NanoFabric Package

Y. Le Bouar,
LEM, UMR 104, CNRS/Onera, Châtillon, France,
yann.lebouar@onera.fr

February 16, 2014

Abstract

NanoFabric is a Fortran90 program developed for the geometrical construction of free or supported crystalline nanoclusters. Many degrees of freedom are provided to control the structure, shape, microstructure and orientation of the particle. The substrate may be crystalline or amorphous, and a physically based amorphous carbon is provided. The output files can be used in electron microscopy image simulation programs. The NanoFabric program is free, open source. Small improvements of the program can be done upon request (new structures, new output format, fixing bugs etc...). If you find it useful for you, please acknowledge the use of this program in your publications. Of course, the program comes with no warranty. It is absolutely necessary that you check with your favorite visualization tool that the generated object is correct.

1 Structure of the program

This program was designed during the course of a study on the thermodynamic and kinetic behavior of CoPt nanoalloys performed in collaboration with the MPQ laboratory (Paris Diderot) (see e.g. Refs [1, 2, 3]). The NanoFabric program provides a geometrical construction of free standing as well as supported crystalline nanoparticles. Many degrees of freedom are provided to control the structure, shape, microstructure and orientation of the particle. The substrate may be crystalline or amorphous, and a physically based amorphous carbon is provided (see Ref. [4] for details about this substrate). Output files are classical atom files (.xyz, .pdf) but the program also cut the nano-object into slices in order to be directly used by multislice programs such as JEMS (see http://cimewww.epfl.ch/people/stadelmann/home.htm) to simulate the contrast of transmission electron microscopy images.

The construction of the supported nano-object is divided in five optional steps. The first two steps correspond to the creation of crystalline objects. The third step is a way to have a segregated layer on the particle. The fourth step is a possible rotation of the particle. The last step is either the creation of an amorphous substrate or the reading of an external substrate/object. These steps are detailed in the following sections.

1.1 Step 1: First crystalline nano-object

1.1.1 Atoms

The constructed crystalline object may contain up to two different types of atom. Any type of atom can be used and is defined by the two characters of its chemical symbol. A space must be placed afted a symbol containing only one character. For example 'Au' for gold and 'C ' for carbon.

Note that several printouts of the program are not ready for all atom types, *but the output files will be correct whatever the atom type.* The printouts are ready for the following atoms: H, Li, C, N, O, Na, Mg, Al, Si, Cl, Sc, Ti, V, Cr, Fe, Co, Ni, Cu, Zn, Ga, Ge, As, Zr, Rh, Pd, Ag, Pt, Au.

1.1.2 Structure

The program creates crystalline nanoparticles of different structures. In version 2.2 the following structures are available: fcc, GaAs, $L1_0$, $L1_2$, TiO_2 but other structures can be easily added if necessary. On the atomic sites of the structures, two types of atom (A and B) can be placed.

For example:

- structure = 'fcc', A='Au', B='Au', c_B=0. gives fcc gold structure
- structure = 'GaAs', A='Si', B='Si', c_B=0. gives the silicium structure
- structure = 'fcc', A='Au', B='Cu', c_B=0.5 gives the equiatomic fcc solid solution

When considering ordered structure, an order parameter is necessary to fully describe the structure. This order parameter is assumed scalar and lies between 0 and 1.

- structure = 'L12', A='Sc', B='Al', c_B=0.75, order p. = 1.0 gives the fully ordered Al_3Sc $L1_2$ structure
- structure = 'L12', A='Sc', B='Al', c_B=0.75, order p. = 0.8 gives a partially ordered Al_3Sc $L1_2$ structure
- structure = 'L10, A='Co', B='Pt', c_B=0.45, order p. = 0.7 gives a partially ordered CoPt $L1_0$ structure at composition 0.45.

When atomic sites are partially occupied by several types of atom, it is either possible to select randomly the type of atom at a given site (with a probability equal to the fraction of occupation) or to put all possible atoms with their occupation probabilities. The first case (occupation probabilities='random') is a snapshot of a possible microstructure whereas the second is a structure defined on average (occupation probabilities='average').

