Available online at www.sciencedirect.com
ScienceDirect

Journal of Structural Biology 157 (2007) 64–72

Journal of Structural Biology

www.elsevier.com/locate/yjsbi

2dx—User-friendly image processing for 2D crystals

Bryant Gipson, Xiangyan Zeng, Zi Yan Zhang, Henning Stahlberg *

Molecular and Cellular Biology, University of California at Davis, 1 Shields Ave., Davis, CA 95616, USA

Received 11 May 2006; received in revised form 25 July 2006; accepted 29 July 2006
Available online 1 September 2006

Abstract

Electron crystallography determines the structure of two-dimensional (2D) membrane protein crystals and other 2D crystal systems. Cryo-transmission electron microscopy records high-resolution electron micrographs, which require computer processing for three-dimensional structure reconstruction. We present a new software system *2dx*, which is designed as a user-friendly, platform-independent software package for electron crystallography. *2dx* assists in the management of an image-processing project, guides the user through the processing of 2D crystal images, and provides transparence for processing tasks and results. Algorithms are implemented in the form of script templates reminiscent of c-shell scripts. These templates can be easily modified or replaced by the user and can also execute modular stand-alone programs from the MRC software or from other image processing software packages. *2dx* is available under the GNU General Public License at 2dx.org.

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

1. Introduction

Structural biology of membrane proteins is of central importance for health, disease, and the development of new drugs. Membrane proteins represent the majority of today's drug targets in pharmaceutical research. Nevertheless, the PDB database contains only a few hundred membrane protein structures, only a third of which can be considered unique conformations. Compared with the wealth of knowledge on the structure and function of soluble proteins, the low number of determined membrane protein structures stands in stark contrast to their biological importance.

Membrane protein structure determination faces several technical hurdles. Difficulties in over-expression, non-destructive detergent solubilization and gentle purification limit the amount of membrane protein sample available for structural studies. Structure determination by X-ray diffraction (XRD)[1] of three-dimensional (3D) crystals, nuclear magnetic resonance (NMR), and cryo-electron microscopy (cryo-EM) of two-dimensional (2D) crystals has revealed an amazing array of structural concepts and mechanisms that nature employs to solve the challenging tasks that membrane proteins perform. Recent highlights include the 1.35 Å structure by XRD of the ammonium channel AmtB (Khademi et al., 2004), the structure of the waterchannel Aqp0 from cryo-EM at 1.9 Å and XRD at 2.2 Å resolution (Gonen et al., 2005; Harries et al., 2004), and the structure of Mistic (Roosild et al., 2005) by NMR (Wüthrich, 1998), to name a few.

Electron crystallography uses cryo-electron microscopy to study the structure of membrane proteins that are reconstituted into phospholipid bilayers and laterally crystallized into 2D membrane protein crystals. Atomic models for seven membrane proteins and tubulin have been determined by electron crystallography: BR (Henderson et al., 1990) LHCII (Kühlbrandt et al., 1994), AQP1 (Murata et al., 2000; Ren et al., 2001), nAChR (Miyazawa et al., 2003), AQP0 (Gonen et al., 2004; Gonen et al., 2005), AQP4 (Hiroaki et al., 2006), and MGST1 (Holm et al., 2006), and Tubulin (Nogales et al., 1998). In addition, several low-resolution structures of transporters, ion pumps, receptors and membrane bound enzymes, that

* 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; NMR, nuclear magnetic resonance; cryo-EM, cryo-electron microscopy; MRC, Medical Research Council.

1047-8477/$ - see front matter © 2006 Elsevier Inc. All rights reserved.
doi:10.1016/j.jsb.2006.07.020

reveal secondary structural motifs such as transmembrane helices are likely to produce atomic models in the near future (e.g., Hirai et al., 2002; Schenk et al., 2005; Kukulski et al., 2005; Tate et al., 2003; Vinothkumar et al., 2005; Aller and Unger, 2006).

