET Framework is constituted by a dynamic library that can be linked to user applications. The library provides tools for loading, analysing and saving transformed data as well as for implementation of transformation algorithms that can be further connected in chains and finally executed.

## 2.1. Software Architecture

The library is written in C++ using object-oriented paradigm. The core components are implemented as classes with well-defined responsibilities e.g. computing task execution, input/output operations, logging, option parsing, option validation. Moreover, the package contains a set of classes representing physical entities e.g. part of the scanner or PET-specialized data structures such Line-of-Response (LOR), which form a language that can be used to express the domain-specific concepts (see more details in section 2.2.4).

The basic concept of the J-PET Framework is the decomposition of a data processing chain into a series of standardized modular blocks. Each module corresponds to a particular computing task, e.g. a reconstruction algorithm or a calibration procedure, with well-defined input and output. The processing chain is built by registration of chosen modules in the `JPetManager`, responsible for synchronization of the data flow between the modules (see Figure 2.1). This approach permits to quickly interchange modules and to create processing chains for different experimental setups.

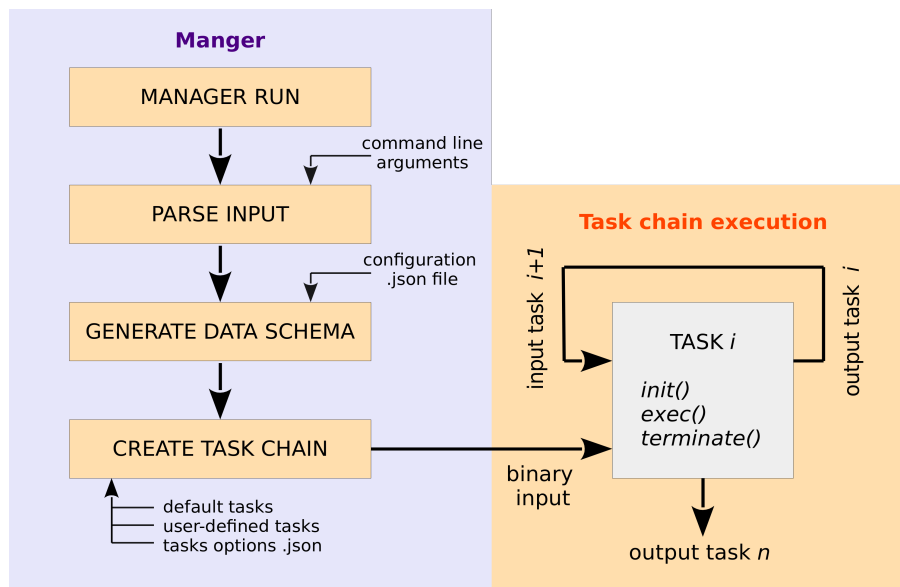


Figure 1: Scheme of Frameworks `JPetManager` structure, showing order of initialization and execution of tasks.

## 2.2. Software Functionalities

J-PET Framework provides a set of functionalities that helps in rapid data reconstruction and analysis prototyping. In this section, we list the most useful features and present selected usage examples. More applications can be found in the repository [23].

### 2.2.1. Handling of multiple data sources

The Framework provides a generic input-output mechanism that through simple extensions allow for processing of data from various sources e.g. low-level data in a binary format from a tomographic data acquisition system, textual representations of complete photomultiplier (PMT) signal waveforms collected using a serial data analyzer at detector testing stages, as well as high-level tomography-specific structures e.g. sinograms or lines-of-response. Other extensions of the input interface feature using results of Monte Carlo simulations in place of data as described in section 2.2.10.