The crystalline structure can be rotated before the definition of the particle shape (see 'basis change before cutting'). This can be done by selecting the zone axis vector (in units of the unit cell vectors). It is also required to provide another vector (which must not be collinear to the fist one). The projection of this vector in a plane normal to the zone axis will be the x axis. For example, to have a fcc structure in [111] zone axis, you can write in the input file:
' Basis change before cutting the particule y/n, axe 3, axe 1' 'y'   1. 1. 1. 1. -1. 0.
Warning: this option is not currently compatible with the shape 'polyhedron'.

1.1.3 Shape

The program constructs all atom positions by the repetition of the crystal unit cell and then exclude all atoms that are not located inside a given shape. Several shapes are available: ellipsoid, hollow_ellipsoid, polyhedron.

- 'ellipsoid': gives an ellipsoid where the principal axis are along the box axis. The three semi-principal axes can be independently specified.
- 'hollow_ellipsoid': gives a hollow ellipsoid where the principal axis are along the box axis. The three semi-principal axes can be independently specified for the external and internal surfaces.
- 'polyhedron': the nanoparticle is defined by computing the vector $\vec{X}$ between the center of the box and the atom position and by projecting this vector along a given direction $\vec{d}$. If $\vec{d} \cdot \vec{X}$ is above a given threshold the atom is removed. The available directions $\vec{d}$ are $\langle 100 \rangle$, $\langle 111 \rangle$ and $\langle 110 \rangle$, where the indices refer to the vector defining the crystal unit cell. Cuts along several directions can be done simultaneously to obtain truncated shapes.

For example:

- Sphere R= 2nm: shape = 'ellipsoid', parameters = 20. 20. 20. 0. 0. 0. (the three last parameters are not used here)
- Oblate spheroid D= 3nm, h=2nm: shape = 'ellipsoid', parameters = 15. 15. 10. 0. 0. 0. (the three last parameters are not used here)
- Sphere with an empty core: shape = 'hollow_ellipsoid', parameters = 10. 10. 10. 20. 20. 20.

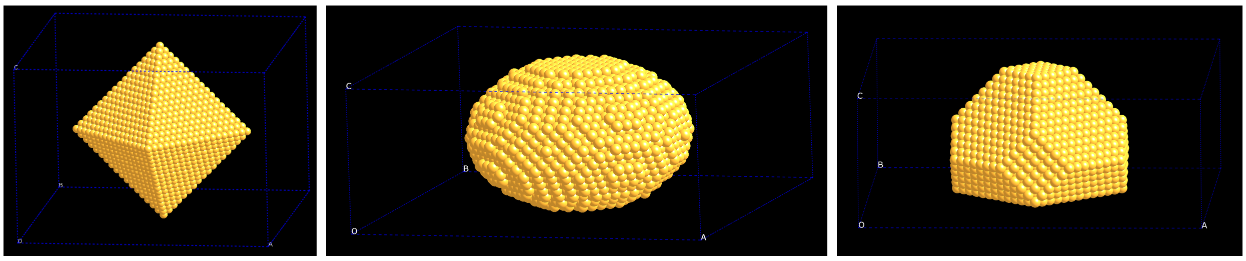

Figure 1: *From left to right: Gold octahedron; Prolate spheroid; Pretty shape.*

- Octahedron: (assuming a cubic structure) shape = 'polyhedron'
  cut along -/+[111],-/+[-111],-/+[1-11],-/+[11-1] ', 'y', 20., 20., 20., 20., 20., 20., 20., 20.
  The cut along other directions must be set to 'n'.
- Truncated octahedron: (assuming a cubic structure) shape = 'polyhedron'
  cut along -[100],[100],-[010],[010] -[001],[001] ', 'y', 22., 22., 22., 22., 22., 22.
  cut along -/+[111],-/+[-111],-/+[1-11],-/+[11-1] ', 'y', 20., 20., 20., 20., 20., 20., 20., 20.
  The cut along other directions must be set to 'n'.

Several shapes are displayed in Fig. 1.

Note that the center of the shape can be selected at any position inside the unit cell. For example, the following choice set the center of the unit cell as the center of the shape. ' Shape center with respect to the crystalline structure (relatives units) ' 0.5 0.5 0.5

Finally, the shape can be translated along the z direction. This is sometimes useful to ensure that the shape fits inside the box. For example:
' Translation along z of the particle center with respect to the box center (in Angstrom) ' -6.

