

2dx_merge: Data management and merging for 2D crystal images

Bryant Gipson, Xiangyan Zeng, Henning Stahlberg *

Molecular and Cellular Biology, Briggs Hall, College of Biological Sciences, University of California at Davis, 1 Shields Avenue, Davis, CA 95616, USA

Received 22 August 2007; received in revised form 29 August 2007; accepted 6 September 2007

Available online 25 September 2007

Abstract

Electron crystallography of membrane proteins determines the structure of membrane-reconstituted and two-dimensionally (2D) crystallized membrane proteins by low-dose imaging with the transmission electron microscope, and computer image processing. We have previously presented the software system *2dx*, for user-friendly image processing of 2D crystal images. Its central component *2dx_image* is based on the MRC program suite, and allows the optionally fully automatic processing of one 2D crystal image. We present here the program *2dx_merge*, which assists the user in the management of a 2D crystal image processing project, and facilitates the merging of the data from multiple images. The merged dataset can be used as a reference to re-process all images, which usually improves the resolution of the final reconstruction. Image processing and merging can be applied iteratively, until convergence is reached. *2dx* is available under the GNU General Public License at <http://2dx.org>.

© 2007 Elsevier Inc. All rights reserved.

Keywords: *2dx* software; Electron crystallography; 2D crystals; Membrane protein; Structure determination; Computer image processing; MRC software; Merging

1. Introduction

Electron crystallography can determine the structure of membrane proteins that are reconstituted into a lipid membrane and two-dimensionally (2D)¹ crystallized in the membrane. These crystals are imaged by transmission electron microscopy, and the 3D structure of the membrane proteins can be reconstructed by computer image processing (Renault et al., 2006). This method has led to the determination of several atomic resolution structures of membrane proteins, ranging from the work of Henderson and Unwin (1975), to the 1.9 Å 3D structure of Aqp0 by Gonen et al. (2005).

Detergent-solubilized and purified membrane proteins can be crystallized as 2D sheets by dialysis or other methods (Jap et al., 1992; Kühlbrandt, 1992; Levy et al., 2001;

Remigy et al., 2003; Signorell et al., 2007), and high-resolution cryo-electron microscopy (cryo-EM) images are recorded from non-tilted and tilted preparations of the planar membrane protein crystals (Fujiyoshi et al., 1991; Gyobu et al., 2004; Schmidt-Krey, 2007). Images are recorded on CCD cameras, or on photographic film with subsequent digitization. The resulting 2D image datasets are usually of large pixel dimensions and may contain high-resolution information covered by very high levels of noise.

Computer image processing can be applied to extract the structural information from the recorded images. Henderson at the Medical Research Council in UK, and coworkers have developed a powerful suite of programs for this purpose, which is available as the so-called MRC software (Crowther et al., 1996). The MRC programs are a collection of programs, mostly written in Fortran-77, which perform various image processing and data merging operations. The basic algorithm for the processing of one image consists of determining the 2D crystal distortions that might be present in one image, and correcting these by computer image “crystal unbending”. The distortion-

* Corresponding author. Fax: +1 530 752 3085.

E-mail address: HStahlberg@ucdavis.edu (H. Stahlberg).

¹ Abbreviations used: 2D, two-dimensional; 3D, three-dimensional; XRD, X-ray diffraction; cryo-EM, Cryo-electron microscopy; MRC, Medical Research Council.

corrected image is then Fourier transformed, and for every crystal lattice diffraction spot, values for amplitudes and phases are extracted, together with data that allow one to judge the reliability of the measured values. These data are then corrected for the effect of the electron microscope's transfer function and are used for the synthesis of a final 2D reconstruction map, which shows the projection map of one or more unit cells of the 2D crystal structure.

If data from several different images are available, these can also be merged into one common dataset, which in the case of several non-tilted images will then give a more reliable, or better resolved, projection map of the crystal structure. If data from differently tilted 2D crystal samples are available, these can be merged into a 3D dataset, which eventually can lead to a full 3D reconstruction of the membrane protein structure.

Structure determination of membrane proteins by electron crystallography usually employs large amounts of data. Depending on the crystal quality and the performance of the sample preparation and data collection, several thousands of images usually need to be recorded, from which a sub-set of micrographs are selected by optical laser diffraction (Aebi et al., 1973). Photographic films that show optical diffraction are then digitized. If a typical micrograph (e.g. Kodak SO-163) is digitized at 7 μm pixel size as 16 bit gray-value pixels, then an image file of 13,500 \times 10,700 pixels or 289 Mbyte in size can be produced. If image processing of several hundreds or thousands of such images is to be performed, a computer system capable of archiving, processing and managing Terabytes of image data is required. Computer image processing of 2D crystal images has mostly been done with the MRC programs. An interesting new development for 2D crystal image processing is the IPLT software package (Philippsen et al. (2003)).

We have previously published our software system *2dx*, which assists the user in the processing of 2D crystal images (Gipson et al., 2007). The central component of this software system is *2dx_image*, which allows the fully automated processing of one 2D crystal image. *2dx_image* is based on the MRC software, and partners the slightly adapted Fortran programs of the MRC package with a graphical user interface (GUI) that guides the user through the different processing steps, assists in the choice of processing parameters, and provides graphical and commented feed-back on the processing results. In addition to the MRC programs, *2dx_image* contains several functions and independent programs to assist the user, streamline the processing, and allow automation—as in the case of the automatic lattice determination for 2D crystal images (Zeng et al., in this issue-b). *2dx_image* also contains a module for Maximum Likelihood processing of data from one 2D crystal image, and for a cross-correlation based single particle processing of the 2D crystal unit cell images (Zeng et al., in this issue-a).

We present here the program *2dx_merge*, which assists the user in the management of a 2D crystal image process-

ing project, and additionally performs the merging of data from multiple images. *2dx_merge* is part of the *2dx* software, and interacts with the other components of the package. *2dx* is available at <http://2dx.org> and runs natively on Mac OSX and various Linux versions.

2. Software design

2dx_merge was designed following the same philosophy as used for *2dx_image*. Both allow the user to edit scripts, change and save parameters, and view images and log file output at different verbosity levels. *2dx_merge* manages the processing of a series of images, each individually processed with *2dx_image*, and can merge the results. *2dx_merge* combines in one program the “merger” and the “manager” functions that were announced in Gipson et al. (2007).