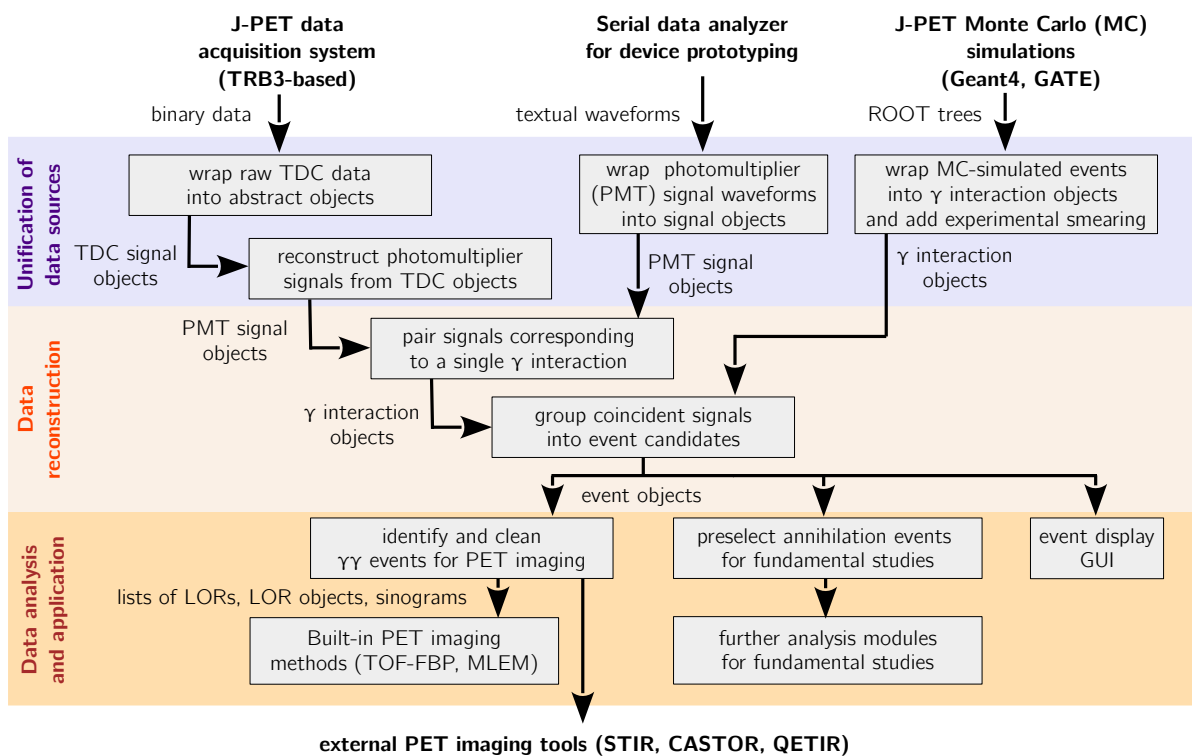


Figure 2: Scheme of the data processing paths realized with the J-PET Framework for different cases of analysis of data from the J-PET prototypes and the corresponding Monte Carlo simulations. Each gray rectangle represents a single module whereas arrows denote the flow of data represented as abstract objects.

Figure 2 presents how inputs from various data sources are unified at higher analysis levels so that more abstract steps of reconstruction can act on data independently of their origin. PMT signals recorded with a serial data analyzer, for example, correspond to PMT signal representations already assembled from single Time-to-Digital Converter (TDC) signals in case of the TRB3-based data acquisition system and are thus injected to the analysis chain at the corresponding level, i.e. before a module pairing PMT signals from the same detection module to identify photon interactions.

### 2.2.2. Input/Output mechanisms

Besides the source-specific data formats handled by dedicated wrapper modules, the J-PET Framework relies on binary internally-compressed data format provided by the ROOT package, widely adopted in both particle physics and nuclear medicine research. The framework provides automatic handling of input and output files for standard data analysis modules, abstracting the actual storage away from the analysis or reconstruction logic. User code is only responsible for deciding whether an entry processed by a module should be preserved or discarded. Depending on the option chosen by the user, output from every analysis module is either saved to a separate file in the ROOT format or directly fed as input to the subsequent module in the chain. While the former is useful at the stages of testing the analysis, the latter approach allows to create a pipeline of analysis modules minimizing I/O load as the only output saved to disk is the one produced by the last module in the chain which typically corresponds to the most filtered data stream where the data volume is reduced by 1-2 orders of magnitude with respect to the raw input. This is particularly important when multiple analysis processes are operating on the same disk space, which is a common use case in data-driven parallel computing specific to particle physics and low-level PET tomography event filtering and reconstruction.

### 2.2.3. Options

The library provides multiple manners of loading optional information for any custom processing task. These collections of various parameters would be required for a successful reconstruction of PET data, giving i.e. descriptions of a experimental setup, measurement conditions, necessary calibrations or desired form of the output. The Framework provides the following interfaces for dynamic configuration:

- command line options (e.g. input file, configuration files, progress display)
- JSON file with the description of experimental setup - parametrization of objects, that serve as data schema,
- JSON file with user-provided options - any custom settings to be used during execution of tasks, passed as named parameters of elementary C++ types

All the provided options are parsed and validated before execution of chain of tasks and are accessible during its processing.

