Visualization of (single- & multi-wall) carbon nanotubes

A short tutorial to the POV-Ray file.

Florian Simon <florian.simon@aol.com>

August 9, 2012
Saito Laboratory, Faculty of Science, Tōhoku University, Sendai

This is a short tutorial to the POV-Ray-file I created during my time in Saito-Laboratory. It shows how you can get BMP-pictures and GIF-animations of any (n, m)-carbon-nanotube (single- & multi-wall) by using the chiral vector $C_h = (n, m)$.

Contents

1 Requirements

In order to visualize carbon nanotubes on the computer you need to have the following things:

1. POV-Ray (Needed to create the pictures)
2. GIAM (Used to create GIF-animation of the pictures created with POV-Ray)
3. CreateNanotube folder
4. nanotube.inc file

You can download POV-Ray from the official website[1] for free. Do the same with GIAM[2]. Then install both programs on your computer.

Then download and extract "CreateNanotube.rar" from my webpage and save the "CreateNanotube"-folder wherever you like and put "nanotube.inc" into the "include"-folder of POV-Ray (usually *documents/POV-Ray/v3.6/include*).

And that is already everything! Now you can visualize any kinds of chiral nanotubes!

2 How does POV-Ray work?

2.1 POV-Ray

Now open first POV-Ray and open then "CreateNanotube.pov" or click on this file directly. Now you should see a window like seen in Fig. 1.

In the upper left corner you can change the resolution of the picture that is going to be rendered (also Anti-Aliasing is possible for better quality) and by clicking "Run" the program will create the picture and save it into the "CreateNanotube"-folder.

2.2 CreateNanotube.pov

First of all, text that stands behind "//" is not read by POV-Ray (comments). This is useful to outline and explain things directly in the program. This will be seen as green text in POV-Ray.

The file itself is divided into four parts.

1. head
2. explanation of the file
3. textures, properties, variables and other settings
4. the objects (nanotubes; determined by the upper settings)

[1] `http://www.povray.org/download/`

[2] `http://www.homepage3.nifty.com/furumizu/giamd.htm/`

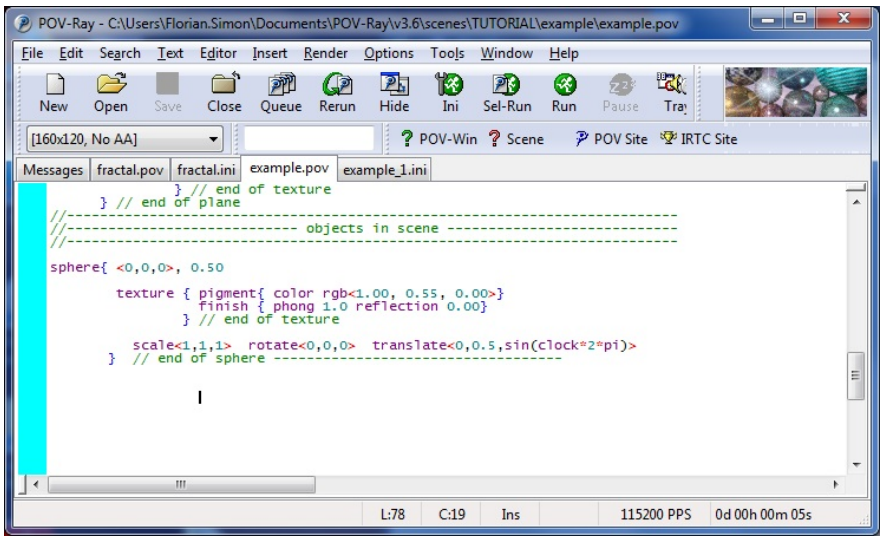

Figure 1: A basic POV-Ray window.

You can ignore the first part. You will create nanotubes by adjusting textures, properties, variables, etc. in this file and write then the object itself below those settings.

Now the question arises, *what exactly* can we create now? Section 3 illustrates the macros (templates) which can be used here:

- xtube & ntube
- fxtube & fntube
- vxtube & vntube

To create a picture of the (n, m)-nanotube, you have to enter/change the variables as you wish and then put the upper mentioned templates at the end of the file. When clicking "Run", POV-Ray will render the picture and save in the same folder where the file is located. Please see also Section 5 for further information.

