Journal of Physics: Conference Series **490** (2014) 012074 doi:10.1088/1742-6596/490/1/012074

# A computational tool to design and generate crystal structures

**R C Ferreira[1], M B Vieira[2] and S O Dantas[3] and M Lobosco[2]**

[1] Instituto Federal Baiano, Campus Bom Jesus da Lapa, Brazil
[2] Department of Computer Science, UFJF, Juiz de Fora, Brazil
[3] Department of Physics, UFJF, Juiz de Fora, Brazil

E-mail: marcelo.lobosco@ice.ufjf.br

**Abstract.** The evolution of computers, more specifically regarding the increased storage and data processing capacity, allowed the construction of computational tools for the simulation of physical and chemical phenomena. Thus, practical experiments are being replaced, in some cases, by computational ones. In this context, we can highlight models used to simulate different phenomena on atomic scale. The construction of these simulators requires, by developers, the study and definition of accurate and reliable models. This complexity is often reflected in the construction of complex simulators, which simulate a limited group of structures. Such structures are sometimes expressed in a fixed manner using a limited set of geometric shapes. This work proposes a computational tool that aims to generate a set of crystal structures. The proposed tool consists of a) a programming language, which is used to describe the structures using for this purpose their characteristic functions and CSG (Constructive Solid Geometry) operators, and b) a compiler/interpreter that examines the source code written in the proposed language, and generates the objects accordingly. This tool enables the generation of an unrestricted number of structures, which can be incorporated in simulators such as the Monte Carlo Spin Engine, developed by our group at UFJF.

## 1. Introduction

The study of physical and chemical phenomena are always related to experimentation and observation. However, sometimes performing such experiments could not be trivial and/or be executed without major costs involved. In other situations, actual experimentation is impossible to be made, for example, in some situations involving the nanotechnology area. Thus, alternatives must be found. In this scenario, the use of software that simulates the phenomena under study can be very useful.

Simulators are built from using physical, chemical, mathematical and computational models, and allow the study of phenomena in a more practical way, since several hypotheses can be tested by the researcher by simply changing the parameters and constants used in the simulation. Also, detailed information about the experiments can be harvested and latter used for a better understanding of the phenomena been studied.

In this context, many simulators are built and deployed allowing simulations to be performed using a limited number of structures peculiar to a particular field of study, often restricting their use. The Monte Carlo Spin Engine (MCSE) [1], for example, allows the simulation of the interaction between spins organized in a three-dimensional way and formed by atoms with

magnetic properties. A set of basic shapes (spheres, cylinders and cubes), composed of equally spaced atoms, can be chosen for simulation. So, it precludes the simulation of a broader set of structures. The present work deals with this limitation, presenting a new programming language that allows the user to generate crystal structure using for this purpose implicit objects. However, although the language was conceived to be used with the MCSE, it should be stressed that it generates as output a generic structure that can be used by any simulator.

This paper is organized as follows. Section 2 presents the new language. Section 3 presents some related works while Section 4 concludes the work.

## 2. The new language

The new language has been proposed with the following objectives in mind: a) to allow users to describe a Bravais lattice; b) to allow users to define a solid based on the composition of other solids previously defined; c) to perform operations that transform a solid previously defined; d) after the compilation of its description, to allow the generation of a crystal structure. The proposed language is not ambiguous, that is, if a code was not modified, a new compilation of the same code will generate the same structure again.

Using the language, four steps must be followed to create a crystal structure:

- Creation of properties. This is the part of the code where the programmer defines properties for atoms or molecules that form the structure.
- Definition of characteristics. In this part of the code the programmer defines to which Bravais lattice the structure belongs to, which vectors form the lattice and which are the positions of each atom or molecule in the base.
- Description of the solid. The programmer defines the solids or compositions that will be used. A solid is described by an expression that defines their points. A composition is a piece of code where solids are aggregated.
- Construction of a structure. In this step the programmer defines the spatial region and the solids that will be inserted on it.

The language allows the definition of variables of two types: *Number* or *Solid*. A variable of type *Number* can store any value. A *Solid* variable is used to store a solid. The language provides logical, arithmetic and relational operators (such as ==, <>, <, >, &&, ||, +, −,...) and pre-defined functions. These functions are the typical mathematical ones (such as *sin*, *cos*, *tan*, *exp*, ...), CSG operators to manipulate a solid (*union*, *intersection*, $difference$ and *not* - negation), as well as functions to rotate, translate, scale and shear the solid. The *while* control structure is also available to be used by the programmer.

### *2.1. Example*

Due to the lack of space to present in details the language and their functions, an illustrative example was choosen to show how complex structures can be easily constructed with the use of the proposed language. The Listing 1 presents the code that defines a spherical shell perforated by eight spheres. This structure is filled with face-centered cubic cells. Figure 1 presents the result obtained from its compilation and visualization using our compiler. A point cloud is them generated by our tool. This point cloud can be incorporated in any simulator. Figure 2 presents its incorporation in the MCSE.