2.1. File structure conventions

Upon launch, *2dx_merge* requests the user to navigate to a project root directory, which, for example, could have the location “~/MYPROT/”. If not yet existing, *2dx_merge* will create a sub-directory “merge” in the project root directory, where it installs a local configuration data file called “*2dx_merge.cfg*”. *2dx_merge* will then open a graphical user interface (GUI) window that shows several panels to display data, or to select and execute specific functions or routines (Fig. 1). 2D crystal images should each reside within one directory; these directories should be placed within tilt-angle group directories, which should themselves reside within the “~/MYPROT/” top-level project directory. If, for example, the user has a set of non-tilted images and sets of images that were recorded at 30° and at 45° nominal sample tilt-angles (rounded to nearest 5°), then three sub-directories should be created called “~/MYPROT/MYPR-00”, “~/MYPROT/MYPR-30”, and “~/MYPROT/MYPR-45”. These should contain the image directories, which each should contain one 2D crystal image, as also described in Gipson et al. (2007). The purpose of the tilt-angle grouped directory structure is to facilitate the definition of tilt-angle specific default settings, as each tilt-angle directory has its own copy of the default parameter configuration file *2dx_master.cfg*. A user could for example have as default settings for the 30° tilted images a resolution of 5 Å, and as default setting for the 45° tilted images a resolution of 7 Å. This directory structure is automatically generated by *2dx_merge* during import and directory selection. Nevertheless, *2dx_merge* will also accept a different directory organization, as long as each image directory contains a copy of the *2dx_image.cfg* file, and each image-group directory contains a copy of the *2dx_master.cfg* file.

2.2. Adding images to an image processing project

2dx_merge assists in adding newly recorded images into the list of available images for processing. This is done via a

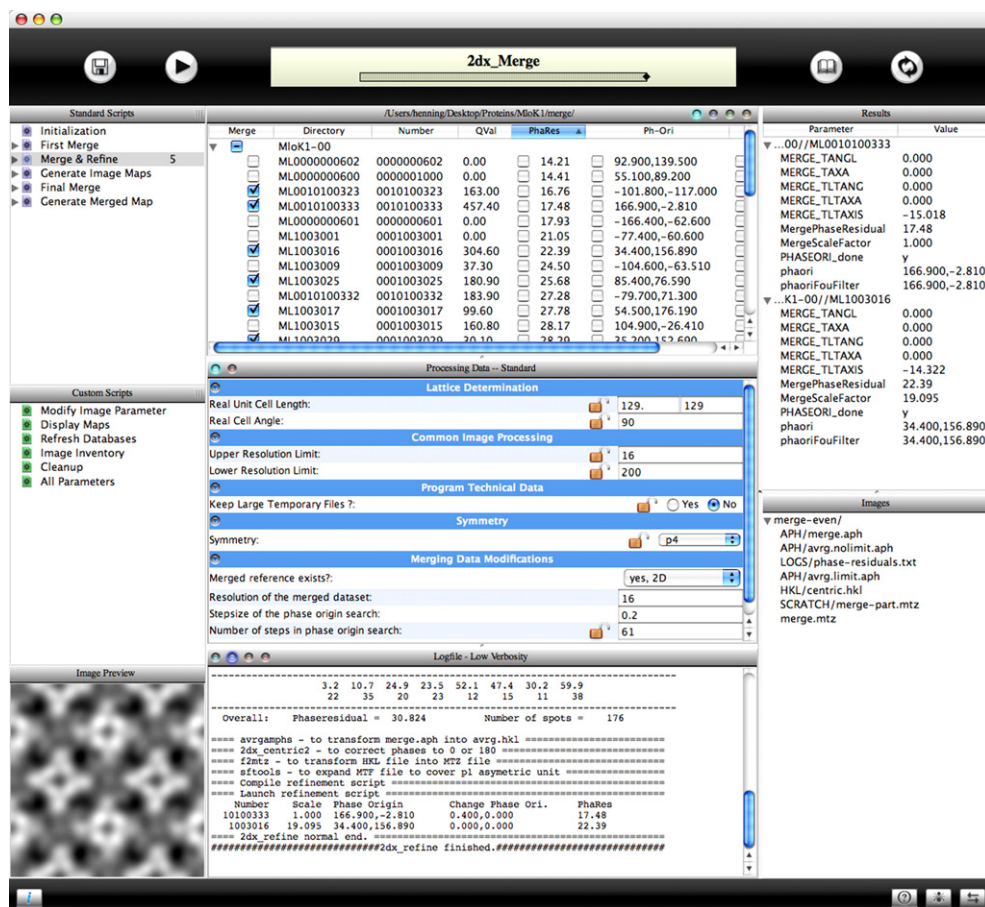


Fig. 1. The graphical user interface of the *2dx_merge* program.

pull-down menu option “Import Images ...”, which prompts the user to locate either an individual image or a directory containing several image files. Image files can be in MRC or TIFF file format, and their name should follow the following convention: the first four characters of the name identify the protein, the next two digits describe the tilt-angle category, the next six digits describe the micrograph number, and the last two digits are reserved for potential copies of an image, in case this image should be processed several times. This might, for example be useful if one image contains two overlaying crystal lattices. A 30°-tilt image of micrograph 123456 could then for example be called “MYPR3012345600.tif”. *2dx_merge* will use the information contained in the image file name to recognize the tilt-angle and micrograph number, create the corresponding image sub-directory, move the image file into that directory, and prepare it for image processing by running the initialization script of *2dx_image* on it.

2.3. Graphical user interface

The central GUI of *2dx_merge* (Fig. 1) is structured similarly to the GUI of *2dx_image*: The main part of the window is split into three columns, each of which is split into several vertically arranged sub-panels. In the default setting, the top section of the central column displays the

list of all image-containing sub-directories that belong to the current project, with each image directory displayed as a row in the central table, grouped by tilt-range category. The data for this table are drawn from the “*2dx_image.cfg*” data files that reside in each image sub-directory. This central table displays for each image several selectable values, like QVal, phase residual, resolution, etc., the specific selection and arrangement of which can be modified by the user and saved as default setting for future usage. For instance, if “Quality Value” is chosen as a sorting data column via single click of the header of the column, the images are organized according to their QVal values. The QVal number for an image is a one-dimensional value that quantifies the quality of the processed image. The QVal for an image is calculated by an empirical formula in *2dx_image*, as described in Gipson et al. (2007). It can be used to recognize the “good” images for merging. It would then be a simple matter to select the first N images above a certain QVal to form the basis of a merging step. Selection sets of images may also be saved for quick re-application of previous merge results, or to run parallel reconstructions.