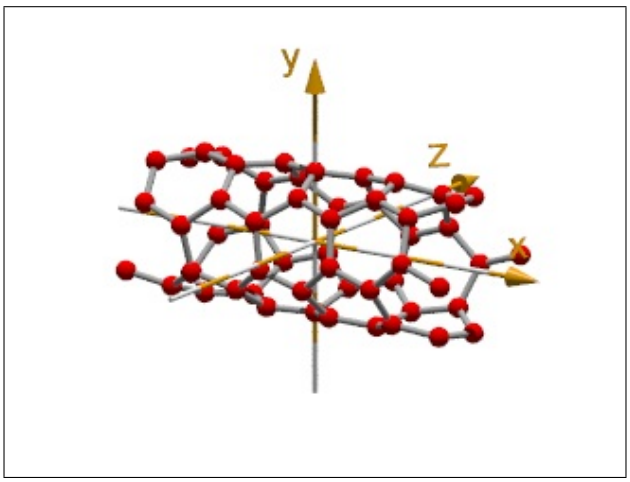

Figure 2: Positioning of the nanotube in relation to the coordinate-axes.

In Fig. 2 you can see how the resulting nanotube will be located in relation to the coordinate-axes. The camera will be located on −15 on the Z-axis (<0,0,-15>).

2.3 Animation (GIAM)

In order to create the pictures for the animation open the INI-file "CreateNanotube.ini" with POV-Ray (in the same folder as "CreateNanotube.pov").

```
; POV-Ray animation ini file
Antialias=on
Antialias_Threshold=0.1
Antialias_Depth=2

Input_File_Name="CreateNanotube.pov"

Initial_Frame=1
Final_Frame=50
Initial_Clock=0
Final_Clock=1

Cyclic_Animation=off
```

Here you can change the number of pictures POV-Ray should render for you. Just change the number after "Final_Frame=" to the number of pictures you want. More pictures result in the final animation in smoother movements. Leave everything else like it is.

If you click now "Run" in the toolbar, POV-Ray will render the pictures we need for the animation (and save it in the same folder). (This is only needed for fxtube, fntube, vxtube and vntube.)

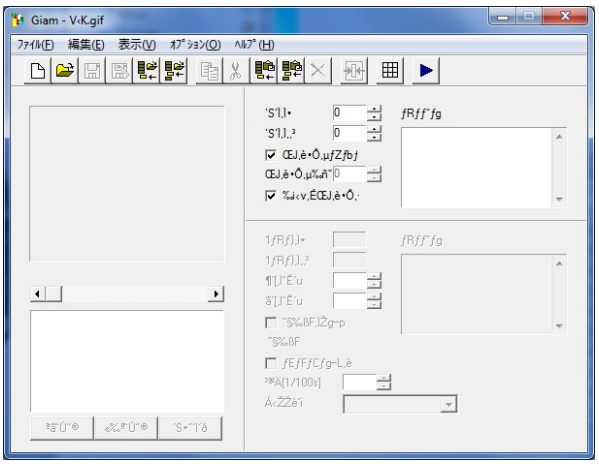

Figure 3: A basic GIAM window.

GIAM is used to combine the pictures created with POV-Ray to a GIF-animation. Just open GIAM and drag all pictures into the big white box and click the save symbol. And that's already everything.

3 Nanotube-templates (macros)

3.1 xtube & ntube

This template will create the unit cell of a nanotube either in plane- or tube-shape. You can also decide if you want to connect the atoms through bonds.

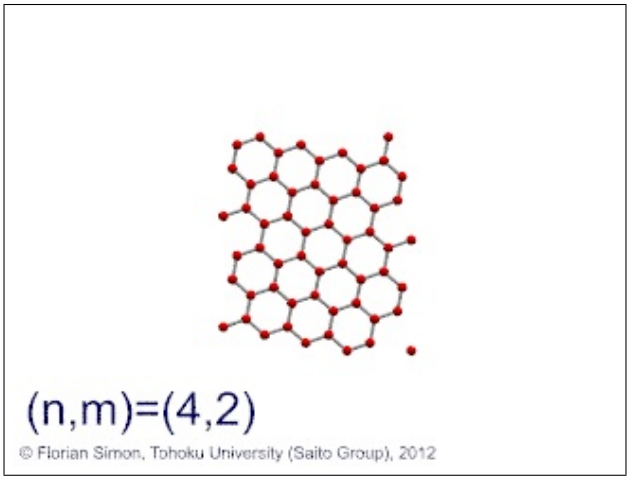

Figure 4: (4,2)-nanotube in plane-shape with bonds.