The crystallization of membrane proteins in a 2D array within the lipid bilayer represents a valuable alternative route for structure determination. Electron Crystallography has matured into a methodology that allows the determination of membrane protein structures at a resolution of 3 Å or better (e.g., Grigorieff et al., 1996; Mitsuoka et al., 1999; Gonen et al., 2005). 2D membrane crystals offer the possibility of assessing membrane-inserted protein conformations. Existing 2D crystals can be incubated with ligands or other protein binding partners, or they can be exposed to different buffer conditions, and the structure of the complex or altered conformation can then be studied by electron diffraction. However, electron crystallography remains a labor-intensive method: beam-induced charging and/or drumhead-type movement of tilted samples in the electron microscope still affect the success rate for recording high-resolution images—despite recent advances though the use of the SpotScanning method (Downing, 1991) and/or the sandwich sample preparation method (Gyobu et al., 2004). During the screening of crystallization conditions, high-resolution data collection or computer image processing, the lack of automation also requires time-intensive operator interaction.

Computer image processing of electron crystallography data in almost all the aforementioned cases has, to date, been performed by the "*MRC programs*" for image processing (Crowther et al., 1996). These "*MRC programs*" are a compilation of individual programs, most written in Fortran-77, that were designed to process images of two-dimensional crystals as well as electron diffraction patterns (Unwin and Henderson, 1975; Henderson et al., 1990; Kühlbrandt et al., 1994; Murata et al., 2000). While this software collection offers a vast repertoire of tools for the processing of 2D crystal images, learning how to employ these programs is time-intensive, and the their usage involves a high amount of direct user interaction.

The *MRC programs* and *bsoft* programs (Heymann, 2001) are a collection of stand-alone programs written in Fortran-77 or C/C++. These programs need to be executed either manually, one-by-one in a terminal window, or from a shell script. The later has the advantage of facilitated usage, along with high flexibility and adaptability, but maintaining such scripts can be labor intensive. The execution speeds of computational tasks in scripts are slow, and readability of the scripts and interpretation of results in the form of log-files can be difficult.

A number of other software packages exist for the processing of 2D crystal images. SPECTRA from the ICE package facilitates the usage of the *MRC* software (Schmid et al., 1993; Hardt et al., 1996). Wilko Keegstra at the University of Groningen, The Netherlands, is currently developing the Groningen Image Processing Package (GRIP) that can also interface with the *MRC* software (unpublished). The Image Processing Library and Toolkit (*IPLT*) is a new ground-up image processing development for electron crystallography (Philippsen et al., 2003).

We present a new software system, ***2dx*** that is designed for the electron crystallography community. The purpose of this software system is to facilitate and streamline the processing of electron crystallography data, by providing a user-friendly interface, user-guidance throughout data processing, and a high degree of automation. In the current implementation, *2dx* utilizes programs from the *MRC* software, as well as additional stand-alone programs written specifically for interaction with the *2dx* environment as well as providing additional functions and features. *2dx* is highly dynamic and can easily be used in conjunction with other image processing packages, including *IPLT* (Philippsen et al., 2003), *bsoft* (Heymann, 2001), and/or *Spider* (Frank et al., 1996). *2dx* is developed under the Gnu Public License (GPL), and is freely available as open source. *2dx* is available at http://2dx.org and runs natively on Mac OSX and Linux/X11 (Linux, IRIX and other Unix variants).

2. Software design

2dx is a collection of five programs, *2dx_manager, 2dx_image, 2dx_diffraction, 2dx_merger* and *2dx_logbrowser* (Fig. 1). *2dx_manager* assists in the management of an image-processing project, which typically amounts to 3D structure determination of one membrane protein. *2dx_manager* maintains control over the existing data (images or diffraction pattern), their parameters (e.g., resolution, sample tilt geometry) and results. *2dx_manager* also launches other programs such as *2dx_image* and *2dx_diffraction* as interactive instances, or submits them to a distributed computing cluster. *2dx_merger* manages 2D and 3D merging of the data. The *2dx_diffraction* program will perform the computer evaluation of electron diffraction patterns where *2dx_image* performs the processing of one image of a 2D crystal. *2dx_logbrowser* assists in analyzing the log-files that result from processing. The *2dx_merger*