Below the image directory pane is the “Processing Data” pane, which displays the processing parameters that are relevant to the current script. The pane below this displays the “Logfile” of the currently selected script, once

processing has been performed and a log file has been produced.

The left column in the GUI displays, in the top pane, the “Standard Scripts” that the user can select by mouse click, and execute by pressing the “play” button in the top bar of the GUI. This pane shows the scripts that are to be executed sequentially during the normal merging procedure. The pane below the “Standard Scripts” shows a set of “Custom Scripts”, which the user might utilize for specific tasks. The last pane in the left column displays a view of the currently selected image’s header information, or an image preview if an image is selected (see below).

If an executing script produces results in form of defined variables or identified image files, these will be displayed in the right column of the GUI: The top pane on the right side of the GUI displays variable parameters, grouped by their relevant merging or image directories. If for example the phase origin and the beam tilt are to be refined for several images, then the “Results” pane will display these newly refined values grouped under their respective parent directory sub-sections. Finally, the lower right pane in the GUI displays the list of image files or other files that a script might have produced and reported for display.

The top of the GUI features a header bar, which contains the progress bar, indicating the name of the currently running script, and the progress of the execution of this script. On the left side of the progress bar are buttons to “Save” the current set of parameters, and the “play” button to run the currently selected script. On the right side of the progress bar are buttons to activate the “Manual” function of each script, or to “Reload” the image value table in the top central pane.

The footer bar of the GUI has on the left edge a button that toggles the “Image Preview” pane between the file header display mode, and the image preview mode. On the right edge is a button allowing the user to toggle between normal execution mode and “dry run” mode, in which a script can be executed, but the results are not written back into the local or image sub-directory datafiles. A bug report button, immediately left of the “Dry Run” button, allows users to submit bugs/issues/feature requests relating to 2dx. It points to the bug report tool at: http://2dx.org/documentation/bug_report. Immediately to the left of the Bug Report button is a “Help Button” which points the user to general documentation of 2dx at: <http://2dx.org/documentation/2dx-software>.

3. Communication between GUI and scripts

The program architecture of *2dx_image* and *2dx_merge* are nearly identical. As with *2dx_image*, scripts are implemented as cshell-style templates. Upon clicking the “run” button in the GUI, the empty variable declarations in the top part of the script templates are completed with data from the database, and an executable cshell script is created and placed in the “proc” sub-directory in the local image directory. The GUI then executes that script as an indepen-

dent sub-process, and interprets the output of the running script. For example, the *2dx_merge* script “*2dx_refine.script*” produces a logfile “LOGS/*2dx_refine.log*” and a results file “LOGS/*2dx_refine.results*”. The content of the logfile is displayed by the GUI of *2dx_merge* in the lower central “Logfile” pane, and the content of the results file is used to populate the “Results” and “Images” panes in the GUI.

The interpretation of the results files is different for *2dx_image* and *2dx_merge*: *2dx_image* allows single parameter values to be updated to a local parameter set. In *2dx_merge*, a script can update the parameters of the merging directory, but also of any image directory. A directive “set {variable} = {value}” will update the merging parameters, while encapsulating a set of such directives between the flags <IMAGEDIR= ‘{directory name}’ > and </IMAGEDIR> will set variables in the relevant image sub-directory to the appropriate value.

Scripts can indicate produced image files with an entry in the results file as “# Image:”. The names of these files are then displayed in the “Images” pane, again grouped by their relevant sub-directories. Files in this pane can be opened by double-clicking on the file name.

Scripts can now assign “Nicknames” to files as well, which can be any plain text description of that file (Fig. 2). For example, an entry in the results file as:

```
#IMAGE: ‘PROT0012345600.ref.fft.mrc’ <Fourier Transform of Reference>
```

would prompt the GUIs of *2dx_image* and/or *2dx_merge* to list this file as “Fourier Transform of Reference” in the results pane, and showing as *mouse-over* help text its real file name “PROT0012345600.fft.mrc”.

Files can also be categorized as normal or important files:

```
#IMAGE-IMPORTANT: ‘PROT0012345600.fft.mrc’ <Fourier Transform>
```

would prompt the GUI to display this file highlighted in red. The “Images” file list can be filtered such that only these “important” images are shown in the results view.

Similarly to the parameters, all image/results files appear in the results output relative to the directory they are contained in.

While processing errors in an individual image are often easily repaired, unwanted changes to a collection of images (e.g. the coarse refinement of phase origins for a large number of images) can be labor-intensive to recover from. To aid in efficient user exploration of parameter space or new processing configurations, we have added a “Dry Run” option allowing the results of processing to be displayed with none of the results committed to any directories/parameters—the goal of such steps being to promote user experimentation in an environment where it is safe to do so.

3.1. Translator infrastructure for viewing results

We have added a “translator” architecture, to facilitate the viewing of results files. By default, files with extensions