1.2 Step 2: Second crystalline nano-object

The creation of the second crystalline object follows exactly the same lines as the ones presented for the first object. The only difference is the definition of a minimal distance between one atom of the first object and one atom of the second object. If atoms of the second object are closer (to an atom of the first object) than this distance, this atom is removed. This is useful to discard unphysicaly close atoms when combining two objects. For example:
' Minimal allowed distance between atoms when adding object 1 and 2 (en Angstrom) ', 1.8

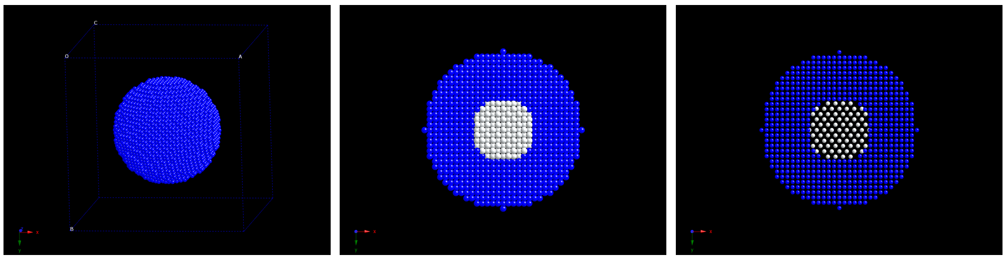

Figure 2: *AgCu core shell particle. Left: 3D view. Center: slice of a coherent particle. Right: slice of an incoherent particle.*

This option was initially designed to arbitrary select the crystal structure inside a shell using the shape 'hollow_sphere' (see Fig. 2). But this step may also be used to define a cristalline substrate using the 'polyhedron' cut along z (see Fig. 3). For example:
' If polyhedron : along -[100],[100],-[010],[010] -[001],[001] ', 'y', 1000., 1000., 1000., 1000., -20., 40.

The cut distances along ±[100] and ±[010] are set to a high value (here 1000.) to ensure that the cut is only performed along z.

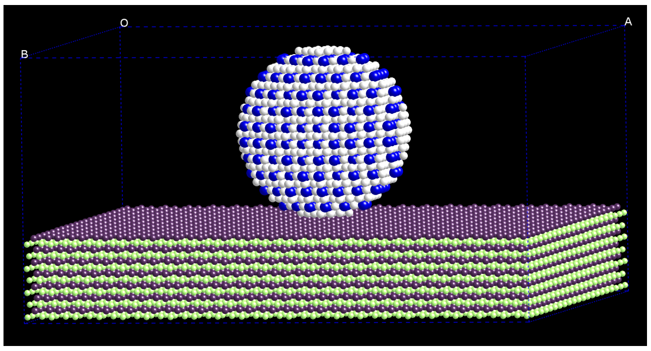

Figure 3: *CoPt nanoparticle with the $L1_2$ structure on a GaAs crystalline substrate*

1.3 Step 3: Segregation

This option is a convenient way to study the consequences of surface segregation on TEM images. The program detects atoms that are on the surface and changes their type to the one selected. Obviously, this does not conserve the overall composition of the object. An example is presented in Fig. 4. More precisely,

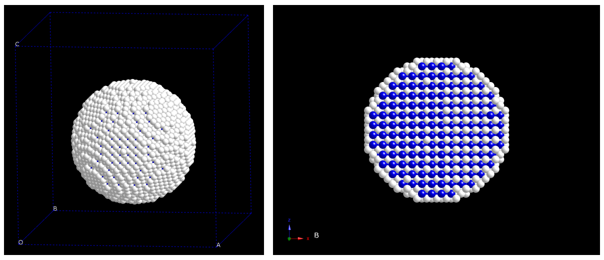

Figure 4: *From left to right: CoPt $L1_0$ spherical particle with a surface layer of platinum. Left: 3D view. Right: slice of the particle.*

a surface atom is defined as an atom that has less neighbors than the reference coordination number. A neighbor, is an atom that is closer (to the considered atom) than a given 'capture length'. This capture length must be selected between the first and second nearest neighbor distance.

