w90pov manual

Daniel Åberg

(Dated: May 1, 2013)

This document briefly describes how to use the ray-tracing software POV-Ray to render isosurfaces from `xsf` files generated by the program package wannier90. It is assumed that POV-Ray is already installed. Specifically, to be able to render smooth isosurfaces using tri-linear interpolation, the POV-Ray version number needs to be larger than 3.6 (currently it's in v3.7 RC5, binaries and sources are available at `http://www.povray.org/beta`).

A working knowledge of POV-Ray is of course nice, but is no prerequisite. The interested reader is referred to the POV-Ray official site `http://www.povray.org/` and Friedrich Lohmueller's tutorial on anaylical geometry, `http://www.f-lohmueller.de/pov_tut/a_geo/a_geo__e.htm`. Comments and/or bugs can be sent to `aberg2@llnl.gov`.

Compilation

Modify `Makefile` according to your installation and then type '`make`'.

Usage

It is recommended to create a separate directory and copy/move/link the already produced `xsf` files to that directory. Edit the input file w90pov.inp (to be described in the next section) and then run w90pov:

```
$> w90pov w90pov.inp
```

This will produce four text files as well as a number of `df3` files. The latter are binary files containing the Wannier function data in a format suitable for POV-Ray. The text files contain all other information about the atomic structure and isosurfaces.

- `<seedname>.pov`
  This is the main scene file and only tells POV-Ray which other files to include.
- `mydefs.inc`
  Contains information on camera, light, where to aim the camera, cell metric, as well as macros to render atoms, bonds, and isosurfaces.
- `densities.inc`
  Contains definitions of the `df3` files and the corresponding cell metric.
- `unitcell.inc`
  Contains information on atomic positions, bonds, and colors.
- `blobs.inc`
  Contains information which isosurface at what isolevel to render (as well as isosurface color)

Feel free to experiment with the settings in these files, specifically the camera position, zoom, where to aim the camera, and which atoms to include. Examples can be found in the `examples` directory.

To render the scene then enter:

```
$> povray <seedname>.pov +H{height} +W{width} +A0.14
```

Of course replace `<seedname>` with the seedname from the wannier90 calculation, and `{height} and {width}` with the desired size of your picture. The "`+A`" option enables anti-aliasing. In addition, the output format can be controlled using "`+F`"*x*, where *x* can take, for example, the values `C` (compressed Targa-24), `N` (PNG), `P` (PPM), or `T` (uncompressed Targa-24). The output file name can be set using "`+O<outfile>`". Please refer to the POV-Ray documentation for further details.

The `w90pov.inp` file

- `numwan` - integer;
  Number of Wannier functions to render.
- `seedname` - character string;
  The root name of wannier90 files.
- `wanlist` - integer array;
  List of `xsf` files to read, of length `numwan`.
- `isolevel` - real array;
  List of isolevels to render, of length `numwan`. The allowed range `(MAX,MIN)` for a given function is shown during the run.
- `isopm` - real array;
  List of how to render each function, of length `numwan`.
  `-1` = negative values;
  `0` = don't render this function;
  `1` = positive values;
  `2` = positive and negative values.
- `wancol` - real array;
  List of colors in rgb format to use for isosurfaces, of length 3×`numwan`. The values, three for each color, are assumed to range from 0-1.
- `trans` - real array;
  List of degree of transparency for each surface, of length `numwan`.
  `0` = opaque;
  `1` = transparent.
- `camera` - character string;
  Camera position. There are six choices, valid entries are `x`, `y`, `z`, `a1`, `a2`, and `a3`. In each case, the camera is positioned along the corresponding vector. By default, the camera looks at the center of the cell (see `lookpos`).
- `lookat` - real array, optional; default: center of unit cell;
  Position where camera is aimed in Cartesian coordinates; length 3.
- `bondcut` - real number, optional; default: `0.9`;
  Bond cut-off prefactor. Bonds are "drawn" if the distance between two atoms are less then the sum of their covalent radii × `bondcut`. If `bondcut` is negative then a bond is drawn if the distance between two atoms is smaller than the absolute value of this number.
- `bondrad` - real number, optional; default: `0.2`;
  Bond-radius prefactor. The radius of the bond is set to min(radius(atom1),radius(atom2)) × `bondrad`.
- `radialfactor` - real number, optional; default: `0.5`;
  Atomic radius prefactor. Atoms are rendered with the covalent radius × `radialfactor`. The covalent radii are taken from `http://www.ccdc.cam.ac.uk/products/csd/radii/table.php4#group` and can be found in the the subroutine `write_unitcell` of `driver.f90`.
- `interpolation` - integer number, optional; default: `2`
  Determines the degree of interpolation of isosurface. Use `2` for production and, e.g., `1` to test the orientation of the camera.
- `zoom` - real number;
  Zoom factor. Here you have to play around. It's recommended to set `isopm` to an array of zeros or set `interpolation` to zero when finding to the best position and zoom.
- `cellim` - real array, optional; default: `0 1 0 1 0 1`;
  Defines a "box" in which atoms are rendered. The format is
  `<a1_min> <a1_max> <a2_min> <a2_max> <a3_min> <a3_max>`
  and is given in units of the Bravais lattice vectors. Thus, the default values correspond to the atoms inside the unit cell. Note that the code accepts non-integer numbers.