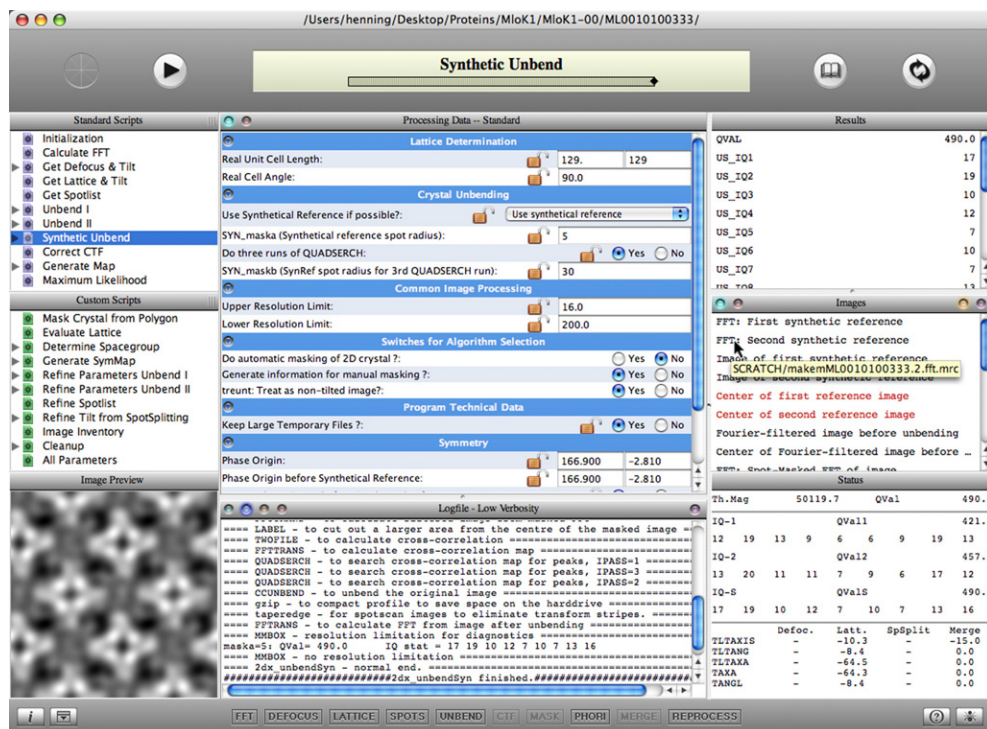


Fig. 2. The graphical user interface of the *2dx_image* program. A “Synthetic Unbend” script is now included, which uses the merged dataset to generate synthetic references. The resulting IQ statistics from the “Synthetic Unbend” run are listed as third entry in the “Status” pane bottom right, labeled IQ-S. Images are displayed as nicknames, with the real file names as “mouse-over” text.

.aph, .hkl, .mrc, .mtz, .ps, and .txt, are viewable in platform-specific standard viewers (navigator, text editing software, etc.). Additionally, users may define custom translators in the plugins/translators directory. These csh scripts take the form of “{ext}.tr” where {ext} defines the file extension the translator should be applied to.

In most respects these translators are identical to the “script” files that are executed as processing commands. The main difference is that they inherit a list of configured editors as defined by config/2dx.cfg as \${program_name}_2dx_app. For instance, the translator file for MTZ files (a binary CCP4 format for reflection data) is called mtz.tr. We provide it as a cshell script that uses the CCP4 program “mtzdump” to transform the binary MTZ file into a temporary text file “outfile”, which is then displayed with the standard platform (and implementation) dependent script editor, by calling “\${scriptEditor}_2dx_app \${outfile}” as the last command in mtz.tr.

The user can add a new translator by creating a new script for the extension one wishes to operate on, and adding it to the plugin/translators directory. All translators found there will automatically over-ride *2dx_merge*’s default behavior. This could be used, for instance if one wished to open .aph files in a different browser than .txt files.

3.2. Visualization

Included with *2dx_merge* is the standard navigator available for *2dx_image* allowing users to view Fourier

and real space images. Since the initial release of the *2dx* processing platform several new improvements have been made to this viewer.

One such addition is that Fourier space images are viewable as complex numbers showing the amplitude and phase. As also possible in the MRC program Ximdisp, Fourier pixels are displayed via the HSV (Hue, Saturation, Value) table, where Saturation is set to 1, Value displays the amplitude, and Hue (based on a periodic color wheel) shows the phase (Fig. 3). This feature is useful to inspect synthetically generated references, and can be helpful for the manual selection of lattice spots (Zeng et al., in this issue-b).

The display of reconstructed projection maps indicates their dimensions and coordinates in phase origin space, i.e. x, y coordinates are mapped relative to the unit cell covering the range of -180° to 180° . This facilitates the manual adjustment of phase origin, via single click, to aide in choosing symmetry centers for the merging process.

2dx_merge also features an image “photo album”, which displays images in a readily viewable grid, which could be useful for example to quickly assess the validity of the phase origins of the images (Fig. 4).

3.3. Inheritance

To facilitate efficient automated processing, a hierarchy of configuration files exists relative to the directory structure. The project root directory contains a master configu-

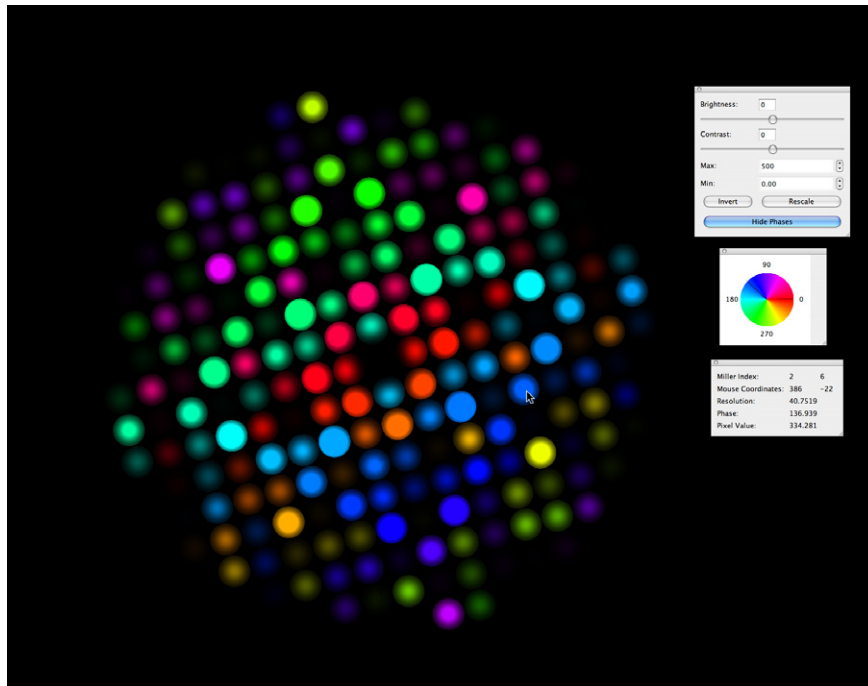


Fig. 3. The full-screen browser of *2dx_image* can display Fourier transform data with color-coded phase information. This example shows a synthetic reference generated in the “Synthetic Unbend” script by the MRC program MAKETRAN.