### 2.2.4. Data and parameter structures

The library includes classes representing abstract entities common for analysis of data from J-PET measurements. Parameter Objects represent hardware parts of the detector, along with their working parameters, in-setup placement and connections with other parts, e.g. a single object per each plastic scintillator strip with its location in the detector or a photomultiplier coupled to a given scintillator. Data Objects are structures representing subsequent stages of reconstructed data – from elementary ones containing only TDC time and data acquisition channel number, to a detailed reconstruction of a physical event or a line-of-response.

Data Objects refer to particular elements of the setup encapsulated in Parameter Objects where the physical signals have originated. Moreover, mapping of connections between such components imposes relations between Parameter Objects themselves. These relations are implemented using persistent object references (*TRef*) provided by the ROOT libraries [24] which ensure  $\mathcal{O}(1)$  lookup of corresponding elements as well as persistence of the relations across file storage.

On the user side, encapsulation of data and setup properties into abstract objects allows for definition of reconstruction and analysis logic even by users without programming proficiency which is one of the objectives of the J-PET Framework. Listing 2 demonstrates the interplay between Data (*JPetHit* and *JPetEvent*) and Parameter Objects (*Scintillator* and *BarrelSlot*) in a simple task.

### 2.2.5. Setup description

Since experimental setup and its conditions can change from one measurement to another, the set of parameters describing it must be generated dynamically. The library provides dedicated tools to handle a setup representation in a form of a configuration `JSON` file. Based on its content, the Framework generates the collection of Parameter Objects (see section 2.2.4) together with relations between them expressed in a standardized format. Once this file is parsed, a Parameter Bank encapsulating the latter is embedded in all output data files to allow for their further stand-alone analysis.

### 2.2.6. Processing control and logging

Each execution instance of any application based on the Framework environment, generates a log file with a unique name. By default all input parameters and options are stored in the log file, moreover each task can produce custom messages with one of appropriate tags: `INFO`, `DEBUG`, `ERROR`.

User Tasks classes can use tools for creating control histograms. All such objects are then automatically stored in the output file. An usage example can be found in code snippet 3.

### 2.2.7. Compressed input files

It is also possible to provide a raw data file in a compressed format; in that case a task is added by default, that simply decompresses the input file before any other procedures begin. Supported formats are: `xz`, `gz`, `bz2`, `zip`.

### 2.2.8. Handling of binary data format

The library can read raw data input files provided from the scanner data acquisition or from digital oscilloscope measurements. Binary data is transformed with dedicated tasks in the `ROOT` format, making it available for further processing by the consecutive tasks in the stream.

### 2.2.9. Iterative tasks

The structure of task chain allows the implementation of iterative tasks schemes, in which a module can be executed in a loop till a given condition is fulfilled. The stopping condition can be based on desired number of consecutive iterations or on the return value of the function defined by the user. This functionality is especially useful for optimization goals, i.e. refining detector calibration constants or estimation of event classification parameters.

### 2.2.10. Interfaces to Monte Carlo simulation packages

Testing and debugging of data analysis modules is often supported by using Monte Carlo-simulated events in place of actual data. To this end, the J-PET Framework offers interfaces to two Monte Carlo simulation packages: the custom J-PET MC simulation software [25] based on the Geant4 toolkit [26] as well as the GATE package for simulation of PET and SPECT tomography [27].

MC-simulated events are wrapped into the same data structures as data so that analysis modules intended to process experimental data can be applied transparently to the simulation results. At the same time, all MC-specific event information is preserved and accessible on demand.

### 2.2.11. Event Display

J-PET Event Display [28] is a visualization tool based on the J-PET library. It can load files with the Framework data structures to visualize the reconstructed PET data in an event-by-event manner at different phases of the processing. Information on input geometry of the detector is provided by the same configuration files in the `JSON` format used for reconstruction of data described in section 2.2.5. A usage example of the Event Display is shown in Fig. 4.

## 2.3. Development philosophy, testing and continuous integration

The J-PET Framework developer community is trying to consistently adopt good coding rules and practices in the development routine to assure the quality of the software. In particular, any new code before being merged into official repository must be reviewed and accepted by at least one person not being the author. Moreover, it must pass a set of unit and integration tests defined for the platform. The contributors are strongly encouraged to add unit tests together with new classes and to format the code consistently using the clang-format tool.