Differences between xtube & ntube:

- ntube(n,m,g,xx,s) creates nanotube that is centred around <0,0,0>.

- xtube(n,m,g,xx,s) creates nanotube that begins at <0,0,0>.

The variables are changed in the already existing section in the POV-Ray file (just enter new numbers; now writing of program code is necessary). Possible values are:

- "n" and "m" are values for chiral vector (0<|m|<=n)
- "s" defines shape of unit cell
- "g" stands for grid reap. bonds
  - g = 0 (no grid)
  - g = 1 (grid)
- "xx" stands for the number of unit cells
  - s = 1 (plane)
  - s = 2 (cylinder)

In Fig. 4 you can see the result when you put in the following code in the file and click on "Run".

```
#declare n = 4;
#declare m = 2;
#declare g = 1;
#declare xx = 1;
#declare s = 1;

object { ntube(n,m,g,xx,s) scale 1.0 rotate <0,0,0> translate <0,0,0> }
```

The following codes both also creates the same picture (but it's less clear):

```
object { ntube(4,2,1,1,1) scale 1.0 rotate <0,0,0> translate <0,0,0> }
object { ntube(4,2,g,1,1) }
```

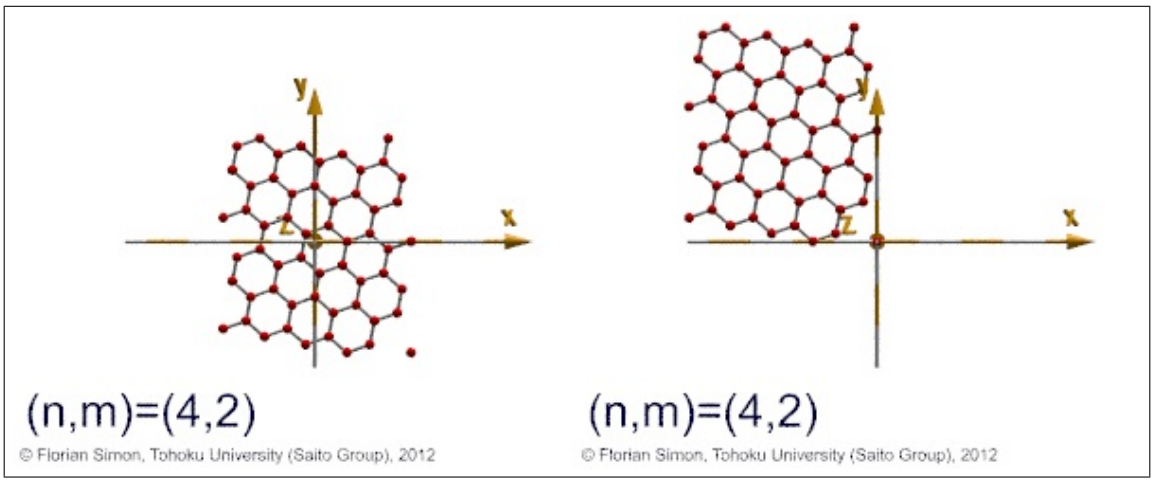

Figure 5: Positioning difference of ntube(...) & xtube(...) (plane-coordinates).