The segregation can be extended to several layer. The second layer is defined as the surface of the object when the first layer is removed. An so on ...

When considering a pure particle, this option can also be used to change the absorption and Debye-Waller parameters for the surface atoms.

1.4 Step 4: Object rotation

Once the object is created using steps 1, 2 and 3, the object can be globally rotated. The rotation is definedusing a vector of the crystal unit cell of the first object. The object is rotated to have the zone axis of the structure of the first object (step 1) along the z direction. As before, a second vector is needed to define the x direction.

The example, presented in Fig. 5, is a rotation of the fcc object to have (111) surface normal to z.

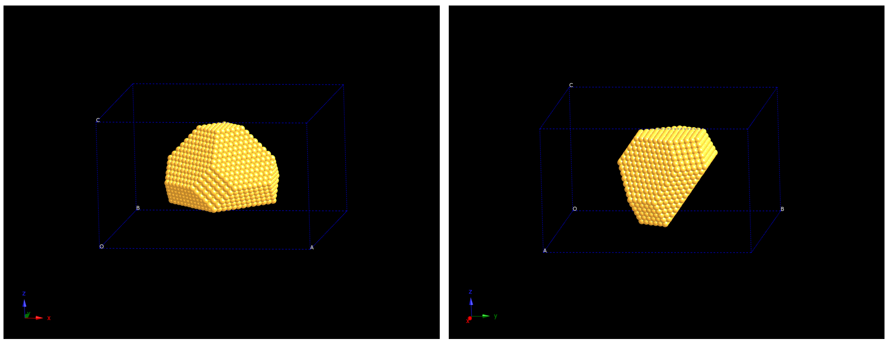

Figure 5: *Truncated Au particle before rotation ((001) surface along z) and after rotation ((111) surface along z).*

1.5 Step 5: Substrate and external objects

The program is able to create a random amorphous substrate or to read an external substrate or objet.

1.5.1 Amorphous random substrate

The program is able to create an amorphous substrate using a random procedure. The atom type, density and thickness can be selected for the substrate. The program tries to insert new atoms inside the layer by selecting random new position for the atom. The new atom is removed if it is closer than a minimal distance to other existing substrate atoms. The procedure is iterated until the target density is reached. For example, to have an amorphous random substrate of carbon with a thickness of 5nm, the input file

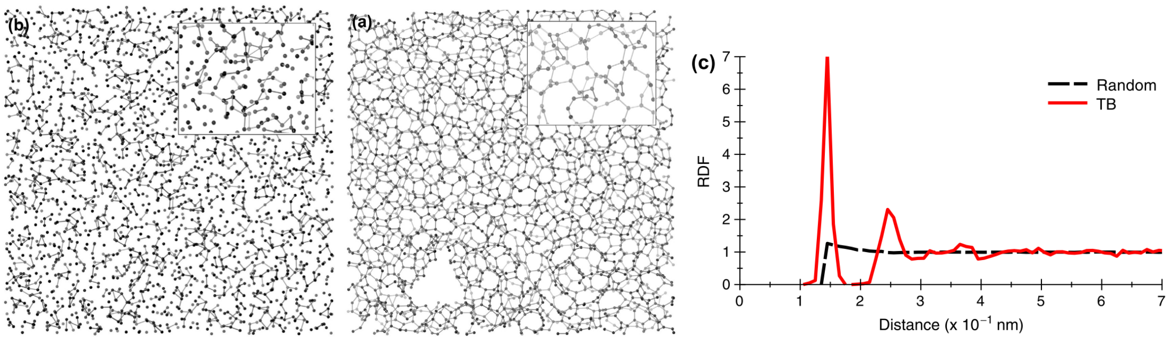

Figure 6: *3D perspective view of the carbon generated by: (a) tight-binging model and (b) random model. The thickness of the box corresponding to these two projected structures is 1 nm. (c) 3D radial distribution functions (RDF) of carbon atomic positions generated by tight-binding (red) and random (black) models.*

must contain the following lines:
' Adding a substrate ? (y/n/r) : name file if read', 'y' 'toto' <- name file not relevant here
' Atom type (used unless substrate is read): ', 'C '
' Average atomic volume (Angtrom^3), minimal distance between atoms ', 9.084 1.4
' Substrate thickness (Angstrom) ', 50.