The Continuous Integration process is integrated with the project Github repository. Any new pull request launches automatic set of tests based on the Travis [29] and Jenkins [30] services. The unit tests are operated by the Travis system, while larger integration tests, which typically require some input data, are run by a dedicated Jenkins server. Both services deliver a detailed report about possible failures. The testing system is fully automatized on the servers and can be launched manually for local testing. .

Issue and bug tracking is performed with the Redmine service which also serves also as a user support forum. The reference guide is automatically generated from the code using the Doxygen tool and is available online [31]. Additionally, an analysis user guide is provided in the repository and it is being updated with every new version of the Framework.

## 2.4. Sample code snippets analysis

Listing 1 presents the instance of `JPetManager` registering user tasks to form a chain of procedures. With the following `useTask` method, the user is specifying the input and output data format of each tasks. In the example, the output of the first task serves as input for the second one. The processing of all algorithms with the provided arguments is triggered by the `run` method. This simple construction allows to create an analysis from custom building blocks even for a user with little programming experience.

Listing 2 presents a snippet of an analysis module identifying 2-photon coincidence events in a stream of single recorded photon interactions (referred to as *hits*). Listing 3 shows a basic usage of the statistics facilities for creation of histograms to be filled during data analysis.

```

1 #include <JPetManager/JPetManager.h>
2 #include "Task1.h"
3 #include "Task2.h"
4
5 using namespace std;
6
7 int main (int argc, const char * argv []) {
8     JPetManager& manager = JPetManager::getManager();
9
10    manager.registerTask<Task1>("Task1");
11    manager.registerTask<Task2>("Task2");
12
13    manager.useTask("Task1", "data.input", "data.type1");
14    manager.useTask("Task2", "data.type1", "data.type2");
15
16    manager.run(argc, argv);
17 }

```

Listing 1: Exemplary main class of a program based on J-PET Framework library.

```

1 for(JPetHit& hit_1: gamma_hits){
2     for(JPetHit& hit_2: gamma_hits){
3         // find double coincidences within 5000 ps
4         if(hit_2.getTime() - hit_1.getTime() < 5000.){
5             // check if the two gamma interactions were recorded
6             // in distinct scintillators of the setup
7             if(hit_1.getScintillator() != hit_2.getScintillator()){
8                 // check if locations of the two detection modules
9                 // differ by more than 160 degrees in azimuthal angle
10            if(fabs(hit_1.getBarrelSlot().getTheta() -
11                hit_2.getBarrelSlot().getTheta()) > 160.){
12                // reconstruct e+e- -> 2gamma annihilation point
13                TVector3 point =
14                    EventCategorizerTools::calculateAnnihilationPoint(hit_1, hit_2);
15                // assemble an event containing the two hits
16                JPetEvent event;
17                event.addHit(hit_1);

```

```

18     event.addHit(hit_2);
19     event.setEventType(JPetEventType::k2Gamma);
20     // automatically store the event in the output file
21     // or pass on to the next analysis module
22     fOutputEvents->add<JPetEvent>(event);
23 }
24 }
25 }
26 }
27 }

```

Listing 2: Exemplary naive procedure of finding 2-photon coincidence events demonstrating the ease of operations on the data structures provided by the Framework.

```

1 // Creating histogram with JPetStatistics class
2 getStatistics().createHistogram(
3     new TH1F("hit_z_pos", "Z-axis position of photon interaction in plastic
4         scintillator", 200, -25.0, 25.0));
5 getStatistics().getHisto1D("hit_z_pos")->GetXaxis()->SetTitle("Z-axis
6     position [cm]");
7 getStatistics().getHisto1D("hit_z_pos")->GetYaxis()->SetTitle("Number of
8     Hits");
9
10 // Invoking a histogram by title from statistics interface for filling
11 getStatistics().getHisto1D("hit_z_pos")->Fill(hit.getPosZ())

```

Listing 3: Example of using tools for creating filling histograms, that are stored in output files.

### 3. Illustrative Examples

In this section we present two examples developed with the Framework library. The first application can be used to perform tests of a prototype PET scanner based on the Monte Carlo simulations of various phantoms. The program loads the data sample generated by the GATE Monte Carlo simulation package and transforms it by smearing the measured observables such as time, energy and position based on the parametrization of experimental uncertainties determined for a given prototype scanner. This procedure mimics the real measurement effects. Next, the data is reconstructed and finally transformed to a sinogram, which serves as an input to the image reconstruction task implementing the Time-of-Flight Filtered-Backprojection algorithm (see Figure 3) or can be send to an external image reconstruction package such as STIR [32], CASTOR [33] or QETIR [34]. All operations are implemented as consecutive tasks executed by the framework.