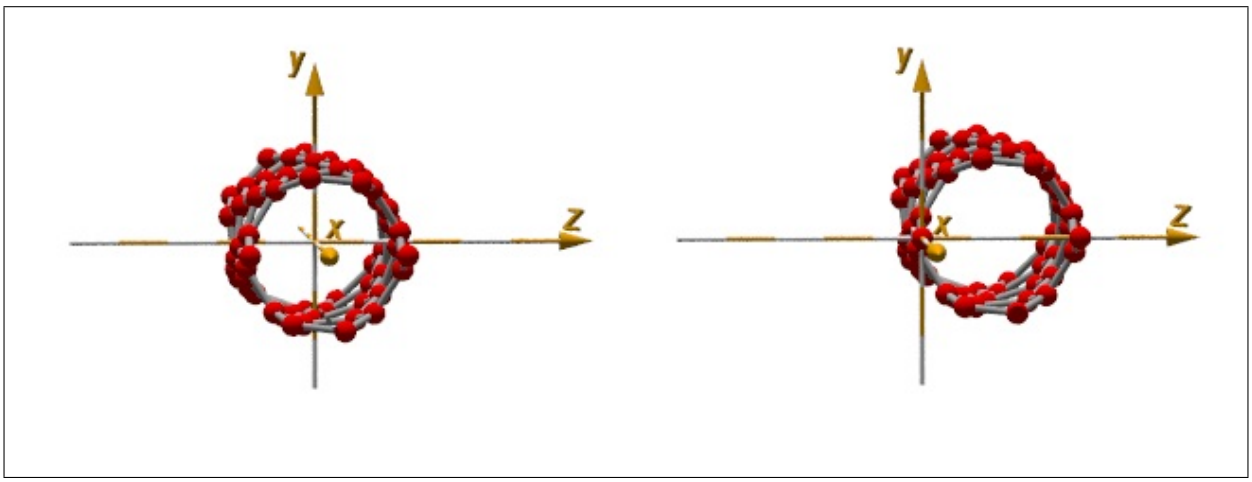

Figure 6: Positioning difference of ntube(...) & xtube(...) (tube-coordinates).

3.2 fxtube & fntube

This template will create a *folding* unit cell of a nanotube. Differences between fxtube & fntube:

- fntube(n,m,g,xx,dir) creates *folding* nanotube that is centred around <0,0,0>.
- fxtube(n,m,g,xx,dir) creates *folding* nanotube that begins at <0,0,0>.

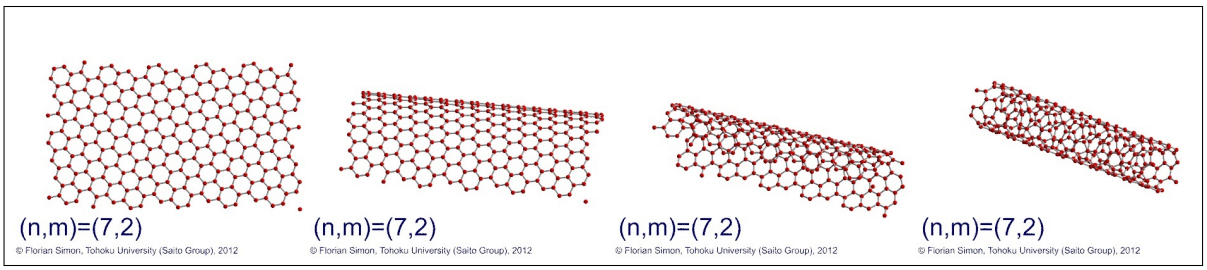

Figure 7: Screenshots of a $(7, 2)$-nanotube during the folding process.

The variables are also changed in the already existing section in the POV-Ray file (just enter new numbers; now writing of program code is necessary). Possible values are:

- "n" and "m" are values for chiral vector $(0 < |m| <= n)$
- "g" stands for grid reap. bonds
  - g = 0 (no grid)
  - g = 1 (grid)
- "xx" stands for the number of unit cells

- "dir" stands for direction of the folding process
  - dir = 1 (from plane to tube)
  - dir = 2 (from tube to plane)

3.3 vxtube & vntube

This template will create a *vibrating* unit cell of a nanotube. Differences between vxtube & vntube:

- vxtube (n,m,g,xx,s,AcoVibA,AcoVibB,AcoVibAZ,AcoVibBZ,OptVibMode,WaveLength) creates *vibrating* nanotube that begins at <0,0,0>.
- vntube (n,m,g,xx,s,AcoVibA,AcoVibB,AcoVibAZ,AcoVibBZ,OptVibMode,WaveLength) creates *vibrating* nanotube that is centered around <0,0,0>.