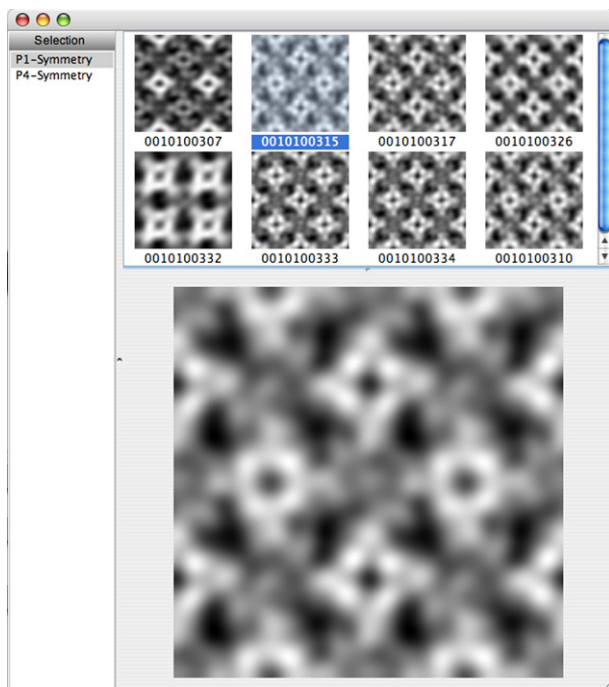


Fig. 4. Image album showing reconstructed projection maps from individual 2D crystal images of MloK1 (Chiu et al., 2007). In this example, all images are from non-tilted crystals of the same space group, and should show a similar projection map. Comparison of these maps allows identification of images of a different crystal form (e.g. the differently stained crystal in image 0010100307, first in first row), or wrongly assigned lattices (e.g. image 0010100332, first image in the second row) or wrongly assigned phase origins (e.g. image 001010333, second in second row).

ration file, called *2dx_master.cfg*. This file defines the parameters that are valid for the entire project, like crystal space group, and real space crystal lattice dimensions. The merge directory contains a merge parameter set, *2dx_merge.cfg*. Each tilt-range directory contains another *2dx_master.cfg* parameter file, which is used to provide default values for a number of parameters that should be used upon initialization of a new image directory in that tilt-range (e.g. default defocus settings or magnification). These copies of *2dx_master.cfg* could also be symbolic links to the upper-level *2dx_master.cfg* file. Finally, each image directory contains a local image parameter set, *2dx_image.cfg*, governing local image parameters.

The *2dx* software system applies two types of inheritance, called Initialization and Sync. Sync inheritance keeps selected parameters constantly in line with the configuration file in the immediately superior directory, while Initialization inheritance parameters are copied only on the first image directory initialization. This can be useful, if, for example, all images of 30° tilted samples were recorded on a 300 kV instrument at 1000 nm defocus, and digitized at 5 μm pixels size, while all non-tilted images were recorded at 200 kV and 500 nm defocus, and digitized at 7 μm pixel size.

The behavior for Initialization and Sync of each variable is defined in the configuration files themselves. It allows permanently synchronizing certain parameters (e.g. space group), or preparing the default configuration file for a newly imported image such that the fully automatic processing in *2dx_image* has the highest chance of success.

4. Workflow

4.1. Implemented algorithms

As with *2dx_image*, we have provided a collection of standard and specific scripts designed for the merging of 2D crystal images, which are mostly based on the MRC programs. These scripts are in editable text form, and can easily be replaced or modified by the user.

2dx_merge fulfills two functions, the management of a processing project, and the merging of the data. The latter is done by processing several images, using their Fourier-filtered maps as local references. *2dx_merge* then creates an initial merged dataset, which can be used to create a synthetic reference for refined unbending of all images. This iterative refinement process should continue, while new data are added to the process.

As implemented in the MRC programs, each reflection from each image is accompanied by a figure-of-merit (FOM) value, which reflects the reliability of an individual

amplitude and phase measurement. During merging of a large number of images, amplitude and phase values are weighted according to these FOM values. This has the effect that up to a certain extent the user can merge high-resolution and low-resolution images together, without running the risk that the lower-resolution images destroy the high-resolution data of other images. Nevertheless, the user can include or exclude data via the selection criteria.

Available image sub-directories are displayed in the central pane of *2dx_merge*. Double-clicking on one row in that pane will launch *2dx_image* in that image directory, where the user can then process one image either manually or automatically with the *2dx_image* program, using the scripts Unbend-I and Unbend-II. A switch in the database decides if these scripts operate with static CTF correction, or make use of the “tt” set of MRC programs for correction of the tilted transfer function (TTF), as depicted in Fig. 5, top panels. After having processed (or re-processed) a sufficient number of images, the user can return to the