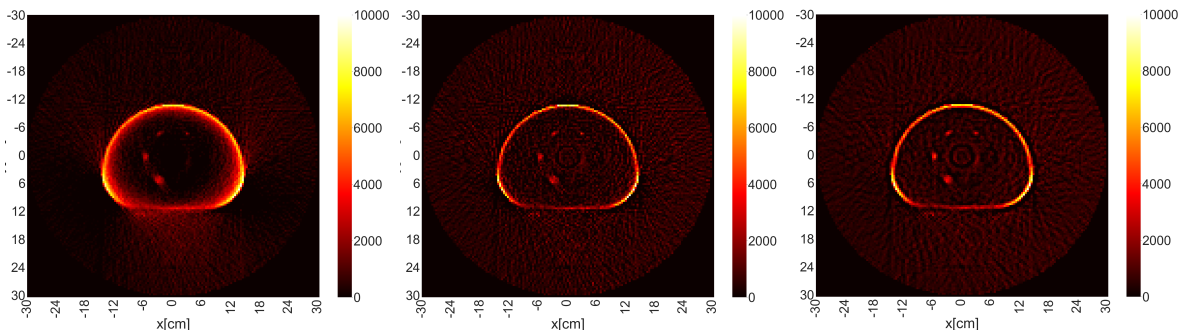


Figure 3: Example image reconstructed with the Time-of-Flight Filtered-Backprojection algorithm with various filters: Ramlak (left), Shepp-Logan (center), Hamming (right). The input sample is based on Monte Carlo simulations of NEMA IEC phantom performed with the GATE package [27], and further processed by the Framework-based parser which applies the experimental parametrizations to fully imitate a measurement of the scanner.



The second program implements a full reconstruction chain for the real data collected by the 3-layer J-PET scanner. The example reconstruction and analysis is based on the test measurement with the radioactive source placed in the center of the scanner. The reconstructed results are visualized with the J-PET Event Display tool (see Figure 4).

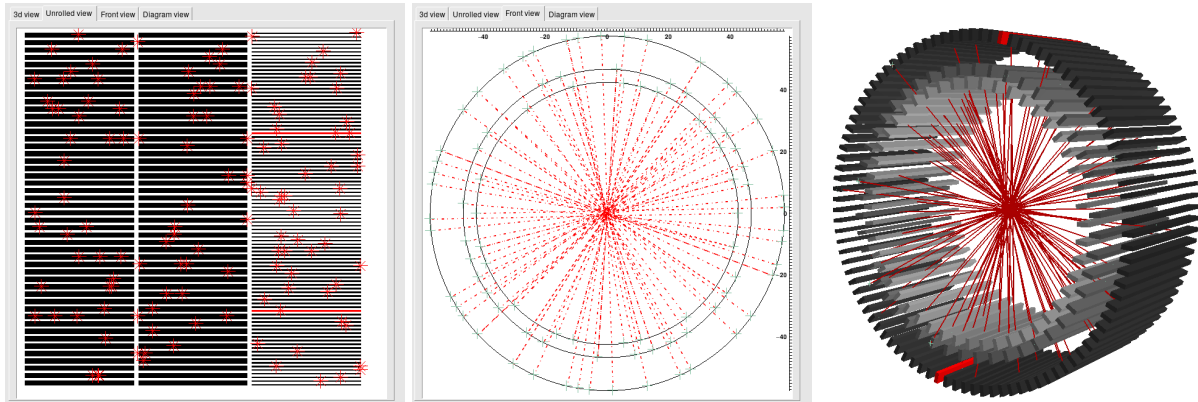


Figure 4: Screenshots of J-PET Event Display [28], (see 2.2.11). This example visualizes result of data reconstruction acquired from a test measurement with radioactive source in the center of the scanner. Figures show the model of the J-PET detector, consisting of plastic strips, arranged as 3 concentric cylinders. The right image shows a 3D view, center is a frontal view and left shows these detector elements with positions of reconstructed interactions marked with red stars. In the center and right figures red lines connect pairs of reconstructed positions, that satisfy selection criteria aiming in preparing the sample of events of electron-positron annihilation.

