Nuclear Instruments and Methods in Physics Research B 306 (2013) 59–63

Contents lists available at SciVerse ScienceDirect

Nuclear Instruments and Methods in Physics Research B

journal homepage: www.elsevier.com/locate/nimb

An ion beam analysis software based on ImageJ

C. Udalagama *, X. Chen, A.A. Bettiol, F. Watt

Centre for Ion Beam Applications (CIBA), Department of Physics, National University of Singapore, 2 Science Drive 3, Singapore 117 542, Singapore

ARTICLE INFO

Article history:
Received 1 August 2012
Received in revised form 21 December 2012
Accepted 21 December 2012
Available online 20 January 2013

Keywords:
Ion beam analysis
ImageJ
Software

ABSTRACT

The suit of techniques (RBS, STIM, ERDS, PIXE, IL, IF,...) available in ion beam analysis yields a variety of rich information. Typically, after the initial challenge of acquiring data we are then faced with the task of having to extract relevant information or to present the data in a format with the greatest impact.

This process sometimes requires developing new software tools. When faced with such situations the usual practice at the Centre for Ion Beam Applications (CIBA) in Singapore has been to use our computational expertise to develop ad hoc software tools as and when we need them. It then became apparent that the whole ion beam community can benefit from such tools; specifically from a common software toolset that can be developed and maintained by everyone with freedom to use and allowance to modify. In addition to the benefits of readymade tools and sharing the onus of development, this also opens up the possibility for collaborators to access and analyse ion beam data without having to depend on an ion beam lab. This has the virtue of making the ion beam techniques more accessible to a broader scientific community.

We have identified ImageJ as an appropriate software base to develop such a common toolset. In addition to being in the public domain and been setup for collaborative tool development, ImageJ is accompanied by hundreds of modules (plugins) that allow great breadth in analysis. The present work is the first step towards integrating ion beam analysis into ImageJ. Some of the features of the current version of the ImageJ 'ion beam' plugin are: (1) reading list mode or event-by-event files, (2) energy gates/sorts, (3) sort stacks, (4) colour function, (5) real time map updating, (6) real time colour updating and (7) median & average map creation.

1. Introduction

The last few decades have witnessed substantial development and refinement of the ion beam techniques and instrumentation to the extend that we are now in a position to image with resolution of tens of nanometres [1], resolve depths at sub-10 nm resolution [3–5], produce structures with sub-100 nm resolution [6] and even produce 3D structures [7–9]. The suit of ion beam techniques offer so much versatility that it comes as no surprise that these techniques can be reused for newer, more topical, research projects [2,10–12].

In order to advance the field of ion beams even farther we need to constantly make our presence felt in the scientific community by using our repository of techniques to produce noteworthy science; especially in areas potentially inaccessible to the other techniques (e.g. whole cell imaging [10]). Presently, one aspect that makes this challenging is the extensive training that is required by an individual to properly use a nano/micro beam system, which precludes a non-expert using it as one would a Scanning Electron Microscopy (SEM) or Atomic Force Microscopy (AFM). Further, even if an expert does indeed perform the necessary experiments and collect data, the non-expert still faces difficulty in analysing this data owing to ion beam software being expensive, highly specialised and, more often than not, confined (spatially) to the ion beam laboratory.

The versatility of the ion beam techniques allow many different types of information to be extracted from a given set of data. A good example would be Rutherford Backscattering Spectroscopy (RBS) where one traditionally extracts composition or depth data [13]. However by incorporating beam focusing and scanning we can extend this to 2D images. RBS can actually be extended further to produce 3D images. Such 3D images may allow greater insight into the topic being studied and, in todays competitive scientific arena, such 3D images may be the difference that pushes that balance towards a manuscript being accepted for publication. Although such 3D imaging techniques have been assiduously studied and software developed [14,15] it is non-trivial to implement with traditional data acquisition software [16–19]. In fact it is common to encounter such situations where the theory and background have been previously studied but requires a substantial investment in time and effort to reproduce to be applicable to a

* Corresponding author.
E-mail address: chammika@nus.edu.sg (C. Udalagama).

0168-583X/$ - see front matter © 2013 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.nimb.2012.12.026

new project. This same argument applies to many other data analysis tools (e.g. an angular line profile with a broad line, dynamic colour adjustments,...) the unavailability of which might limit many opportunities where raw data could have been better presented or more thoroughly analysed.

Whenever such a software tool is necessary one strategy is to combine whatever available computational expertise and computational software to come up with ad hoc solution. This has been the practice at the Centre for Ion Beam Applications (CIBA), NUS Singapore, where we 'mix and match' software tools such as C++, Java, MATLAB (www.mathworks.com), Mathematica (www.wolfram.com) to come up with ad hoc software solutions. One downside to this approach is that the tools become inaccessible when licensing to these software expires or when the developer is no longer available for consultation. Another is that sharing of these software tools between ion beam labs becomes challenging if not impossible. This is unfortunate as a set of common tools will allow each lab to readily benefit from the work and expertise of another, thereby helping the field to progress faster; not to mention the many man hours that will be saved.