The variables are changed in the already existing section in the POV-Ray file (just enter new numbers; now writing of program code is necessary). Possible values are:

- "n", "m", "g", "xx", "s" are exactly like in the templates before.
- "AcoVibA" & "AcoVibB" are the Acoustic Vibration Modes of "A" reap. "B" atoms. Possible values ranges from $+-1$ to $+-4$ (0 stands for no vibration).

  (!) Please take a look at file itself. Explanation for these values is given there.
- "AcoVibAZ" and "AcoVibBZ" lets "A" reap. "B" atoms vibrate in normal direction (Z-Axis in planeshape, R in cylindershape) of the tube. Possible values are 0, -1 and 1.

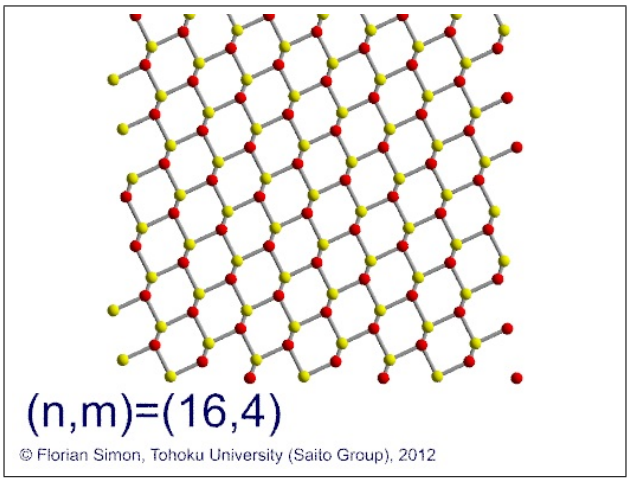

Figure 8: (16,4)-unit-cell (plane-shape) vibrating in acoustic mode (AcoVibA, AcoVibB).

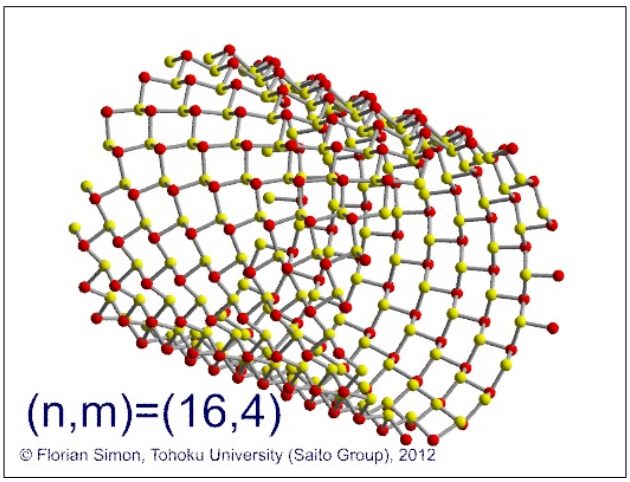

Figure 9: (16,4)-unit-cell-nanotube vibrating in acoustic mode (AcoVibAZ, AcoVibBZ).

- "OptVibMode" stands for the Optical Vibration Mode
  - 0 (no vibration)
  - 1 (longitudinal wave, also known as l-wave or compression wave (standing))
  - 2 (longitudinal wave, also known as l-wave or compression wave (moving))
  - 3 (transverse wave (standing))
  - 4 (transverse wave (moving))
- "WaveLength" determines the Wavelength as a factor of T-vector-length (unit cell) (needed only for Optical Vibration Mode).
  - (example) WaveLength = 2 –> Wavelength will be 2*(Length of T-vector)

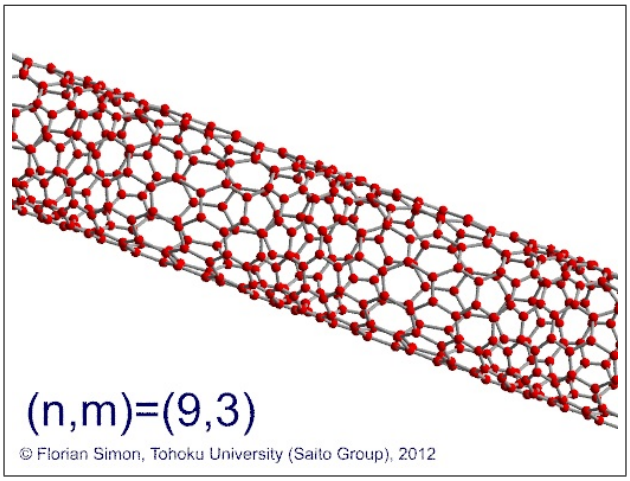

Figure 10: (9,3)-nanotube vibrating in optic mode (OptiVibMode(=1 resp. 2)).