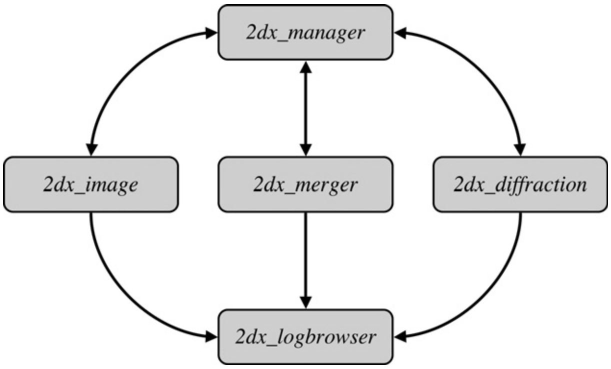


Fig. 1. The five programs of the *2dx* package. *2dx_manager* coordinates the project, and launches the *2dx_image* and *2dx_diffraction* programs for the processing of images and diffraction patterns. Data will be merged by *2dx_merger*. *2dx_logbrowser* assists in the evaluation of the log-files.

and *2dx_diffraction* programs are currently under development, while *2dx_manager* at present assists only in the initialization of a project directory structure (Fig. 2). Here we introduce the programs *2dx_image* and *2dx_logbrowser*.

2dx_image and *2dx_logbrowser* are written in C++, and are based on the Qt Open Source Edition for cross-platform software development (Trolltech, http://www.trolltech.com), and FFTW (Frigo and Johnson, 2005; http://www.fftw.org). *2dx* as well as FFTW are available under the GNU GPL, and Qt is available open source, free of charge for non-commercial software.

The central philosophies guiding the development of the *2dx* software have been ease of use and independence from particular algorithmic implementations and/or platforms. To this end we have developed the software to be intuitive and automatic. That is, users do not need advanced knowledge about the technical details of the image processing in order to process a 2D crystal image in a straightforward way. Ideally, once a few essential parameters, such as the image file name and other parameters concerning the protein, are known and submitted, the software is capable of processing an image from start to finish with no further need for user interaction. Unfortunately, such automated designs easily lead to a trade-off between ease-of-use and processing precision. *2dx* is therefore designed with a high degree of flexibility and customizability, rooted in ground-up platform independence.

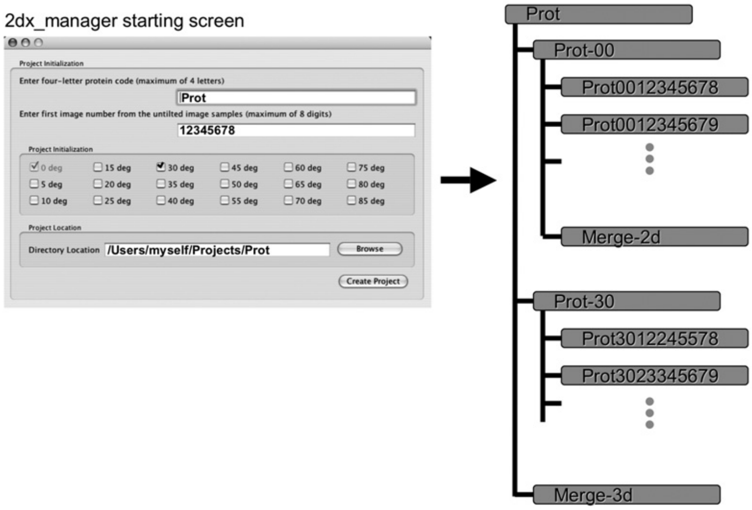
Excellent image processing packages, such as *MRC, IPLT,* and *bsoft* contain numerous efficient, rigorous routines, each with their own benefits. We have kept the *2dx* front-end GUI implementation independent from the software backend, relying on low-level algorithmic templates (reminiscent of c-shell style scripts), which organize processing procedures around modular programs. A processing routine is then subject only to the confines of the modules on which it depends, each of which can be easily replaced as needed. Further, since procedural level changes amount only to modification of template files, large structural changes in workflow become little more than script editing.