- `cutsphere` - real number, optional; no default;
  Any atom that is located outside a sphere, centered at `lookat`, having radius `cutsphere` will not be rendered.
- `lcage` - logical, optional; default: `true`;
  Render the unit cell.
- `aspectratio` - real number, optional; default: `1.0`;
  Specifies the aspect ratio between image width and height. Be sure to use the same ratio in the actual call to `povray` (e.g. `+H`$\{x\}$ `+W`$\{x\times$`aspectratio`$\}$)

Examples

A number of examples can be found in the `examples` directory. Be sure to first `gunzip` any `*.gz` files.

1. `examples/1.PdN2_1`
   Rendering of four *d*-orbitals centered at Pd atoms in PdN_2, see Fig. 1.
2. `examples/2.PdN2_2`
   Magnification of one of Pd *d*-orbitals, see Fig. 2. Here I've used `lookpos` and `cutsphere`. You will need to copy across `pdn2_00004.xsf` from example 1. Note that `aspectratio` is set to `2.0`, so make sure to specify the correct ratio between the width and height when rendering ("`+H`$\{x\}$ `+W`$\{2x\}$").
3. `examples/3.PdN2_3`
   Same orbital as in the last example (i.e., you will need to copy across `pdn2_00004.xsf` from example 1), but with several transparent isosurfaces (see Fig. 3). This was generated using the example input file, and manually editing the file `blobs.inc`. So, to create multiple isosurfaces, find a line with `elblob`, copy this as many times you wish, and edit the first and last number in each line. These numbers correspond to the isolevel and transparency, respectively. To render this figure, first run `w90pov`, copy `blobs.inc_more` to `blobs.inc`, and run `povray`.
4. `examples/4.LaBr3_1`
   Bromine *p*-orbital in $LaBr_3$, see Fig. 4.
5. `examples/5.LaBr3_2`
   Lanthanum *f*-orbital in $LaBr_3$, see left panel in Fig. 5.
6. `examples/6.LaBr3_3`
   Lanthanum *f*-orbital in $LaBr_3$, see right panel in Fig. 5. This represents a manual "tweak" of example 5. You will need to copy across the two `df3` files generated in example 5. In the file `mydefs.inc` you'll find a new isosurface-macro (`elblob2`) that turns the "blob" into glass. Also, the camera-type has been changed into perspective (instead of orthographic), a plane and sky have been added, and finally the scene is rendered using photons (that is, realistic reflection/refraction).

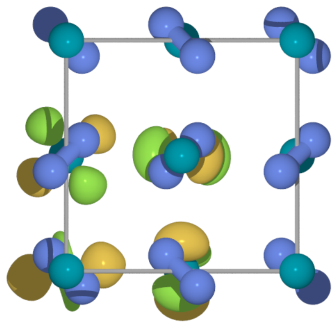

FIG. 1. Four d-orbitals in PdN_2.

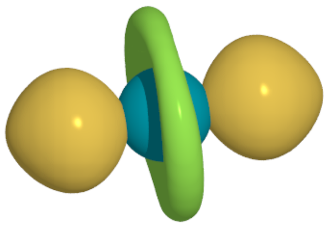

FIG. 2. One of the d-orbitals in PdN_2.

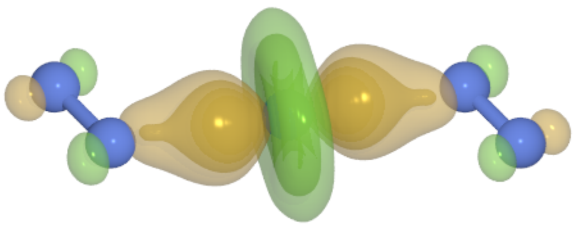

FIG. 3. One of the d-orbitals in PdN_2 with several transparent isosurfaces.

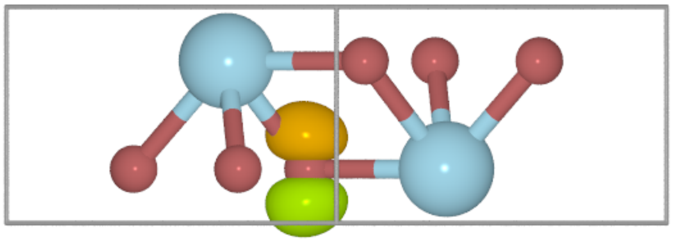

FIG. 4. Bromine p-orbital in $LaBr_3$.

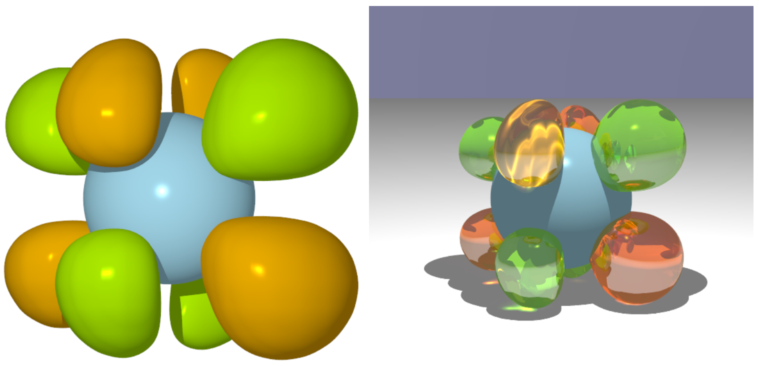

FIG. 5. Lanthanum f-orbital in $LaBr_3$.