**Listing 1.** Code that defines a spherical shell perforated by eight spheres.

```
1cubic (face, (3.56, 0, 0), (0, 3.56, 0), (0, 0, 3.56))
2base (C1 , (0, 0, 0))
3base (C2 , (0.89, 0.89, 0.89))
```

```
4
5 solid MySphere (a, b, c ,r)
6   pow ((x-a), 2) + pow((y-b), 2) + pow((z-c), 2) <= pow(r, 2);
7
8 composite MySphericalShell (a, b, c, r1, r2)
9 begin
10   insert (difference(MySphere(a, b, c, r2), MySphere(a, b, c, r1)));
11 end
12
13 composite PerforatedShell (a ,b ,c , r1, r2)
14 begin
15   myShell = MySphericalShell (a, b, c, r1, r2) ;
16   myShell = difference (myShell , MySphere(a+r2, b+r2, c+r2, r2));
17   myShell = difference (myShell , MySphere(a+r2, b+r2, c-r2, r2));
18   myShell = difference (myShell , MySphere(a+r2, b-r2, c+r2, r2));
19   myShell = difference (myShell , MySphere(a+r2, b-r2, c-r2, r2));
20   myShell = difference (myShell , MySphere(a-r2, b+r2, c+r2, r2));
21   myShell = difference (myShell , MySphere(a-r2, b+r2, c-r2, r2));
22   myShell = difference (myShell , MySphere(a-r2, b-r2, c+r2, r2));
23   myShell = difference (myShell , MySphere(a-r2, b-r2, c-r2, r2));
24   insert (myShell);
25 end
26
27 lattice (-30, -30, -30, 30 , 30 ,30)
28
29 begin
30   insert (PerforatedShell (0, 0, 0, 20, 30), override);
31 end
```

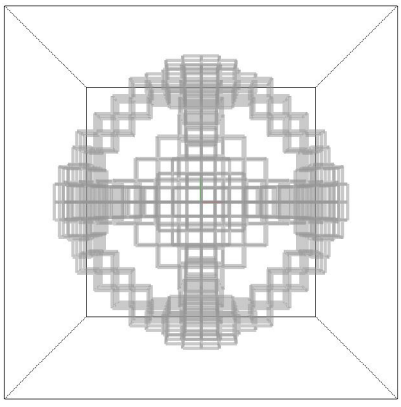

**Figure 1.** The visualization of the structure define by Listing 1.

## 3. Related Work

For the best of our knowledge, this is the first work that uses a programming language to describe crystal structures. However, lots of software [2, 3, 4, 5] are available to construct and visualize them using a graphical interface. As a general rule, in these software the user informs the Bravais lattice to be used, the atoms that compose the base and some parameters

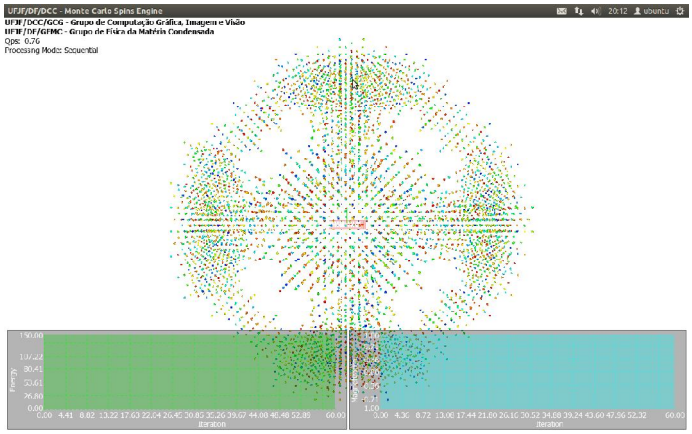


**Figure 2.** The structure generated by the language was read by the MCSE.

of the lattice. Some elements related to the crystallography, such as a crystallographic point group and the Miller indices, are present in some tools, as well as the definition of defects and impurities in the arrangements of the structure. Also some tools can simulate the X-ray diffraction phenomenon. The first version of the proposed language do not address these issues, although they can be incorporated in future versions. However, these simulators present some limitations. For example, they do not allow the user to define a spatial region where the structure must be inserted, as our proposed language does. Also, our proposed language allows the definition of monocrystalline and polycrystalline structures formed by equivalent unitary cells with distinct orientations while other tools only allow the definition of monocrystalline structures formed by only one unitary cell. Indeed, in these tools the generation of structures is based on the expansion of a unitary cell, while in our tool the generation is done filling a spatial region, defined by the user into his/her code, with unitary cells. Finally, the proposed tool allows the construction of structures of distinct shapes as well as it allows the definition of distinct properties to the elements that constitute then (atoms, ions or molecules). This characteristic was not found in other software.

## 4. Conclusion

This work presented a new computational tool to design and generate crystal structures. The tool consists of a programming language and a compiler that generates the objects accordingly. With the use of the proposed language, complex structures can be easily constructed using, for exemple, the composition and/or the transformation of objects previously defined.

## Acknowledgements

The financial support by FAPEMIG, CNPq, CAPES, and UFJF is greatly acknowledged.

## References

[1] Campos A M, Peçanha J P, Pampanelli P, de Almeida R B, Lobosco M, Vieira M B and De O Dantas S 2011 *Proceedings of the 2011 international conference on Computational science and its applications - Volume Part III* ICCSA'11 (Springer-Verlag) pp 654–667
[2] Shape software, atoms, on-line URL `http://www.shapesoftware.com`
[3] Ozawa T C and Kang S J 2004 *Journal of Applied Crystallography* **37** 679
[4] Crystalmaker software, crystalmaker, on-line URL `http://www.crystalmaker.com`
[5] Finger L W, Kroeker M and Toby B H 2007 *Journal of Applied Crystallography* **40** 188–192