SoftwareX 6 (2017) 148–149

Contents lists available at ScienceDirect

SoftwareX

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

Software update

Update 1.5 to "Takin: An open-source software for experiment planning, visualisation, and data analysis", (PII: S2352711016300152)

Tobias Weber

Physik-Department E21, Technische Universität München, James-Franck-Str. 1, 85748 Garching, Germany

ARTICLE INFO

Article history:
Received 4 May 2017
Accepted 8 June 2017

Keywords:
Triple-axis spectroscopy
Instrument control
Reciprocal and real space visualisation
Brillouin zones
Instrumental resolution

ABSTRACT

We present an updated version of our inelastic neutron scattering software package *Takin*, which is a programme for neutron triple-axis experiment planning and evaluation. The new version features several additional programme modules mainly concerning three-dimensional calculations and visualisations. In addition, existing modules have been improved and extended.

Code metadata

Current code version	1.5
Permanent link to code/repository used for this code version	https://github.com/ElsevierSoftwareX/SOFTX-D-17-00033
Legal Code License	GNU GPL Version 2
Code versioning system used	git
Software code languages, tools, and services used	C++ 11, CMake 3
Compilation requirements, operating environments	Linux, OS X, Unix like, Windows (via MinGW); GCC 4.8 (or later) or Clang, Boost, Qt 4 or 5, Qwt 5 or 6, Clipper, Minuit 2, Freetype
If available Link to developer documentation/manual	
Support email for questions	tobias.weber@tum.de

Software metadata

Current software version	1.5
Permanent link to executables of this version	http://wiki.mlz-garching.de/takin
Legal Code License	GNU GPL Version 2
Computing platform/Operating System	Linux, OS X, Unix like, Windows (MinGW test version)
Installation requirements & dependencies	
If available Link to user manual	http://wiki.mlz-garching.de/takin
Support email for questions	

1. Introduction

Several updates to the free and open-source software (FOSS) *Takin* [1,2] have been made over the course of the last year. Apart from general performance optimisations and improvements of the existing programme modules, new features have been introduced. Of these features, we present two selected ones in the present short paper.

2. 3D Brillouin zones

The previous version of our software included code to calculate a two-dimensional (2D) slice of the first Brillouin zone relevant

DOI of original article: http://dx.doi.org/10.1016/j.softx.2016.06.002.
E-mail address: tobias.weber@tum.de.

http://dx.doi.org/10.1016/j.softx.2017.06.002

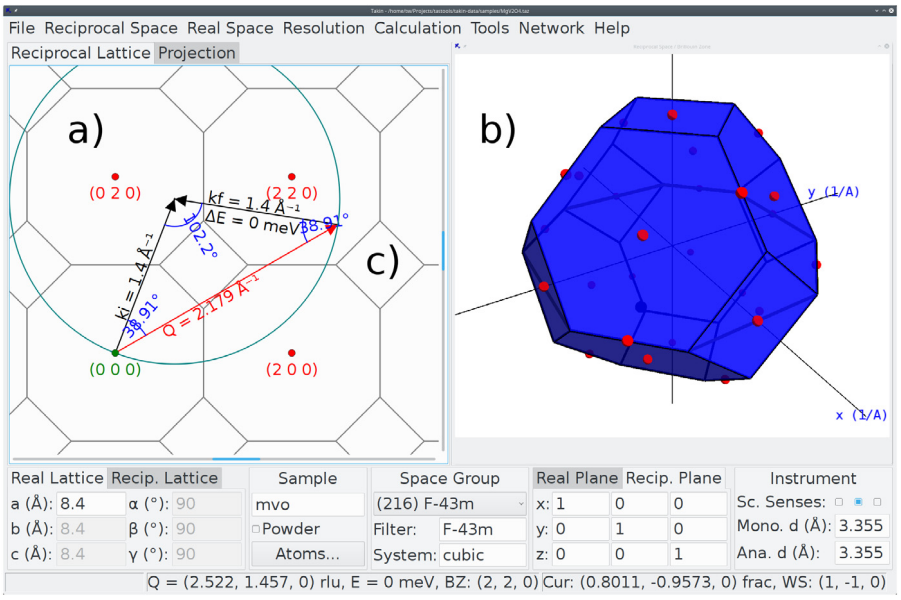

Fig. 1. Example calculation of the first Brillouin zone in the fcc crystal MgV_2O_4. (a) The two-dimensional cuts of the Brillouin zone in the $\langle hk0 \rangle$ plane can only be correctly calculated using the full three-dimensional information. In a 2D calculation, the diamond-shaped borders of the zone along the [110] directions – marked with (c) – would be missing. (b) The full results of the 3D calculations are displayed using *OpenGL*. Here, the red points mark positions of high symmetry. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

for the selected scattering plane. For complexity and performance reasons, the code used for the task remained purely 2D. While this gives reasonable results in most cases, it is only an approximation.

For the present version, we created a new programme module which calculates the full three-dimensional (3D) Brillouin zone from the single-crystal parameters and the space group on-the-fly. The code is highly optimised and has only a very slight performance penalty when compared to the old 2D version. The Brillouin zone can now either be visualised in the software using *OpenGL* [3] or be exported to the open *X3D* file format. The *X3D* export allows for processing the data in external applications, for instance the FOSS programme *Blender* [4]. Furthermore, 2D cuts of the 3D Brillouin zones are generated using the selected scattering plane. These cuts are presented in the main reciprocal space display, replacing the old pure-2D code. An example for these calculations is shown in Fig. 1.

3. Scripting interfaces

In its resolution convolution code, *Takin* uses externally supplied scripts for the calculation of the dynamical structure factor $S(\mathbf{Q}, E)$. In addition to *Python* [5], we now support the novel scripting language *Julia* [6] for this task. As *Julia* is just-in-time compiled, it considerably speeds up the calculation of the convolution integral.

In order to circumvent threading limitations in the *Python* interpreter and for stability reasons, both the *Python* and *Julia* scripting interfaces are now separated from the main programme. $S(\mathbf{Q}, E)$ script modules are now spawned in their own process space and communicate with *Takin* using a shared-memory messaging system.

Acknowledgements

This work was supported by the German Research Foundation (DFG) and the Technische Universität München within the funding programme Open Access Publishing. We are mainly indebted to Peter Böni and Robert Georgii for their continuing support of these scientific computing projects. Additionally, we want to thank Petr Čermák, Astrid Schneidewind, Jitae Park, Greg Tucker, and Bertrand Roessli for helpful discussions and for reporting bugs in the previous version of the software. Many thanks go to Alain Filhol for his continuous help on improving the *OS X* build. Also thanks to Petr Čermák for his website at MLZ, which hosts the binaries.

References

[1] Weber T, Georgii R, Böni P. Takin: An open-source software for experiment planning, visualisation, and data analysis. SoftwareX 2016;5:121–6. http://dx.doi.org/10.1016/j.softx.2016.06.002.
[2] Weber T. *Physikdepartment E21* [Ph.D. thesis], Garching, Germany: Technische Universität München; 2016.
[3] Khronos Group. OpenGL. URL https://www.khronos.org/opengl/.
[4] Blender Foundation. Blender 3D software suite. URL https://www.blender.org/.
[5] Van Rossum G., et al., Python programming language. URL https://www.python.org/.
[6] Bezanson J, Edelman A, Karpinski S, Shah VB. Julia: A fresh approach to numerical computing. SIAM Rev 2017;59(1):65–98. http://dx.doi.org/10.1137/141000671. URL https://julialang.org/.