The defining features of a template file include a variables section, describing parameters necessary for the execution of the script; a script section, describing the actual program flow; and a series of simple semaphore, which allow communication with the GUI front end (Fig. 3).

Parameters found in the variables section of a template are drawn from a configuration file containing all variables necessary to execute the script. Variables appearing in this configuration file are distinguished by unique identifiers and defined by a human readable data structure, which describes every aspect of the variable's appearance in the GUI. This structure allows control over how the user will interact with the variable through the front end, in addition to providing basic information about the variable itself. The variable's 'LEGEND' value, for instance, contains a brief line of text describing the meaning of the parameter, whereas the 'HELP' value contains an html link, which points to a more detailed discussion of the variable on the *2dx.org* web server. Each help description page on the *2dx.org* server features a discussion thread in the form of an online blog, so that users can discuss their experiences or questions regarding the *2dx.org* documentation (Renault et al., 2006).

Since the content of the configuration file defines the appearance of the *2dx_image* GUI (and is designed with readability in mind) adding, deleting and reorganizing processing parameters and their layout can be easily achieved. Even large structural changes in the layout and appearance of the GUI can be done by editing this configuration file.

The executable portion of any template generally corresponds to a c-shell script in flow and syntax. Since neither a



Project Initialization

Enter four-letter protein code (maximum of 4 letters): Prot

Enter first image number from the untilted image samples (maximum of 8 digits): 12345678

Project Initialization

☑ 0 deg □ 15 deg ☑ 30 deg □ 45 deg □ 60 deg □ 75 deg
□ 5 deg □ 20 deg □ 35 deg □ 50 deg □ 65 deg □ 80 deg
□ 10 deg □ 25 deg □ 40 deg □ 55 deg □ 70 deg □ 85 deg

Project Location

Directory Location: /Users/myself/Projects/Prot

Fig. 2. *2dx_manager* in its current state assists in the generation of a default directory structure for a protein project, which here is called "Prot". Images should reside in their own dedicated directory (e.g., Prot0012345678), which are grouped according to their nominal tilt angles (here: non-tilted in "Prot-00", and 30-deg tilted in "Prot-30"). Merging directories for the 2D merging of the non-tilted images, and for the 3D merging of the entire project are also provided.

```
#!/bin/csh -e
###############################################################################
# Title: Determine Spacegroup  (simplified version)                           #
###############################################################################
#
# SORTORDER: 45
#
#==============================================================================
# SECTION: Allspace parameters
#==============================================================================
#
# LABEL: Spacegroups to test
# LEGEND: Choose the spacegroup groups that should be considered.
# EXAMPLE: test_spacegroups_val= "ALL"
# HELP: http://2dx.org/documentation/variable/allspace
# TYPE: Drop_Down_Menu "ALL;HEXA;SQUA;RECT;OPLI;TWO;THRE;SIX;FOUR"
set test_spacegroups_val = "ALL"
#
# GLOBAL: RESMAX
#
#$end_local_vars
#
set RESMAX = ""
set realang = ""
set realcell = ""
#
#$end_vars
#
\rm -f 2dx_allspace.results
#
echo "<<@progress: 10>>"
#
2dx_allspacea.exe << eot
${test_spacegroups_val}
T T F 4000                        ! SEARCH,REFINE,TILT,NCYC
0. 0. 0. 0.                       ! ORIGH,ORIGK,TILTH,TILTK
3.0,121                           ! STEPSIZE, PHASE SEARCH ARRAY SIZE
${realcell} ${realang} 200.0 ${RESMAX} 2.0, 200.0
F 0 5                             ! ILIST,ROT180,IQMAX
"SPCGRP.txt"
eot
#
echo "<<@progress: 80>>"
#
set SPCGRP = `cat SPCGRP.txt | cut -d\  -f1`
set PHAORI = `cat SPCGRP.txt | cut -d\  -f2-`
#
echo "::Spacegroup ${SPCGRP} is best."
echo "set SYM = ${SPCGRP}" >> 2dx_allspace.results
echo "set phaorip1 = ${PHAORI}" >> 2dx_allspace.results
echo "# IMAGE: outputimage.mrc" >> 2dx_allspace.results
#
echo "<<@progress: 100>>"
```