1.5.2 Physically based amorphous carbon substrate

The program is able to create an amorphous layer by reading an amorphous super-cell created using a Tight Binding scheme. The structural differences between a random carbon and a more physically-based one is presented in Fig. 6. The consequences of the amorphous carbon choice on the TEM images can be found in Ref. [4]. A possible super-cell is provided here with the name 'AMO13760_13760at_5nm5nm5nm.xyz'.

It is important that you enter the correct average atomic volume in the input file, i.e. the one corresponding to the input file. If a wrong value is given, the program stops and gives you the correct value. You just have to copy this value in the input file an re-run the program again.

Two Euler angles are provided to rotate the structure before cutting the requested substrate thickness. For an amorphous super-cell, changing these angle is a way to get different substrate layers from the same super-cell.

For example, to have a physically-based amorphous random substrate of carbon with a thickness of 5nm, the input file must contain the following lines:
' Adding a substrate ? (y/n/r) : name file ', 'r' './AMO13760_13760at_5nm5nm5nm.xyz'
' If read: theta and phi angles to rotate the unit cell (in degrees) ', 10. 10.
' Average atomic volume (Angtrom^3), minimal distance between atoms ', 9.084 1.4
' Substrate thickness (Angstrom) ', 50.

Note that the input supercell may contain an arbitrary number of atom types.

1.5.3 Crystalline substrate

As discussed above, a way to create a crystalline substrate is to use the second object (see step 2). The option initially designed to read an amorphous substrate provides another way to create a crystalline substrate. The crystalline structure must then be entered in a file and read in step 5. The crystal structure can be rotated using the Euler angles. This option is convenient when the crystal structure is not implemented in the program and when more than two types of atoms are inside the substrate. For example, to create

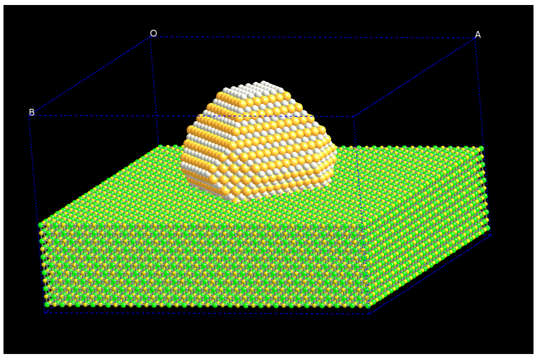

Figure 7: *AuPd $L1_0$ facetted particle on NaCl*

a NaCl substrate with a (001) surface (see Fig. 7) the input file must be as follows:
' Adding a substrate ? (y/n/r) : name file if read ', 'r' './NaCl/NaCl.xyz'
' If read: theta and phi angles to rotate the unit cell (in degrees) ', 0. 0.
' Average atomic volume (Angtrom^3), minimal distance between atoms ', 22.42815 1.4
' Substrate thickness (Angstrom) ', 50.

The file containing the unit cell of the substrate is a classical '.xyz' format but a header must be added at the beginning of the file. More precisely, the file must be formatted as below:

```
  8                                   ! First ligne with 8
                                      ! Free line for comments
H 0.00000 0.00000 0.00000             ! The 8 following lines define the vertices of the box
H 100.00000 0.00000 0.00000
H 0.00000 100.00000 0.00000
H 100.00000 100.00000 0.00000
H 0.00000 0.00000 194.44201
```

```
H 100.00000 0.00000 194.44201
H 0.00000 100.00000 194.44201
H 100.00000 100.00000 194.44201
8452                                 ! Number of atoms
                                     ! Free line for comments
C 17.9936 0.0000 0.0000              ! From this line on, the atom positions are listed
C 17.9378 1.4164 0.0772              ! Atom position should be inside the box !
...
```

1.5.4 Combining nanoparticles with other objects

The program is able to read an object and combine it with a nanoparticle. The file format for the object is the same as the one detailed in Sec. 1.5.3. An example is shown in the Fig. 8 where a gold particle has been put inside a carbon nanotube.

