spsim User Manual

Version 1.0 (Alpha)

Filipe Maia

September 19, 2006

Contents

1 Introduction **1**

1.1 spsim.conf 1

1.2 Output 3

2 Theory **4**

3 Installation **7**

Chapter 1

Introduction

spsim is a small program which aims to realistically simulate the output of a single particle diffraction experiment. The program reads it's input from a single file named spsim.conf which must reside on the directory where the program is run. It then runs for while and produces several VTK files related to the detector output from the simualted an experiment.

1.1 spsim.conf

spsim.conf is composed of several "key = value lines;". All lines starting with "#" are comments and not interpreted by the program. *All values are in SI units!*
The required keys are:

- number_of_dimensions = <int> - currently only the value of 2 is valid. Number of dimensions of the output.
- input_type = <string> - currently only the value "pdb" is accepted. Defines the type of the input structure on which the experiment is going to be performed.
- pdb_filename = <string> - a string containing the path to the PDB file to be used.
- detector_distance = <float> - the distance to the detector*in meters.*
- detector_width = <float> - the width of the detector*in meters.*
- detector_height = <float> - the height of the detector*in meters.*
- detector_pixel_width = <float> - the width of a single pixel*in meters.*
- detector_pixel_height = <float> - the height of a single pixel*in meters.*
- detector_quantum_efficiency = <float> - the quantum efficiency of the detector. A number from [0.00,1.00] indicating 0 to 100% efficiency

- detector_electron_hole_production_energy = <float> - energy required to produce an electron on the detector*in joules*.
- detector_readout_noise = <float> - RMS of the readout noise, in electrons per pixel.
- detector_dark_current = <float> - dark current of the CCD in electrons/(pixel.second).
- detector_linear_full_well = <float> - maximum capacity of a pixel in electrons.
- detector_binning <int> - number of pixels to bin. Binning is assumed to be the same in all dimensions so a value of 4 means 4x4 binning.
- detector_maximum_value = <float> maximum value that the detector output. For a 16 bit detector this would be 2^{16} or 65535.0.
- experiment_wavelength = <float> the wavelength of the experiment *in meters*.
- experiment_exposure_time = <float> total exposure time in seconds.
- experiment_beam_intensity = <float> total beam intensity integrated over the entire exposure time in photons/m^2.

Due to the way libconfig works all floating point values *must* include a decimal point, otherwise they will be treated as 0. So for example if your detector_readout_noise is 10 you need to write "detector_readout_noise = 10.0;".
Here is an example configuration file:

```
number_of_dimensions = 2;
input_type = "pdb";
pdb_filename = "DNA_triangle.pdb";
# 5cm detector distance
detector_distance = 5.000000e-02;
# 26.8mm x 26mm CCD
detector_width = 2.680000e-02;
detector_height = 2.600000e-02;
#20um square pixels
detector_pixel_width = 2.000000e-05;
detector_pixel_height = 2.000000e-05;
# 15% quantum efficiency
detector_quantum_efficiency = 1.500000e-01;
# 3.6 eV
detector_electron_hole_production_energy = 5.800000e-19;
```

```
detector_readout_noise = 1.000000e+01;
detector_dark_current = 1.000000e-01;
detector_linear_full_well = 2.000000e+05;
# 4x4 binning
detector_binning = 4;
detector_maximum_value = 65535.0;
#13 nm
experiment_wavelength = 1.300000e-08;
# 100fs
experiment_exposure_time = 1.000000e-13;
# 1e15 photons, 100um^2 area
experiment_beam_intensity = 1.0e25;
```

1.2 Output

The output of spsim consists of the following files:

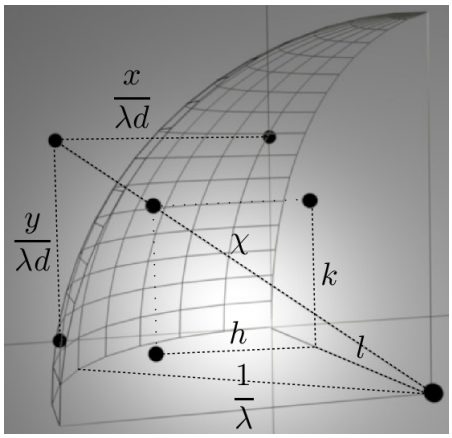
- spsim.confout - contains all information read by the program from spsim.conf
- scattering_factor.vtk - the scattering factors of the input molecule on the detector
- thomson_correction.vtk - the thomson correction factor for each pixel. Simply the differential electron cross-section times the pixel solid-angle
- solid_angle.vtk - the solid angle for each pixel
- photon_count.vtk - the number of photons detected by each pixel. This already takes into account the quantum efficiency and poisson noise.
- electrons_per_pixel.vtk - electrons generated on each pixel due to the incomming photons.
- real_output.vtk - the value that detector output, including all the noise effects.
- noiseless_output.vtk - the value which the detector would output if there were no sources of noise.