Fig. 3. An example for the script template used by *2dx_image*. This c-shell template contains code words that control the widget generation in the graphical user interface (GUI) of *2dx_image*. *"# Title:"* and *"# SORTORDER:"* allow the definition of the title and order under which the script will appear in the GUI. *"# SECTION:"* signals the beginning of a new parameter section in the central panel of the GUI. The following 6 lines define one parameter entry for that pane: *LABEL* is the title of the parameter, *LEGEND* is the short explanation in the pop-up window associated with that parameter. *EXAMPLE* allows suggesting the syntax for a correct entry. *HELP* defines the web page, where online help can be found. *TYPE* instructs the GUI to construct the widget for this parameter in a specific way (here as *Drop_Down_Menu*). Finally, "*set test_spacegroups_val* =" defines the default value for that parameter. The following section with the code words *"# GLOBAL:"* requests other globally known parameters that should appear in the GUI (here only RESMAX). This section is terminated with the flag *"#$end_local_vars"*. The following section requests parameters, which the GUI will enter when translating this script template into the actual executable script. In this example, *"realang"* and *"realcell"* will not be editable in the GUI for this script, because they are not declared as *"# GLOBAL:"*. However, these values will available for this script. This section terminates with *"#$end_vars"*. The remainder of the script template is a normal c-shell script. The output of the command *echo "≪@progress: 10≫"* will cause the GUI to advance the progress bar, setting it here to 10% of the execution progress. Logfile output starting with "::" will be displayed by the GUI also under only the lowest verbosity settings. ":" defines moderate verbosity output, and lines without leading colons appear only under highest verbosity settings. Output into the file *2dx_allspace.results* (<filename>.results) in the form of, for example, *"echo "set SYM* = ${*SPCGRP*}*" ≫ 2dx_allspace.results"* would return a new value for the parameter *SYM* to the GUI, which would store it in the *2dx_image.cfg* database. The results file can also be used to flag image files that should appear in the list of images for inspection. *"echo "# IMAGE: outputimage.mrc" ≫ 2dx_allspace.results"* in this example instructs the GUI to include this image file in the list of viewable images. The final command *"echo "≪@progress: 100≫"* advances the progress bar to 100%.