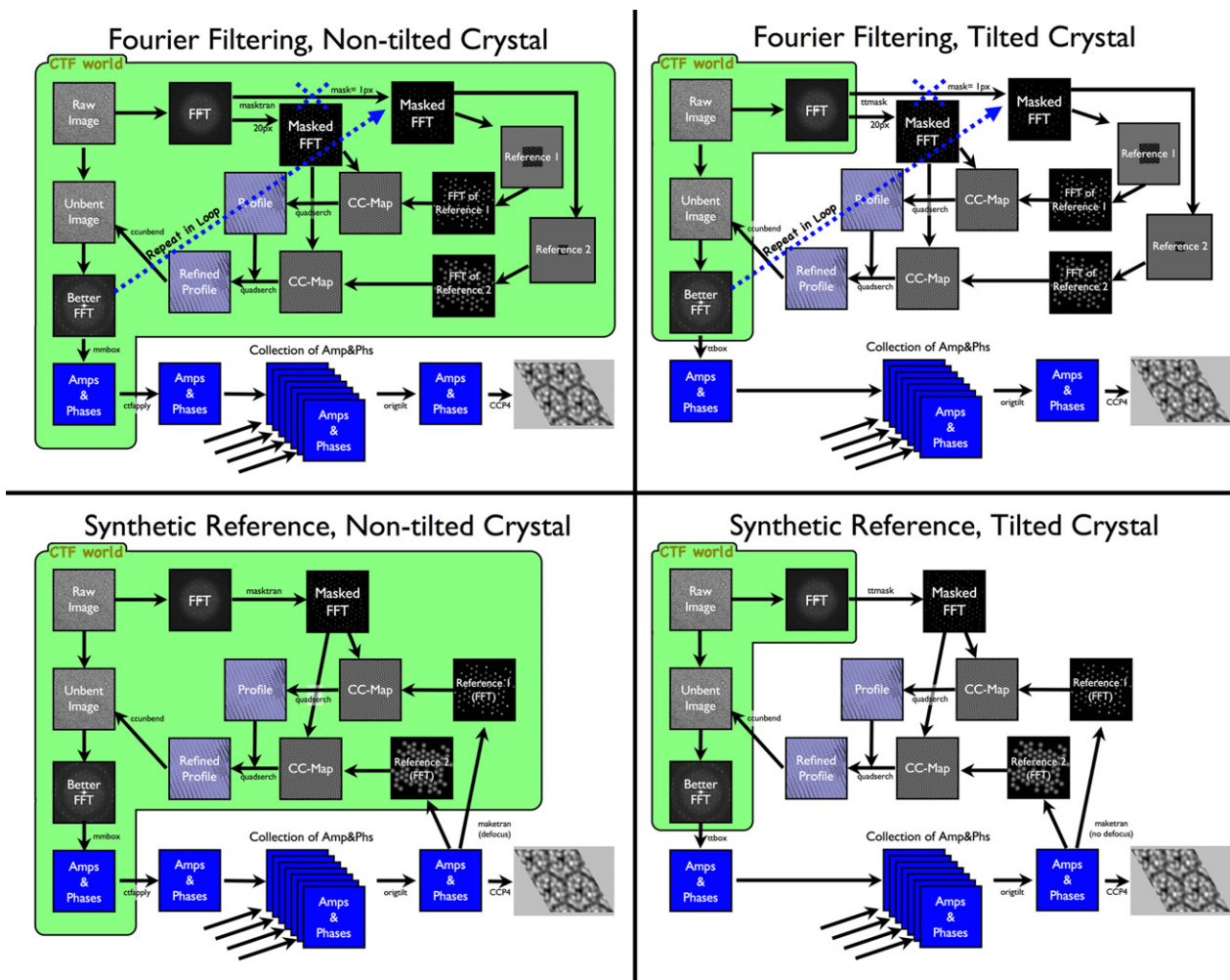


Fig. 5. Four algorithms implemented in the *2dx* software suite, displayed for images of non-tilted (left) or tilted (right) samples, and using the Fourier-filtered image itself (top) or a synthetically generated map (bottom) as reference for the unbending. The processing steps that deal with CTF-modulated data are located in the fields labeled by “CTF world”.

2dx_merge program. Clicking on the top right “re-scan” button will refresh the central image directory pane, and update the displayed values in that table. This set of images may then be merged into either a 2D merged projection map dataset, or a 3D volume dataset. In the currently released version, *2dx_merge* supports the merging into a 2D merged dataset. Extension of the scripts to include 3D volume merging functionality is ongoing. A first preliminary 3D merging function is included in the released software.

The user can choose a sub-set of images to merge by selecting their image directories by mouse-click on the button in front of each image sub-directory row in the central pane (Fig. 1). The currently selected group of images can then be merged, by sequentially following the “Standard Scripts” in the top left pane of the GUI:

First Merge. This script merges the currently selected images into a merged file, which will be created in the merging directory, and will be called “~/MYPROT/merge/merge.mtz”. This “First Merge” merges the available images together using their current phase origins.

Merge & Refine. This script allows the merging of selected images into one merge.mtz merged dataset (in repetition of the previous script), and subsequently uses this merged dataset as reference to refine the parameters of the individual images. This script can be executed a pre-set number of times, using the iteration counter that appears to the right of the script name in the “Standard Scripts” pane. By default, this counter is set to 5, prompting *2dx_merge* to iteratively merge and refine the merging parameters 5 times, in each round using the latest merged dataset as reference for the refinement of phase origin and other parameters. Double-clicking the iteration counter allows the user to change this number.

Generate Image Maps. This script re-creates the projection maps for all selected images, using the refined phase origins and defocus values. These maps allow manual verification of the correctly defined phase origins, by manually browsing over the list of images in the “Images” pane and visually inspecting their image previews. This can be useful, for example, to verify all determined phase origins or handedness assignments.

Final Merge. This script is again a repetition of the merging script (as “First Merge”), and is used to calculate a final merged dataset to be used as reference for re-processing of all images (see below).

Generate Merged Map. This script finally allows the calculation of a resulting map of the merged dataset. This script concludes the merging phase of the chosen set of images.

Merging is based on a slightly adapted version of the MRC program *origtiltk.for*, here called *2dx_origtiltk.for*. We created a simple Fortran program *2dx_merge_compileA* that receives from *2dx_merge* a list of the directories that contain image data to be merged. *2dx_merge_compileA* will then extract the required parameters from the *2dx_image.cfg* datafiles in the rel-

evant image directories, and create as output a cshell script that contains a call to *2dx_origtiltk.for*, providing that program will all required parameters. That cshell is then launched and performs the merging of the data. A similar program *2dx_merge_compileB* prepares the cshell script for the refinement of variables, again based on *2dx_origtiltk.for*. The GUI of *2dx_merge* will then evaluate the results file of that running cshell script and update the relevant parameters (e.g. better phase origins) into the image datafiles *2dx_image.cfg*, or into the merging datafile *2dx_merge.cfg*. A third Fortran program *2dx_merge_compileM* is responsible for the compilation of the cshell script that re-creates all image maps.

These simple Fortran “helper programs” allow the usage of the MRC program *origtiltk.for* without the need to modify large fractions of that program. Future versions of the MRC program *origtiltk.for* can therefore be easier updated into *2dx*, and the functionality of iteratively re-merging of a growing number of images is maintained. *2dx_merge* provides in a local file *2dx_merge_dirfile.dat* a list of the selected directories, which the scripts can use for any required purpose. The “Cleanup” Custom-Script, for example, uses the list of provided directories to delete all temporary files in the specified image directories.

The remainder of the merging process is done by the MRC programs *avrgamphs.for*, *2dx_centric2.for* (based on *centric.for*, but now including systematic absences), and the CCP4 programs *f2mtz* and *sftools*. The scripts for the calculation of 3D lattice line values and 3D volume reconstruction are in preparation.