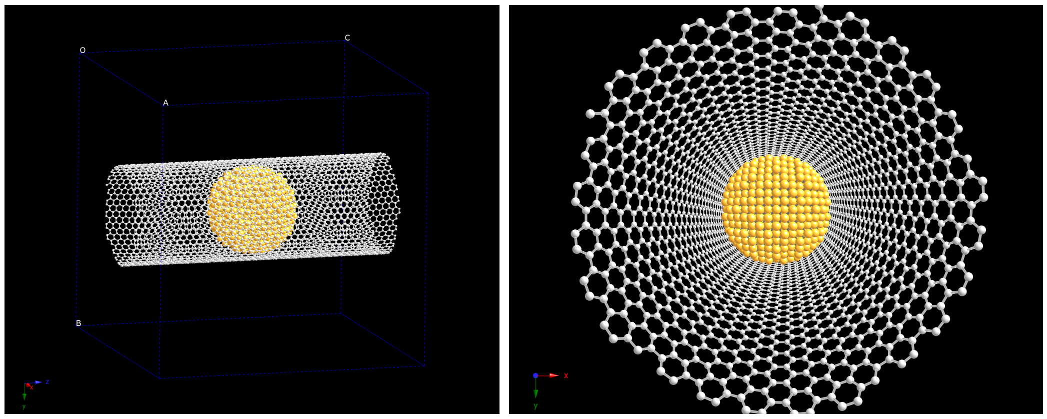

Figure 8: *Gold nanoparticle inside a nanotube.*

1.6 Last section of the input file

The last section of the input file provides several options:

- The seed of the random generator can be changed. This is useful to get several sets of atom positions in the case of partially ordered particles or for generating random substrates. The seed is an integer between 0 and 2^{31}.
- The box may be cut into slices of the same thickness. The number of slices is requested by the program.
- The generic name for the output files can be specified.
- A peeling of the object can be performed, layer by layer, to analyze the number of atoms and the atom types in each layer. A layer is defined exactly the same way as for the segregation (see Sec. 1.3).
- A random displacement of the atoms can be added to the atom positions.
- The first object can be removed. This is necessary to study the substrate alone.

2 Installation and execution

2.1 Installation

The NanoFabric program is written in Fortran 90. It does not require specific libraries, but you need to have a fortran compiler (fortran 90 or above). Free compilers are available such as gfortran (http://gcc.gnu.org/wiki/GFortran). I recommend to use the double precision option during compiling. I have not checked that everything is correct without it.

Using gfortran, the compiling command is:

```
gfortran NanoFabric.f90 -fdefault-real-8 -o NanoFabric.exe
```

2.2 Execution

To run the program, the executable file (NanoFabric.exe) as well as the input file (input_NanoFabric.txt) must be placed in the same directory. Then, you just have to type the command: ./NanoFabric.exe

During execution, the program displays a few pieces of information. The most important lines are the last ones in which a 'summary of the atomic configuration is given'. The total number of atoms in the box is given as well as the characteristics of each atom type (occupation number, min/max positions, Debye Waller, absorption). It is always a good idea to carefully read these lines.

2.3 Bench tests and examples

The program execution should not take more than few seconds, except for the case of random amorphous substrate with a large density. Several examples are provided to help you understand how the program can be used. Small images are in Fig. 9, and the list of the worked example is detailed below.

1. Spherical fcc nanoparticle of pure Pt with a diameter of 2nm.
2. Disordered CoPt particle on a fcc lattice
3. CoPt spherical nanoparticle surrounded by a copper shell
4. CoPt spherical nanoparticle on a crystalline GaAs substrate
5. ZrSc3 spherical nanoparticle with a $L1_2$ structure and a segregation of oxygen
6. Al3Zr spherical nanoparticle with a $L1_2$ structure on a random amorphous substrate
7. Al3Zr spherical nanoparticle with a $L1_2$ structure on an amorphous substrate (obtained from Tight Binding calculations).
8. An amorphous carbon layer
9. A gold nano-tetrahedron
10. A gold oblate spheroid
11. A gold truncated shape
12. A CoPt nanoparticle with an external layer of Pt
13. A coherent core-shell particle
14. An incoherent core-shell particle
15. A rotated gold particle
16. AuPd $L1_0$ particle on a NaCl substrate

2.4 Box size: Comments for multislice applications

When creating the nano-object, it is important to keep in mind that the choice of the box size will influence the multislice calculation results. Let (B_x, B_y, B_z) be the box sizes where z denotes the direction of the electron beam.

