Diffraction Image: a new CCP4 library

Francois Remacle, Graeme Winter, CCLRC Daresbury Laboratory Warrington WA4 4AD, United Kingdom

What is this library?

The Diffraction Image library started as part of XIA automation project. Since it appeared that it could be useful to have it as a separate entity, it was decided to include this library with the other CCP4 core libraries and therefore make it available publicly. XIA2 now relies a couple of simple programs using this library.

The purpose of the library is to have a single way of handling diffraction image that can be originally of various different formats. Therefore, even if the library contains several different files implementing similar methods for different format of images it has been designed so that you only need to use a single object. All the work of idenitifying what type of image is done internally.

Currently, the following format are supported by this library:

- ADSC
- MAR: MAR345, MAR300, MARCCD
- RIGAKU: R-AXIS IV, SATURN
- CBF

The library also contains a PeakList object that can be populated with the peaks (spots) found on particular images, there are then some operations that can be performed on the peaks.

The Diffraction Image Library comes with Tcl and Python interfaces so that the library can be used with both of these scripting languages as well.

How to use it and what can you do with it?

Firstly, below is the description of the classes from the Diffraction Image library, there are only two classes / objects availables which make things a lot simple to use.

The are different ways for initialising a diffraction Image Object. Which one is more appropriate mainly depends on what you want to do.

- You can create an empty object and load an image in the object via the

  `load`

  method.

- You can create an object giving the filename as a parameter to have everything already loaded (i.e. header and image).

- You can create an object and only load the header via the

  `loadHeader`

  method. This is useful if you just need to access information from the header.

To illustrate this let us use some source code

```
void printHeader(char* filename)
{
DiffractionImage* diff=new DiffractionImage();
diff->loadHeader(filename);
printf("Image type : %s\n",diff->getFormat());
printf("Collection date : %s\n",diff->getDate());
printf("Exposure time : %f s\n",diff->getExposureTime());
printf("Detector S/N : %s\n",diff->getSerialNo());
printf("Wavelength : %f Ang\n",diff->getWavelength());
printf("Beam center : (%f mm,%f mm)\n",diff->getBeamX(),diff->getBeamY());
printf("Distance to detector : %f mm\n",diff->getDistance());
printf("Image Size : (%d px, %d px)\n",diff->getWidth(),diff->getHeight());
printf("Pixel Size : (%f mm, %f mm)\n",diff->getPixelX(),diff->getPixelY());
printf("Oscillation range : %f -> %f deg\n",diff->getPhiStart(),diff->getPhiEnd());
printf("Two Theta value: %f deg\n",diff->getTwoTheta());
}
```

We used the third option because in this method we just want to output the header information. But the lines

```
DiffractionImage* diff=new DiffractionImage();
diff->loadHeader(filename);
```

Could have been replaced by

```
DiffractionImage* diff=new DiffractionImage(filename);
```

or

```
DiffractionImage* diff=new DiffractionImage();
diff->load(filename);
```

There is not one more suitable than another in general, it mainly depends on what you intend to do, especially whether or not you need to load the image in memory which can slow down a treatment of large number of images.

Examples of programs

Currently the library provides also two simple programs called "diffdump" and "printpeaks" that are respectively used to display all the "standard" information of a Diffraction Image and printing the list of peaks found on the image. These are also simple examples of how to use the object from the library.

diffdump usage:

```
"diffdump <filename>"
```

Example of running diffdump:

```
>diffdump adsc_example_001.img
Image type : adsc
Collection date : Sun Sep 26 14:01:35 2004
Exposure time : 5.000000 s
Detector S/N : 445
Wavelength : 0.979660 Ang
Beam center : (105.099998 mm,101.050003 mm)
Distance to detector : 170.000000 mm
Image Size : (2048 px, 2048 px)
Pixel Size : (0.102400 mm, 0.102400 mm)
Oscillation range : 290.000000 -> 291.000000 deg
Two Theta value: 0.000000 deg
```

printpeaks usage:

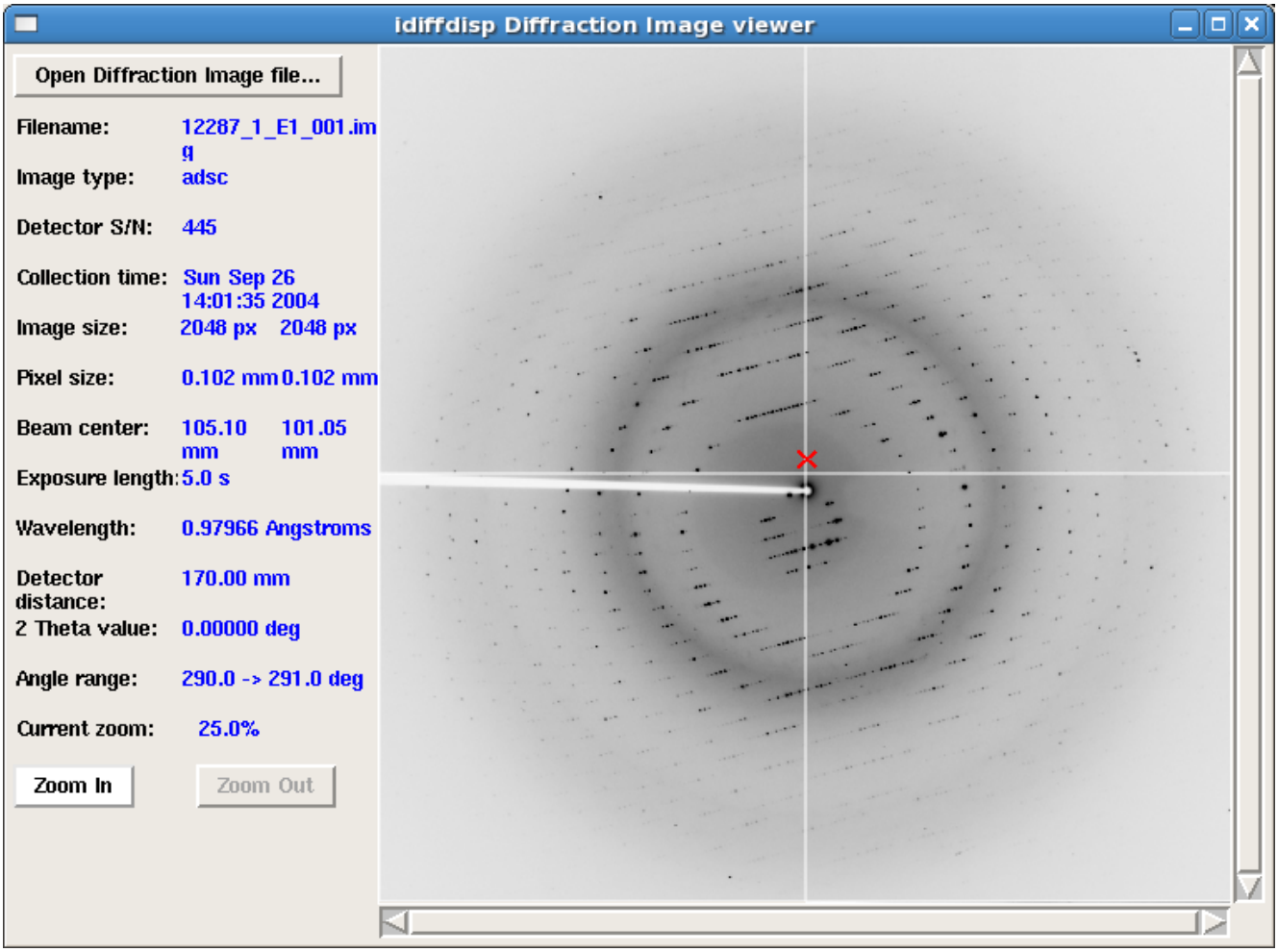
```
"printpeaks [-th <intensity_threshold>] <filename>"
```