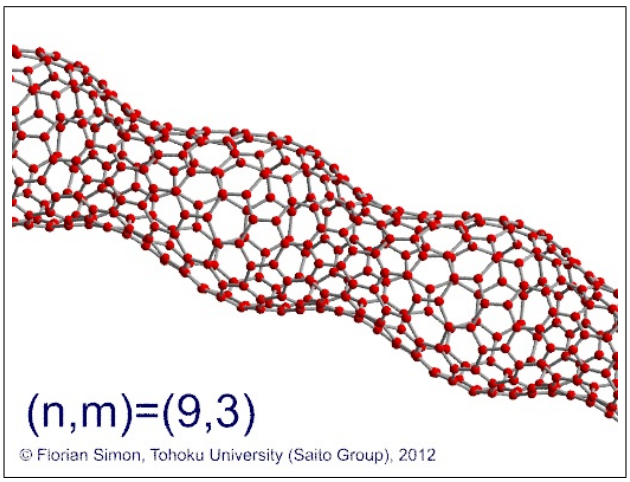

Figure 11: (9,3)-nanotube vibrating in optic mode (OptiVibMode(=3 resp. 4)).

4 Textures and properties

To change size and colors of atoms and bonds, as well as the amplitude of the upper mentioned vibration modes, just change the values of the following section in the POV-Ray-file.

```
//-------------------------------------------------------------
// textures, properties and vibration (amplitude) settings
//-------------------------------------------------------------
//
   #declare Atex = pigment{color rgb<1,0,0>} ;
   #declare Btex = pigment{color rgb<1,0,0>} ;
   #declare Botex = pigment{color rgb<0.5,0.5,0.5>} ;
//
   #declare Ar   = 0.3 ;
   #declare BondR = Ar/3 ;
   #declare a    = 2.49 ;
//
   #declare AcoAmp = a/10 ;
   #declare OptAmpNTA = AcoAmp/4 ;
   #declare OptAmpZ = AcoAmp*3 ;
//
//-------------------------------------------------------------
```

Atex	atom texture (A) <red,green,blue>
Btex	atom texture (B)
Botex	bond texture
Ar	atom radius
BondR	bond radius
a	length of unit vector (2.49 Å)
AcoAmp	amplitude of Acoustic Vibration
OptAmpNTA	amplitude of Optic Vibration (along NanoTubeAxis)
OptAmpZ	amplitude of Optic Vibration (along Normal Axis)

Table 1: Description for texture-, properties- and amplitude-setting-commands.

5 Useful knowledge about this file

I've put down all the possible templates right at the end of the file as a comment. Just remove the "//" in front of the *object {...}* and POV-Ray will read it. You can also write the line again and change the parameters inside (e.g. for creating Multi wall carbon nanotube).

```
1 // object { ntube(n,m,g,xx,s) scale 1.0 rotate <0,0,0> translate <0,0,0> }
```

- *scale* is for changing the size of the tube.
- *rotate* rotates the whole structure around <x,y,z> axis (in degree). *Not* around the structure itself!
- *translate* moves the whole structure along <x,y,z> vector.

It's also possible to put more than just one *rotate* or *translate* command in a row.

6 Examples

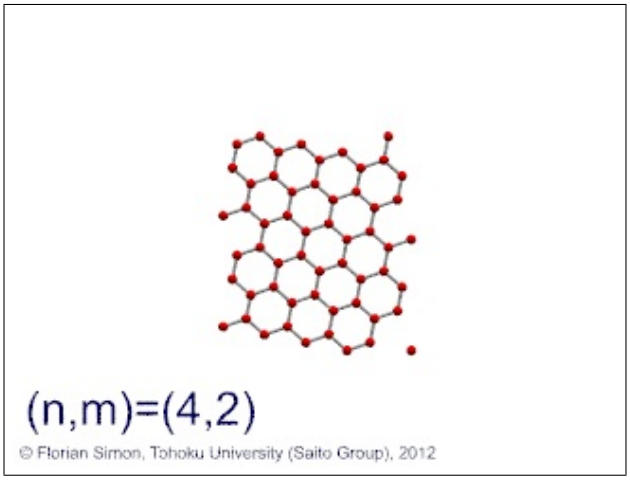

Figure 12: object { ntube(4,2,1,1,1) scale 1.0 rotate <0,0,0> translate <0,0,0> }.

