spsim-gpu New features

Version 1.0

Sergey Bobkov

November 4, 2015

Contents

1 Additions to spsim.conf **1**

1.1 GPU computing 1

1.2 Experiment parameters 1

Chapter 1

Additions to spsim.conf

In order to provide better possibilities for simulation of diffraction images, spsim obtained a list of new features. These features can be activated by changing parameters in spsim.conf

1.1 GPU computing

To activate gpu-powered calculation one should use following option:

- compute_using_gpu = <boolean> - true to compute on GPU, false to compute on CPU;

1.2 Experiment parameters

Also there is a list of other parameters:

- number_of_experiments = <int> - number of diffraction images that will be generated during simulation. File names has pattern "04d_filetype.vtk", i.e. "0001_photon_count.vtk", "0012_real_output.vtk" and so on. All images will be simulated for the same *.pdb file.
- random_orientation = <boolean> - random orientation of a particle for every image.
- uniform_rotation = <boolean> - Particle will be rotated uniformly with regular angles along 2 axes (perpendicular to detector is excluded) during the series. Cannot be true if random orientation is activated. If both random_orientation and uniform_rotation is false, particle will have the same orientation for all images in a series.
- beamstop_radius = <int> - radius (in pixels) of a circle in center in which there is no signal.
- intermediate_files = <boolean> - if set to false, spsim wont create intermediate files, only "real_output.vtk" will be created.
- random_position = <boolean> - random position of particle near center of the detector. Position shift length has standard distribution.

- position_std = <float> - if random position is activated, this parameter controls standard deviation (in pixels) of position shift length.
- binary_output = <boolean> - if activated, all output will have different format. Files will have *.bin extension instead of *.vtk. Inside each file, there are serial values of intensity in float format. Binary format saves up to 60% of storage space. File can be read directly without vtk framework. For example, in python one can read it with Numpy:

  ```
  import numpy as np
  image_data = np.fromfile( path, dtype=np.float32).reshape((sizeX, sizeY)).
  ```

  Where $sizeX$ and $sizeY$ are image dimensions in pixels.

Here is an example configuration file:

```
number_of_dimensions = 2;
input_type = "pdb";
pdb_filename = "examples/DNA.pdb";
detector_distance = 1.000000e-01;
detector_width = 2.0000e-01;
detector_height = 2.0000e-01;
detector_pixel_width = 2.000000e-04;
detector_pixel_height = 2.000000e-04;
detector_quantum_efficiency = 1.500000e-01;
detector_electron_hole_production_energy = 5.800000e-19;
detector_readout_noise = 1.000000e+04;
detector_dark_current = 5.000000e-01;
detector_linear_full_well = 2.000000e+05;
detector_binning = 1;
detector_maximum_value = 65535.0;
experiment_wavelength = 3.00000e-10;
experiment_exposure_time = 1.000000e-13;
#1e20 photons, 100um^2
experiment_beam_intensity = 2.102500000e+30;
compute_using_gpu = false;
number_of_experiments = 1;
random_orientation = false;
uniform_rotation = false;
beamstop_radius = 0;
```

```
intermediate_files = true;
random_position = false;
position_std = 1.0;
binary_output = false;
```