4.2. Guided merging

2dx_merge follows a similar approach in user-guidance as *2dx_image*: The user should sequentially run the “Standard Scripts” in the top left pane, to merge a dataset. However, while the processing of one 2D crystal image in *2dx_image* can often successfully be done in the fully automated mode, we recommend that the merging of image data should still be done interactively, executing each step individually.

For instance, the alignment phase of merging remains a subtle process requiring user verification of proper alignment. Generally this can be approached by limiting the upper resolution and using an appropriate phase origin search step size and total number of steps (MergeStep-Size = 6.0° and IBOXPHS = 61), followed by running 5 rounds of “Merge and Refine”. This should roughly align all images up to the specified resolution. This can be followed by further refinement rounds, while gradually increasing the merge resolution and reducing the phase origin search range. Eventually, an alignment with 61 steps of 0.1° step size should align the dataset with high accuracy. Since the difficulty of the search of the phase origin (image alignment) or the determination of the beam-tilt values for

each image varies from one project to the other, this step remains a largely user-guided process.

4.3. Re-unbending of images

The merged dataset in `~/MYPROT/merge/merge.mtz` is then available for the creation of synthetic references for a re-unbending of all individual images, using the MRC program `MAKETRAN` (Kunji et al., 2000). While the normal `2dx_image` Fourier-filtering as implemented in the `Unbend-I` and `Unbend-II` scripts uses repeated refinement of the unbending crystal distortion `PROFILE` to iteratively apply the improved Fourier-filtered reference, the process during synthetic unbending differs: Since the synthetically determined reference is assumed to be a “perfect” reference (or at least better than the information from one image alone could provide), iterative refinement of the reference is not done.

Instead, the script “Synthetic Unbend” in `2dx_image` (Fig. 2) will use the merged dataset to generate two synthetic reference images that will be used for the determination of the unbending profile of an individual image: the first synthetic reference should have dominant low-resolution features, which can be created by using a temperature factor of 0 and a smaller Fourier mask size. The second synthetic reference should then have a stronger presence of high-resolution features, which can be created by using a negative temperature factor and a larger Fourier mask size. (The larger Fourier mask corresponds to the choice of a smaller reference box size in the Fourier filtering procedure.) The script “Synthetic Unbend” will use the first reference to determine a first unbending profile. This profile will then be refined, using a restricted search radius from the identified `PROFILE` node locations, with the second “sharpened” synthetic reference. These algorithms for non-tilted and tilted images are depicted in Fig. 5, lower panels.

4.4. Iterative re-processing

Iterative processing within the `2dx` framework can now be done on the level of interplay between unbending, Maximum Likelihood processing, and merging: A first unbending run using a Fourier-filtered reference can be used to determine an initial `QUADSERCH` unbending profile in `2dx_image`. This profile contains the *X/Y* coordinates of the identified crystal unit cells, and is now also written out in `SPIDER` format, if the user wishes to continue with single particle processing in the `SPIDER` software. This profile can be used for the initial particle stack selection for the Maximum Likelihood (ML) processing within `2dx_image`, the result of which is then translated back into amplitudes and phases (APH-file). The APH-file of either the Fourier-filtered unbending or the post-processing by the ML program is then, together with the results from other images, used by the `2dx_merge` program to generate a merged dataset.

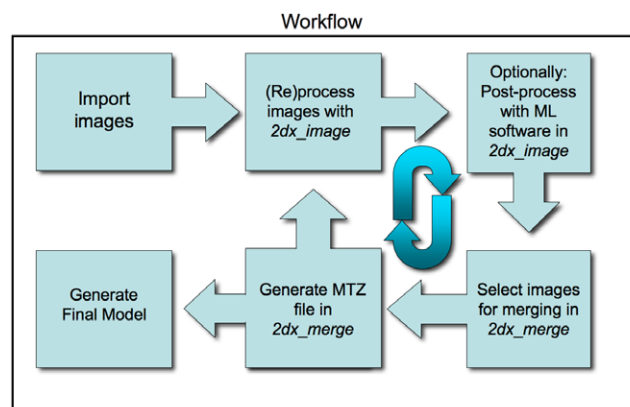


Fig. 6. Generic workflow in `2dx`, from images to reconstruction.

This merged dataset can then be used as reference for a refinement of the unbending process in `2dx_image` with the new standard-script “*Synthetic Unbending*”. This will generally result in a refined `QUADSERCH` profile, which in turn will allow a slightly better ML processing, finally resulting in a better-merged dataset, and so on. This processing can be applied iteratively, but bears the risk of locking in to noise correlation, if not handled carefully, similar to the behavior of single particle processing of noisy images (Grigorieff, 2000). This iterative application of the different algorithms, scripts and programs, while gradually including more images as they become available, can be performed until convergence is reached (Fig. 6).

4.5. Interaction with `2dx_image`

`2dx_merge` and `2dx_image` closely interact with each other. In `2dx_merge`, double-clicking on any of the image directory lines in the central pane opens an instance of `2dx_image`, allowing the user to process or modify variables for that image.

`2dx_merge` can also launch `2dx_image` as a GUI-free background process. For this, it makes use of a new feature of `2dx_image`, which can now be called from the command line with a script name (including wildcard matching) as parameter. `2dx_image` will then execute that script silently. For instance executing the `2dx_fftrans` script through `2dx_image` on a protein directory `~/MYPROT/MYPR-00/MYPROT001` from a `2dx_merge` script could be done with:

```
$app_2dx_image ~/MYPROT/MYPR-00/MYPROT001 '2dx_fftrans'
```

The first argument to `2dx_image` is the directory to be operated on. The second argument is a list of scripts (allowing wildcards), which are to be executed. Standard scripts are defined by their name (i.e. `2dx_fftrans`), custom scripts must be preceded by a “+”. Standard scripts listed in this manner will be executed according to their sort order while custom scripts will be executed in the order in which they are listed. Custom scripts are always executed after all standard scripts are completed.