An easy way to examine images is to use an excellent visualizer called VisIt produced by the LLNL and freely available for many platforms. Please visit the page http://www.llnl.gov/visit/ for more information, including installation details. From now on i'll assume that you have VisIt installed. But any other VTK visualizer like should work.

Chapter 2

Theory

The way spsim simulates the diffraction patterns is very simple. First it starts by calculating the reciprocal space coordinates of the detector pixels, using the Ewald sphere construct [1]. For each pixel x, y on the detector at a distance d from the source and at a wavelength λ the reciprocal h, k, l coordinates are:



$$\chi = \sqrt{(\frac{x}{\lambda d})^2 + (\frac{y}{\lambda d})^2 + (\frac{1}{\lambda})^2} \tag{2.1}$$

$$h(x,y) = \frac{x}{\lambda^2 * d\chi} \tag{2.2}$$

$$k(x,y) = \frac{y}{\lambda^2 d\chi} \tag{2.3}$$

$$l(x,y) = \frac{1}{\lambda} - \frac{1}{\lambda^2 \chi} \tag{2.4}$$

Then the molecular scattering factors are calculated on those coordinates, using CCP4's atomsf.lib to get the atomic scattering factors. Assuming n atoms at positions (x_j, y_j, z_j) and using a_i,b_i and c from atomsf.lib.

$$f_{0j}(h,k,l) = \sum_{i=1}^{4} a_i exp(-b_i \sin^2 \left[\frac{\sqrt{h^2+k^2+l^2}}{2} \right]) + c \tag{2.5}$$

$$F(h,k,l) = \sum_{j=1}^{n}\{f_{0j}(h,k,l)\cos\{2\pi(hx_j + ky_j + lz_j)\}\} + i\sum_{j=1}^{j=n}\{f_{0j}(h,k,l)\sin\{2\pi(hx_j + ky_j + lz_j)\}\} \tag{2.6}$$

After that the Thomson correction is applied. $K1$ and $K2$ are the relative vertical and horizontal polarized parts of the beam. That an horizontal scaterring plane $K1 = 0$ and $K2 = 1$ and vice versa. For unpolarized light $K1 = K2 = 0.5$. P is the polarization factor. 2θ is the angle between the primary beam and the direction of observation. I_{eTh} is the density of the scattered radiation, I_i is the intensity of the indicent radiation, e is the charge of the electron, m the mass of the electron and c the speed of light. $\frac{e^2}{mc^2}$ is known as the classic electron radius.

$$P(\theta) = K_1 + K_2\cos^2(2\theta) \tag{2.7}$$

$$\frac{I_{eTh}}{I_i} = \frac{e^4}{m^2c^4} \times P(\theta) = Thomson(\theta) \tag{2.8}$$

The Thomson correction is applied to each pixel according to it's solid angle, and then the total number of photons that arrive to each pixel is calculated. Ω is the solid angle and A_{px} is the pixel area projected on a plane passing through the center of the pixel and normal to the direction of observation.

$$\Omega(x,y) = \frac{A_{px}}{x^2 + y^2 + d^2} \tag{2.9}$$

The number of photons that arrive at each pixel, Ph_{inc} is then given by:

$$Ph_{inc}(x,y) = Thomson(\theta) \times \Omega(x,y) \times F(hkl(x,y)) \times I_i \tag{2.10}$$

From those pixel that arrive at each pixel only a certain number is detected. The percentage that on average is detected is known as *quantum efficiency*(τ) and is wavelength dependent. $Poisson(x)$ returns a poisson distributed number with mean x. The number of photons detected on each pixel(Ph_{det}) is then:

$$Ph_{det}(x,y) = Poisson(Ph_{inc}(x,y) \times \tau) \tag{2.11}$$

Using μ to represent the energy to produce one electron hole, the number of electrons generated on the pixel by the detected photons(el_{gen} is then simply:

$$el_{gen}(x,y) = \frac{Ph_{det}(x,y) \times \frac{hc}{\lambda}}{\mu} \tag{2.12}$$