#### 4. Impact

Flexibility and robustness of the J-PET Framework library allowed it to be adopted as the main software platform of the J-PET project. The software and applications constructed based on this package have been used for many kinds of scientific studies involving data analyses from the J-PET tomography scanner and will be utilized for future analyses in the fundamental research and in the development of various PET scanners prototypes.

- performance assessment of novel PET scanners [35],
- time calibration techniques for PET scanners [36, 37],
- parametrization of deposited energy in plastic scintillators by Time-over-Threshold measurements [38],
- implementation of PET image reconstruction techniques such as Kernel Density Estimation, Maximum Likelihood Expectation Maximization [39] and Time-of-Flight Filtered-Backprojection,
- development of plastic-based prototype a Positron Emission Mammography scanner [40],
- studies in positronium annihilation reconstruction and imaging [41, 15, 17],
- fundamental research on photon polarization and quantum entanglement [42, 43],
- tests of discrete symmetries [44, 45],
- mirror matter searches [46].

The Framework software platform is currently used by scientists from the Jagiellonian University in Kraków, National Centre for Nuclear Research in Warsaw and INFN Laboratori Nazionali di Frascati and has been successfully deployed on different scales starting from laptops and personal PC-s, through mid-size computing clusters to HPC Swierk cluster.

## 5. Conclusions

In this article we presented the features and range of possible applications of the J-PET Framework, a C++ based library for data processing and analysis for PET tomography and for fundamental searches. The Framework provides tools for the implementation of a wide range of data reconstruction and calibration procedures as well as user-level data analyses and preparation of input for higher-level medical imaging software. The platform is focused on flexibility in adjusting to dynamically changing prototyping environments and asserting ease of implementation of the required logic by users without programming proficiency while maintaining high processing performance.

Currently, use cases of the J-PET Framework span among various data analyses and imaging application of the first J-PET device. In the near future, a new generation light-weight modular J-PET scanner with fully digital readout and high mobility will be commissioned along with sibling devices such as a similar-technology-based mammography scanner. The software platform is currently being extended with modules specific to the new devices, which will allow for reusing the higher-level analysis steps with data from new hardware setups.

Despite having originated solely for the purpose of analysis of data from a single setup, the recent expansion of the scope of its usage in the context of J-PET demonstrates that the flexibility of its architecture allows for use in a wider range of experiments related to nuclear medicine and fundamental studies. .

## 6. Conflict of Interest

No conflict of interest exists: We wish to confirm that there are no known conflicts of interest associated with this publication and there has been no significant financial support for this work that could have influenced its outcome.

## Acknowledgements

This work was supported in part by the Foundation for Polish Science through the Grant No. TEAM POIR.04.04.00-00-4204/17.

## References

- [1] GitHub Repository of the J-PET Framework library, <https://github.com/JPETTomography/j-pet-framework>.
- [2] User manual of the J-PET Framework, [https://github.com/JPETTomography/j-pet-framework/blob/master/doc/manual/J-PET\\_Framework\\_Guide\\_v8.pdf](https://github.com/JPETTomography/j-pet-framework/blob/master/doc/manual/J-PET_Framework_Guide_v8.pdf).
- [3] Redmine issue tracker of the J-PET Framework, <http://sphinx.if.uj.edu.pl/redmine>.
- [4] R. D. Badawi, et al., First Human Imaging Studies with the EXPLORER Total-Body PET, *The Journal of Nuclear Medicine* 60 (3) (2019) 299–303.
- [5] S. R. Cherry, R. D. Badawi, J. S. Karp, W. W. Moses, P. Price, T. Jones, Total-body imaging: Transforming the role of positron emission tomography, *Science Translational Medicine* 9 (381) (2017). doi:10.1126/scitranslmed.aaf6169.
- [6] S. R. Cherry, et al., Total-Body PET: Maximizing Sensitivity to Create New Opportunities for Clinical Research and Patient Care, *The Journal of Nuclear Medicine* 59 (1) (2018) 3–12.
- [7] S. Reardon, Whole-body PET scanner produces 3D images in seconds, *Nature* 570 (2019) 285–286. doi:10.1038/d41586-019-01833-z.
- [8] X. Zhang, J. Zhou, S. R. Cherry, R. D. Badawi, J. Qi, Quantitative image reconstruction for total-body pet imaging using the 2-meter long explorer scanner, *Phys. Med. Biol.* 62 (6) (2017) 2465. URL <http://stacks.iop.org/0031-9155/62/i=6/a=2465>