Example of running printpeaks

```
>printpeaks -th 5 12287_1_E1_001.img
Searching for peaks...
Done!
 1212 peaks found

Peak 1 = x:17.642262  y:5.422019  phi:290.500000  intensity:411.251343
Peak 2 = x:14.642123  y:-38.075809  phi:290.500000  intensity:327.479645
Peak 3 = x:11.323599  y:11.402020  phi:290.500000  intensity:326.635406
Peak 4 = x:16.016949  y:11.563182  phi:290.500000  intensity:312.215332
Peak 5 = x:-14.611582  y:45.403568  phi:290.500000  intensity:297.010834
Peak 6 = x:4.550874  y:-46.884823  phi:290.500000  intensity:285.352112
Peak 7 = x:4.951574  y:-0.162381  phi:290.500000  intensity:284.619934
Peak 8 = x:14.114961  y:-26.515749  phi:290.500000  intensity:265.874084
Peak 9 = x:31.641386  y:-19.859549  phi:290.500000  intensity:258.448975
Peak 10 = x:-17.301394  y:-37.574718  phi:290.500000  intensity:235.708313
... ... ...
Peak 1203 = x:6.917424  y:89.008492  phi:290.500000  intensity:5.065027
Peak 1204 = x:-57.386951  y:78.550629  phi:290.500000  intensity:5.062943
Peak 1205 = x:92.328209  y:19.058695  phi:290.500000  intensity:5.055988
Peak 1206 = x:-80.527000  y:-22.509737  phi:290.500000  intensity:5.051531
Peak 1207 = x:-69.974136  y:51.322342  phi:290.500000  intensity:5.051506
Peak 1208 = x:-67.926010  y:-52.717159  phi:290.500000  intensity:5.049273
Peak 1209 = x:-42.943699  y:-5.723580  phi:290.500000  intensity:5.041045
Peak 1210 = x:-5.873695  y:57.763443  phi:290.500000  intensity:5.033473
Peak 1211 = x:41.841999  y:-63.973579  phi:290.500000  intensity:5.027031
Peak 1212 = x:-43.768307  y:96.585030  phi:290.500000  intensity:5.017936
```

In addition to these a tcl image viewer called idiffdisp is being developed and will be later accessible through ccp4i and be used by XIA2 it uses the tcl interface of the library to access all methods from the library. It currently displays the header information on the left part of the window and the image on the right part, it also enables different zoom level (25%, 50%, 100%, 200%). Below you can see a screenshot of the current development state of idiffdisp.



Acknoledgments

- Rigaku for supplying demonstration code for Raxis IV images.
- Many different suppliers of example images of various detectors.
- Paul J. Ellis and Herbert J. Bernstein authors of the CBF library on which this library relies to support CBF format.

Appendix: Library API

- DiffractionImage

Method	Description
DiffractionImage()	Default constructor of a Diffraction Image.
DiffractionImage(**string** filename)	Create and load everything available from the image file called "filename".
int getWidth()	Return the width of image in pixels.
int getHeight()	Return the height of image in pixels.
float getBeamX()	Return the x position of the beam center in millimeters.
float getBeamY()	Return the y position of the beam center in millimeters.