Representing the maximum value output by the detector by V_{max}, the binned pixels by px_{bin}, the pixel linear full well by W_{max} exposure time by t dark current by ι and readout noise by N, the final output is then calculated by:

$$Output(x,y) = \sum_{x_{bin},y_{bin}}^{px_{bin}} \{(el_{gen}(x_{bin},y_{bin}) + \iota t)\} + Gaussian(N) \times \frac{V_{max}}{W_{max}} \tag{2.13}$$

Where Gaussian(x) returns a normally distributed random number with mean 0 and standard deviation x.

The noiseless output is then calculated by:

$$Noiseless(x,y) = \sum_{x_{bin},y_{bin}}^{px_{bin}} \frac{Ph_{inc}(x_{bin},y_{bin}) \times \tau \times \frac{hc}{\lambda}}{\mu} \times \frac{V_{max}}{W_{max}} \tag{2.14}$$

Chapter 3

Installation

spsim should compile in any linux distribution and most kinds of Unix. It might also compile in windows under cygwin although that has not been tested.

To install first simply extract the source code, go to the libconfig-0.9 directory inside of the source code directory, run configure followed by make, move to the previous directory and run make. For example:

```
$ tar -zxvf spsim-1.0.tar.gz
$ cd spsim-1.0
$ cd libconfig-0.9
$ ./configure
$ make
$ cd ..
$ make
```

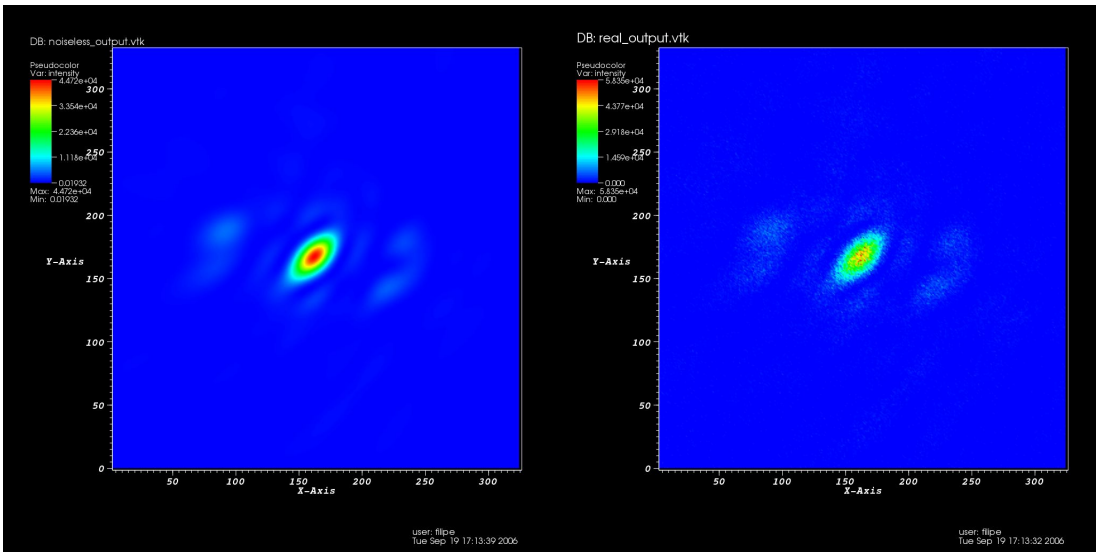
You now should have a binary called "spsim" in your "spsim-1.0" directory, ready to run.

If you wish to turn off MPI support you need to edit the Makefile on the "spsim-1.0" directory and change the MPI = ON line to MPI = OFF.

You can now run the examples by going to the examples directory and running spsim:

```
$ cd examples
$ ../spsim
```

This should produce several files including "real_output.vtk" and "noiseless_output.vtk" that you can now take a look at. They should look like this:



Congratulations you have successfully installed spsim!

Bibliography

[1] *Fundamentals of crystallography. IUCr Texts on Crystallography.* Oxford University Press, 1992.