- [9] P. Moskal, et al., Test of a single module of the j-pet scanner based on plastic scintillators, *Nucl. Instrum. Methods A* 764 (2014) 317.
- [10] S. Niedźwiecki, et al., J-PET: a new technology for the whole-body PET imaging, *Acta Phys. Polon. B*48 (2017) 1567. [arXiv:1710.11369](https://arxiv.org/abs/1710.11369), [doi:10.5506/APhysPolB.48.1567](https://doi.org/10.5506/APhysPolB.48.1567).
- [11] P. Kowalski, et al., Estimation of the NEMA characteristics of the J-PET tomograph using the GATE package, *Phys. Med. Biol.* 63 (2018) 165008.
- [12] P. Moskal, et al., Feasibility study of the positronium imaging with the J-PET tomograph, *Phys. Med. Biol.* 64 (2019) 055017.
- [13] A. Ruciński, et al., Investigations on Physical and Biological Range Uncertainties in Krakow Proton Beam Therapy Centre, *Acta Phys. Pol. B* 51 No. 1 (2020) 9.
- [14] A. Gajos, E. Czerwiński, D. Kamińska, P. Moskal, Multi-tracer morphometric image reconstruction method and system using fast analytical algorithm for calculating time and position of positron-electron annihilation into three gamma quanta, International patent application number PCT/PL2015/050038.
- [15] B. Jasińska, P. Moskal, A new PET diagnostic indicator based on the ratio of 3gamma/2gamma positron annihilation, *Acta Phys. Polon. B*48 (2017) 1577.
- [16] D. Kamińska, et al., A feasibility study of ortho-positronium decays measurement with the J-PET scanner based on plastic scintillators, *Eur. Phys. J. C*76 (8) (2016) 445. [arXiv:1607.08588](https://arxiv.org/abs/1607.08588), [doi:10.1140/epjc/s10052-016-4294-3](https://doi.org/10.1140/epjc/s10052-016-4294-3).
- [17] P. Moskal, B. Jasińska, E. Stepień, S. Bass, Positronium in medicine and biology, *Nature Reviews Physics* 1 (2019) 527–529.
- [18] G. Korcyl, et al., Evaluation of Single-Chip, Real-Time Tomographic Data Processing on FPGA SoC Devices, *IEEE Transactions on Medical Imaging* 37 No. 11 (2018) 2526.
- [19] M. Traxler, E. Bayer, M. Kajetanowicz, G. Korcyl, L. Maier, J. Michel, M. Palka, C. Ugur, A compact system for high precision time measurements ( $< 14$  ps RMS) and integrated data acquisition for a large number of channels, *Journal of Instrumentation* 6 (12) (2011) C12004–C12004. [doi:10.1088/1748-0221/6/12/c12004](https://doi.org/10.1088/1748-0221/6/12/c12004).
- [20] A. Neiser, J. Adamczewski-Musch, M. Hoek, W. Koenig, G. Korcyl, S. Linev, L. Maier, J. Michel, M. Palka, M. Penschuck, M. Traxler, C. Uğur, A. Zink, TRB3: a 264 channel high precision TDC platform and its applications, *Journal of Instrumentation* 8 (12) (2013) C12043–C12043. [doi:10.1088/1748-0221/8/12/c12043](https://doi.org/10.1088/1748-0221/8/12/c12043).
- [21] List of experiments based on the TRB3 readout platform, <http://trb.gsi.de/>, accessed: 2020-01-30.
- [22] W. Krzemien, et al., Analysis framework for the j-pet scanner, *Acta Phys. Pol. A* 127 (2015) 1491.
- [23] GitHub Repository of Examples based on J-PET Framework Library, <https://github.com/JPETTomography/j-pet-framework-examples>.
- [24] R. Brun, F. Rademakers, ROOT - An Object Oriented Data Analysis Framework, *Nucl. Instrum. Methods A* 389 (1997) 81–86.
- [25] GitHub Repository of the J-PET Monte Carlo simulation package, <https://github.com/JPETTomography/J-PET-geant4>.
- [26] S. Agostinelli, et al., Geant4 simulation toolkit, *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* 506 (3) (2003) 250 – 303. [doi:https://doi.org/10.1016/S0168-9002\(03\)01368-8](https://doi.org/10.1016/S0168-9002(03)01368-8).  
URL <http://www.sciencedirect.com/science/article/pii/S0168900203013688>