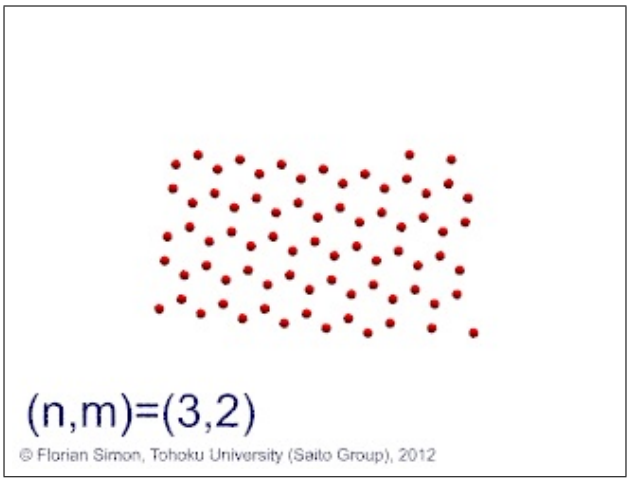

Figure 13: object { ntube(3,2,0,1,1) scale 1.0 rotate <0,0,0> translate <0,0,0> }.

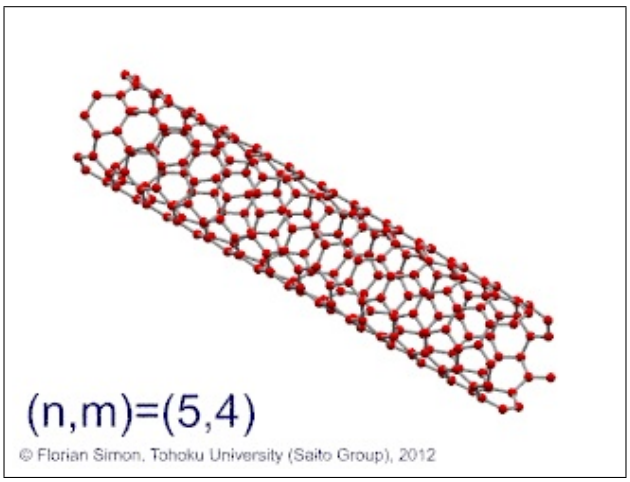

Figure 14: object { ntube(5,4,1,1,2) scale 1.0 rotate <0,30,-30> translate <0,0,0> }.

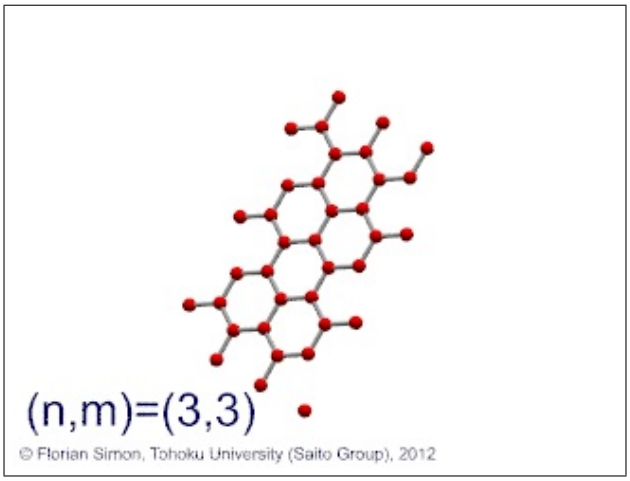

Figure 15: object { ntube(3,3,1,3,1) scale 1.4 rotate <0,30,-30> translate <0,0,0> }.

Also take a look at my web page please. You can find plenty of animations there related to this topic[3].

[3] http://flex.phys.tohoku.ac.jp/~florian.simon/