The command:

```
$app_2dx_image ~/MYPROT/MYPR-00/MYPROTO01
‘‘*, +2dx_evaluateLattice’’ then would execute
all standard scripts according to their sort order and finally
execute 2dx_evaluateLattice (a custom script). This feature
prepares 2dx_image for remote execution on a computer
cluster.
```

5. Conclusions

2dx_merge facilitates the management of an electron crystallography project, and enables the merging of the collected and processed data. At the outset of each execution of *2dx_merge* an initialization script is called to ensure proper setup of all folders, hierarchies and configuration files. An initial merge step is used to create a first merged dataset. The “Merge & Refine” step then iteratively aligns images to the reference. After alignment, the reconstruction maps are re-generated by the “Generate Image Maps” script, and can be used to visually verify alignment. The last two scripts: “Final Merge” and “Generate Merged Map” generate the reconstruction, applying symmetry and resolution limits, from the aligned data. The merged dataset can then be used for the creation of a synthetic reference, to re-unbend all images in the *2dx_image* program, and to provide the Maximum Likelihood processing in *2dx_image* with better particle starting locations. Results of this re-processing of all images can be iteratively used for re-merging.

2dx is designed as a project oriented “meta-processor”, which organizes and executes script templates. The current package with included scripts allows users to process and merge electron crystallography data. Both, the individual processing steps and the over-all merging workflow are entirely customizable and should satisfy most merging needs. Default scripts follow the standard MRC merging path, but users are free to edit/modify/implement custom designs. 3D merging is in preparation. *2dx* is available under the GNU Public License at <http://2dx.org>.

Acknowledgments

This work was supported by the NSF, Grant No. MCB-0447860 and by the NIH, Grant No. U54-GM074929. We thank Remco Wouts, Werner Kühlbrandt, Anchi Cheng, Vinzenz Unger, and Tom Walz for fruitful discussions, and Per Bullough for fruitful discussions and help with symmetry definitions and systematic absences. Development of some of the underlying scripts was started in the laboratories of Jacques Dubochet in Lausanne, and Andreas Engel in Basel, Switzerland.

References

- Aebi, U., Smith, P.R., Dubochet, J., Henry, C., Kellenberger, E., 1973. A study of the structure of the T-layer of *Bacillus brevis*. *J. Supramol. Struct.* 1, 498–522.
- Chiu, P.-L., Pagel, M., Evans, J.E., Chou, H.-T., Zeng, X., Gipson, B., Stahlberg, H., Nimigeon, C.M., 2007. The structure of the prokaryotic cyclic nucleotide-modulated potassium channel MloK1 at 16 Å resolution. *Structure* 15 (9), 1031–1039.
- Crowther, R.A., Henderson, R., Smith, J.M., 1996. MRC image processing programs. *J. Struct. Biol.* 116, 9–16.
- Fujiyoshi, Y., Mizusaki, T., Morikawa, K., Yamagishi, H., Aoki, Y., Kihara, H., Harada, Y., 1991. Development of a superfluid helium stage for high-resolution electron microscopy. *Ultramicroscopy* 38, 241–251.
- Gipson, B., Zeng, X., Zhang, Z.Y., Stahlberg, H., 2007. 2dx—User-friendly image processing for 2D crystals. *J. Struct. Biol.* 157, 64–72.
- Gonen, T., Cheng, Y., Sliz, P., Hiroaki, Y., Fujiyoshi, Y., Harrison, S.C., Walz, T., 2005. Lipid–protein interactions in double-layered two-dimensional AQP0 crystals. *Nature* 438, 633–638.
- Grigorieff, N., 2000. Resolution measurement in structures derived from single particles. *Acta Cryst. D Biol. Crystallogr.* 56 (Pt 10), 1270–1277.
- Gyobu, N., Tani, K., Hiroaki, Y., Kamegawa, A., Mitsuoka, K., Fujiyoshi, Y., 2004. Improved specimen preparation for cryo-electron microscopy using a symmetric carbon sandwich technique. *J. Struct. Biol.* 146, 325–333.
- Henderson, R., Unwin, P.N., 1975. Three-dimensional model of purple membrane obtained by electron microscopy. *Nature* 257, 28–32.
- Jap, B.K., Zulauf, M., Scheybani, T., Hefti, A., Baumeister, W., Aebi, U., Engel, A., 1992. 2D crystallization: from art to science. *Ultramicroscopy* 46, 45–84.
- Kühlbrandt, W., 1992. Two-dimensional crystallization of membrane proteins. *Quart. Rev. Biophys.* 25, 1–49.
- Kunji, E.R., von Gronau, S., Oesterhelt, D., Henderson, R., 2000. The three-dimensional structure of halorhodopsin to 5 Å by electron crystallography: a new unbending procedure for two-dimensional crystals by using a global reference structure. *Proc. Natl. Acad. Sci. USA* 97, 4637–4642.
- Levy, D., Chami, M., Rigaud, J.L., 2001. Two-dimensional crystallization of membrane proteins: the lipid layer strategy. *FEBS Lett.* 504, 187–193.
- Philippson, A., Schenk, A.D., Stahlberg, H., Engel, A., 2003. Iplt—image processing library and toolkit for the electron microscopy community. *J. Struct. Biol.* 144, 4–12.
- Remigy, H.W., Caujolle-Bert, D., Suda, K., Schenk, A., Chami, M., Engel, A., 2003. Membrane protein reconstitution and crystallization by controlled dilution. *FEBS Lett.* 555, 160–169.
- Renault, L., Chou, H.T., Chiu, P.L., Hill, R.M., Zeng, X., Gipson, B., Zhang, Z.Y., Cheng, A., Unger, V., Stahlberg, H., 2006. Milestones in electron crystallography. *J. Comput. Aided Mol. Des.* 20, 519–527.
- Schmidt-Krey, I., 2007. Electron crystallography of membrane proteins: two-dimensional crystallization and screening by electron microscopy. *Methods (San Diego, Calif.)* 41, 417–426.
- Signorell, G.A., Kaufmann, T.C., Kukulski, W., Engel, A., Remigy, H.W., 2007. Controlled 2D crystallization of membrane proteins using methyl-beta-cyclodextrin. *J. Struct. Biol.* 157, 321–328.
- Zeng, X., Stahlberg, H., Grigorieff, N., 2007a. A maximum-likelihood approach to two-dimensional crystals. *J. Struct. Biol.* 160, 360–372.
- Zeng, X., Gipson, B., Zhang, Z.Y., Renault, L., Stahlberg, H., 2007b. Automatic lattice determination for two-dimensional crystal images. *J. Struct. Biol.* 160, 351–359.