- [27] S. Jan, et al., GATE: a simulation toolkit for PET and SPECT, *Phys. Med. Biol.* 49 (19) (2004) 4543–4561. doi:10.1088/0031-9155/49/19/007.
- [28] GitHub Repository of the J-PET Event Display package, <https://github.com/JPETTomography/j-pet-event-display>.
- [29] The Travis Continuous Integration Platform, <https://travis-ci.org/>.
- [30] The Jenkins Automation Server, <https://jenkins.io/>.
- [31] Automatic reference guide of the J-PET Framework, <http://sphinx.if.uj.edu.pl/framework/doc/>.
- [32] K. Thielemans, et al., Stir: Software for tomographic image reconstruction release 2, *Phys. Med. Biol.* 57 (2012). doi:<https://doi.org/10.1088/0031-9155/57/4/867>.
- [33] T. Merlin, et al., Castor: a generic data organization and processing code framework for multi-modal and multi-dimensional tomographic reconstruction, *Phys. Med. Biol.* 63 (2018). doi:<https://doi.org/10.1088/1361-6560/aada1>.
- [34] Quantitative Emission Tomography Iterative Reconstruction at the MEDISIP website., [http://medisip.ugent.be/?page\\_id=50#QETIR](http://medisip.ugent.be/?page_id=50#QETIR).
- [35] M. Pawlik-Niedźwiecka, et al., Preliminary Studies of J-PET Detector Spatial Resolution, *Acta Phys. Polon.* A132 (2017) 1645.
- [36] M. Skurzok, et al., Time Calibration of the J-PET Detector, *Acta Phys. Polon.* A132 (2017) 1641. arXiv:1710.05598.
- [37] K. Dulski, M. Silarski, P. Moskal, A Method for Time Calibration of PET Systems Using Fixed  $\beta+$  Radioactive Source, *Acta Phys. Polon.* B51 (2020) 195. doi:10.5506/APhysPolB.51.195.
- [38] S. Sharma, et al., Estimating relationship between the Time Over Threshold and energy loss by photons in plastic scintillators used in the J-PET scanner (2020). arXiv:1911.12059.
- [39] A. SÁĆomski, et al., 3D PET image reconstruction based on Maximum Likelihood Estimation Method (MLEM) algorithm, *Bio-Algorithms and Med-Systems* 10(1) (2014) 1–7.
- [40] Shivani, E. Luczyńska, S. Heinze, P. Moskal, Development of J-PEM for breast cancer detection and diagnosis using positronium imaging, *Acta Phys. Polon.* B51 (2020) 281. arXiv:1912.04282, doi:10.5506/APhysPolB.51.281.
- [41] A. Gajos, et al., Trilateration-based reconstruction of ortho-positronium decays into three photons with the j-pet detector, *Nucl. Instrum. Methods A* 819 (2016) 54–59.
- [42] P. Moskal, et al., Feasibility studies of the polarization of photons beyond the optical wavelength regime with the J-PET detector, *Eur. Phys. J. C* 78 (2018) 970. arXiv:1809.10397, doi:10.1140/epjc/s10052-018-6461-1.
- [43] B. C. Hiesmayr, P. Moskal, Witnessing Entanglement In Compton Scattering Processes Via Mutually Unbiased Bases, *Scientific Reports* 9 (2019) 8166. doi:10.1038/s41598-019-44570-z.
- [44] J. Raj, et al., A feasibility study of the time reversal violation test based on polarization of annihilation photons from the decay of ortho-Positronium with the J-PET detector, *Hyperfine Interact* 239 (2018) 56. arXiv:1809.00847, doi:10.1007/s10751-018-1527-x.
- [45] A. Gajos, et al., Feasibility study of the time reversal symmetry tests in decays of metastable positronium atoms with the J-PET detector, *Advances in High Energy Physics* 2018 (2018) 10. arXiv:1804.07148, doi:10.1155/2018/8271280.
- [46] W. Krzemien, E. P. del Rio, K. Kacprzak, Feasibility of ortho-positronium lifetime studies with the J-PET detector in context of mirror matter models, *Acta Phys. Polon.* B51 (2020) 165. arXiv:1911.10589, doi:10.5506/APhysPolB.51.165.