float getWavelength()	Return the wavelength of X-Ray source in Angstroms.
float getDistance()	Return the crystal to detector distance in millimeters.
float getTwoTheta()	Return the two-theta angle used in data collection
float getPhiStart()	Return the phi-angle at the beginning of the oscillation
float getPhiEnd()	Return the phi-angle at the end of the oscillation
float getPixelX()	Return the width of a pixel in millimeters.
float getPixelY()	Return the height of a pixel in millimeters.
float getExposureTime()	Return the exposure time in seconds.
char* getFormat()	Return "mar", "adsc", "raxis", etc... according to the image format.
char* getDate()	Return the date of the image collection using a formatted string (dd-mm-YYYY).
char* getSerialNo()	Return the detector serial number if available.
unsigned short * getImage()	Return a pointer to the image.
void setBeamX(**float** x)	Sets the x-position of the beam center in millimeters.
void setBeamY(**float** y)	Sets the y-position of the beam center in millimeters.
void setWavelength(**float** wave)	Sets the wavelength in Angstroms.
void setDistance(**float** dist)	Sets the crystal to detector distance in millimeters.
void setTwoTheta(**float** twotheta)	Sets the two-theta angle used in data collection.
void setPhiStart(**float** phistart)	Sets the phi-angle at the beginning of the oscillation
void setPhiEnd(**float** phiend)	Sets the phi-angle at the end of the oscillation
void setPixelX(**float** pixelwidth)	Sets the pixel size in millimeters
void setPixelY(**float** pixelheight)	Sets the pixel size in millimeters
void setExposureTime(**float** time)	Sets the exposure time in milliseconds
void load(**string** fileName)	Load in an image from the file "filename".
void loadHeader(**string** fileName)	Load header information only from the file "filename".
float* assymetry(**void**)	Give an estimate of how much symmetry there is in the image background.
float* radial(**int** bins)	Generate a radial profile of the image
int corner(**void**)	Determine whether the image goes to the corner or whether it is an inscribed circle.
int* histogram(**void**)	Calculate a contrast histogram.
***float** gain(**void**)*	*Calculate an estimation of the gain (yet to be implemented).*

- PeakList

Method	Description
PeakList()	Default constructor of a empty PeakList.
PeakList(**DiffractionImage*** diffractionimage)	Construct a PeakList from the given DiffractionImage object.
int length()	Gives the length of the list.
void clear()	Clears the list.
void add(**float** x, **float** y, **float** intensity)	Add a peak the image coordinates (x,y) and with the specified intensity.
void remove(**int** offset)	Remove the Peak at the specified position in the list.
void add(**Peak** peak)	Add the specified peak to the list.
void find(**DiffractionImage*** diffractionimage)	Search for the peaks in the specified DiffractionImage object.
void find(**DiffractionImage*** diffractionimage, **int** maxPeaks)	Search for the peaks in the specified DiffractionImage object with a maximum of maxPeaks results.
void find(**DiffractionImage*** diffractionimage, **int** maxPeaks, **float** intensityThreshold)	Search for the peaks in the specified DiffractionImage object with a maximum of maxPeaks results having an intensity of at least intensityTreshold.
void reciprocal(**float** dist, **float** wave)	Convert the peak coordinates from dectetor coordinates to reciprocal space coordinates.
void detector(**float** dist, **float** wave)	Convert the peak coordinates from reciprocal space coordinates to detector coordinates.
Peak pop(**int** offset)	Gets the peak from the specified position and removes it from the list.
Peak * getPeaks()	Return the content of the PeakList as a vector of Peak.
int circle(**int** iter, **float** width, **float**& x, **float**& y, **float**& r)	Find a circle in the spot using a stochastic process and return the most promising candidate. Width determines how close to the arc a spot must be to be accepted. Iter is the number of attemps in fitting a circle.
int remove(**float** width, **float** x, **float** y, **float** r)	Removes the spots that were found to be on the circle fitted with the circle() method described above.

- Peak is a structure having the following 7 fields,
  - **float** x : x coordinate of the peak on the image
  - **float** y : y coordinate of the peak on the image
  - **float theta :** Theta value for the image where the spot is found
  - **float phi :** phi value for the image where the spot is found
  - **float kappa :** kappa value for the image where the spot is found
  - **float**[3] p : Array containing the reciprocal spaces coordinates of the Peak
  - **float** intensity : Intensity of the Peak