- When considering a crystalline particle, the convergence of the calculation is improved if the box size is a multiple of the unit cell structure. Indeed, this condition ensures that the set of lattice vectors used for the multislice calculation contains the reciprocal lattice of the structure.

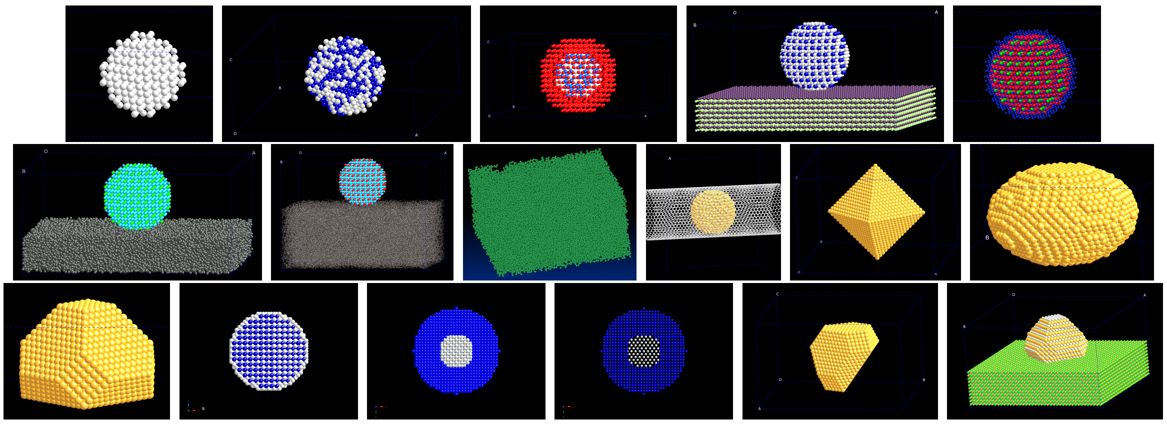

Figure 9: *Images from the worked examples*

- To ensure a correct description of the nanoparticle shape, the box sizes in the x and y directions must be chosen large enough. A rule of thumb is to chose B_x and B_y at least three times larger than the particle size. Indeed, because of the finite size D of the particle, the k vectors of the reciprocal lattice (of the particle structure) have a width of the order of $2\pi/D$. The extension of these peaks is related to the Fourier transform of the particle shape function. To correctly describe these peaks, the shortest k vectors included in the simulation (which are equal to $2\pi/L$ where L is the box size) have to be significantly shorter than $2\pi/D$.
- The box size B_z must be obviously selected as a multiple of the thickness of the slices that you want.

References

[1] D. Alloyeau, C. Ricolleau, T. Oikawa, C. Langlois, Y. Le Bouar, and A. Loiseau. Comparing electron tomography and hrtem slicing methods as tools to measure the thickness of nanoparticles. *Ultramicroscopy*, 109:788–796, 2009.

[2] D. Alloyeau, C. Ricolleau, C. Mottet, T. Oikawa, C. Langlois, Y. Le Bouar, N. Braidy, and A. Loiseau. Size and shape effects on the order-disorder phase transition in copt nanoparticles. *Nature Materials*, 8:940–946, 2009.

[3] D. Alloyeau, T. Oikawa, J. Nelayah, G. Wang, and C. Ricolleau. Following ostwald ripening in nanoalloys by high-resolution imaging with single-atom chemical sensitivity. *Appl. Phys. Lett.*, 101, 2012.

[4] C. Ricolleau, Y. Le Bouar, H. Amara, O. Landon-Cardinal, and D. Alloyeau. Random vs realistic amorphous carbon models for high resolution microscopy and electron diffraction. *J. Appl. Phys.*, 114(213504), 2013.