It is timely to establish a base for the ion beam community to develop and share common software tools. We need to have an optimised toolset that can take us from raw experimental data to a publication quality format in the least possible time with the least possible effort. We also need these tools to be available to our collaborators. In order for this to be practical we not only need to have free access but also the freedom to modify and improve these common tools. This will allow more than one individual to develop these tools, potentially guaranteeing indefinite maintenance of the toolset and independence on a single individual. Another useful feature should be the accessibility to other specialised data analysis tools from other fields to use with our ion beam data. A software base that satisfies all these criteria is ImageJ [20](http://imagej.nih.gov/ij/) and this article describes the first steps towards incorporate ion beam data via a 'ion beam' module (plugin). ImageJ has been developed and used since 1997. The development path adopted by ImageJ has been made robust by the thousands of individuals involved in the process. A look at the ImageJ plugins website also indicates the versatility and extent of the use of ImageJ. It is this ubiquity and utility that has prompted us to adopt ImageJ as its development history indicates future longevity.

2. ImageJ

The following introduction extracted from the ImageJ user manual [21] serves best to describe it: "ImageJ is a public domain Java image processing and analysis program inspired by NIH Image for the Macintosh. It runs, either as an online applet or as a downloadable application, on any computer with a Java 1.5 or later virtual machine. Downloadable distributions are available for Windows, Mac OSX and Linux. It can display, edit, analyse, process, save and print 8-bit, 16-bit and 32-bit images. It can read many image formats including TIFF, GIF, JPEG, BMP, DICOM, FITS and 'raw' ". The user manual also provides extensive information on resources related to image processing and also the ethics in scientific image processing.

As indicated above since ImageJ is within the public domain there is freedom to change and improve those parts of ImageJ that a user deems necessary. Thus the legalities of using, modifying and sharing ImageJ and its components are well defined. Further, since Java runs via a virtual machine it is easily ported between different platforms with almost no modification required, thereby providing platform independence. Although the core of ImageJ has many features, the real power of this software stems from the large number (hundreds) of modules (plugins) that augments additional functionality. These plugins are contributed by various individuals and shared is the open domain. There are many resources on the web related to developing ImageJ plugin, but a good starting point is Werner Bailer's tutorial [22]. One is also directed to Fiji (Fiji is just ImageJ, http://fiji.sc) which offers a convenient scripting environment for testing plugins.

A more detailed description of ImageJ is prohibitive owing to the immensity of its breadth. The reader is directed to the user manual of further details. However, one feature well worth noting is ImageJ's macro language feature, as this can greatly speed-up repetitive tasks, which one often encounters in ion beam analysis.

3. The ion beam plugin

3.1. Entry point

ImageJ requires that the names of the Java classes of its plugins have an underscore. In accordance the main Java class of our plugin is named '`IonBeamAnalysis_`'. More information on how to incorporate a plugin into ImageJ can be found in Section 3.5. Our plugin begins by allowing the user to choose a list mode or event-by-event file that records experiment data according to the TQSA philosophy [23]. The format of the file used is discussed later in Section 4. Once this file is specified the plugin loads the file; keeping the user abreast of the loading procedure via the progress bar in the main ImageJ toolbar. On completion a panel displays a summary dialogue relevant to the file. The user can at this stage specify, by providing a starting and ending percentage, which portion of the experiment to be analysed. The plugin, which accepts data collected from up to five(5) ADCs, then allows the user the option of viewing any of the available spectra via a right-mouse-click popup (contextual) menu.

3.2. The spectral window

The spectrum of the chosen ADC is displayed using dedicated window based on JFreeChart class library. JFreeChart is an open source class made available under the GNUL license and offers many convenient features for displaying graphical data (http://www.jfree.org/jfreechart/). This spectrum window is interactively resizable and offers a convenient zooming feature via the left mouse button. On right-mouse-clicking on this spectrum the user is offered further options to analyse the data. This spectral window can be seen in Fig. 1. Table 1 show a summary of the options that are available in the present version of the plugin which includes generating Median/Average maps, incorporating a Gate/Sort), changing the colour bar and choosing a colour scheme.

3.3. Interactive gates and stack of gates

One of the features of our plugin is to interactively specify a gate via the mouse. Upon activation another window appears with the image relevant to the gated data (Fig. 1). This image dynamically changes according to any adjustments that the user makes to the gate boundaries. In order to avoid unnecessary usage of CPU time, the boundaries of the gate can be changed only by a combination of left-mouse-clicking while holding the ALT button.

The window that is used to display the image is an ImageJ native ImagePlus window with full integration with the ImageJ system thereby allowing all the core tools of ImageJ and the other plugins to be immediately applied to it.

Another option available to the user is to display a gate or sort stack. How the boundaries of a sort stack is specified is identical to