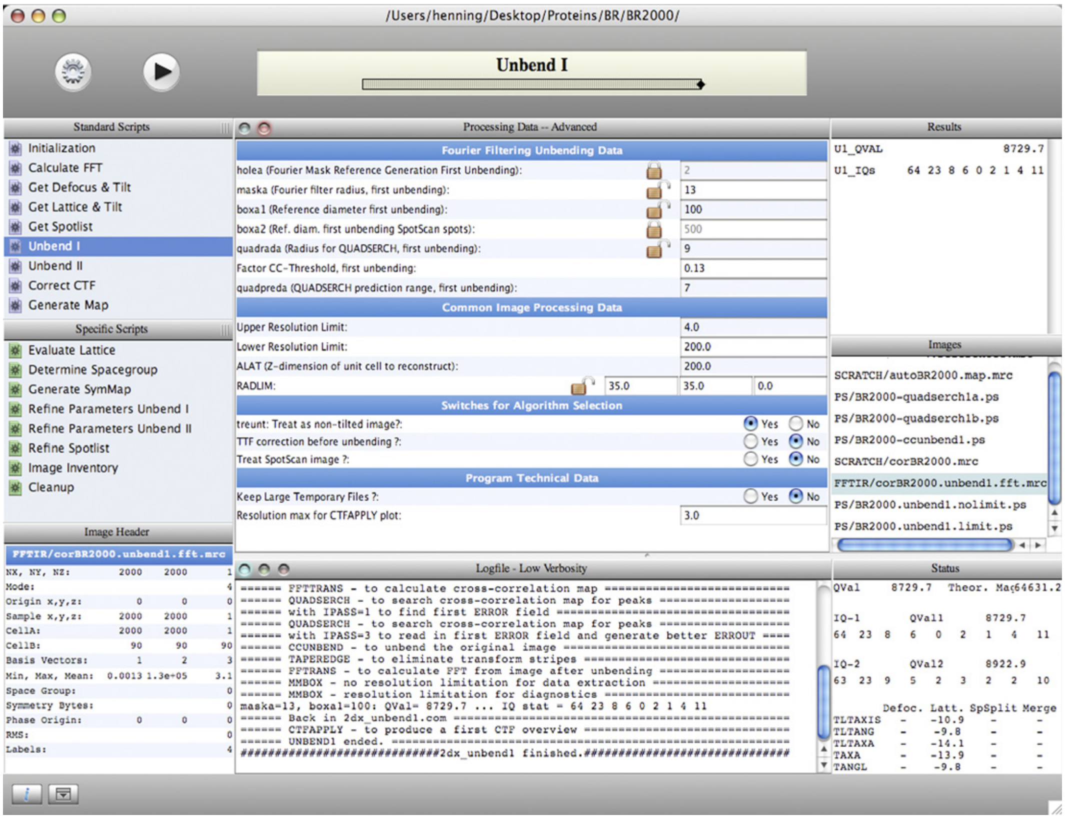
parameter section nor use of semaphore is required for any template, the user is free to incorporate any existing c-shell script they wish into *2dx* with a minimum of alteration.

3. Graphical user interface and work flow

In its current state, the *2dx_manager* assists in the generation of a directory structure for a 2D crystal project (Fig. 2). A four-letter project code and the image number of the first non-tilted image are requested, together with a selection of sample tilt ranges that the user intends to use for data collection. The *2dx_manager* then initializes the directory structure as reproduced in Fig. 2, to be used in the following conventions: Each 2D crystal image should be processed in its own directory, where the image file, its parameter files and output files are stored. Image directories are grouped according to their nominal tilt angle, starting with one directory for all images of non-tilted samples. Residing in the image directories of non-tilted samples is a merge-directory that can be used to generate a 2D merge dataset. Other tilt-angle sessions are organized in their respective directory structures, and the entire project is merged into a 3D dataset in the highest-level merge directory.

The purpose of *2dx_image* is the processing of one 2D crystal image, which resides in its own dedicated directory. *2dx_image* maintains a simple image database in the form of a structured text file (2dx_image.cfg), where all parameters relevant to the processing of that image are stored. Certain project-wide "global" parameters in this text file, such as the crystal symmetry or the real-space unit cell dimensions of the protein crystal, are synchronized at run time of the *2dx_image* program with a project-wide default configuration file.

The *2dx_image* main graphical user interface is reproduced in Fig. 4. The top section houses buttons to "Save" the image parameter file, and to "Execute" one or several selected script(s). This section also displays the currently running script, and its execution status. The central pane, entitled "Processing Data", displays the image and processing parameters—all of which can be edited, saved, and optionally locked to protect against accidental changes by the user or from changes made by executed programs. Two user-levels can be chosen, to allow access to only the most significant parameters ("simple"), or to the full-parameter set ("advanced"). The top left pane entitled "Standard Scripts" lists a set of scripts that are usually sequentially executed when one image is to be processed. This can be done by selecting one script at a time (mouse-click on the script), and executing it via the "Execute" button. Alternatively, any subset of these scripts can be selected and automatically executed sequentially. Upon execution, *2dx_image* loads the template, complements it with the template-requested data from the



Fourier Filtering Unbending Data

holea (Fourier Mask Reference Generation First Unbending): 2
maska (Fourier filter radius, first unbending): 13
boxa1 (Reference diameter first unbending): 100
boxa2 (Ref. diam. first unbending SpotScan spots): 500
quadrada (Radius for QUADSERCH, first unbending): 9
Factor CC-Threshold, first unbending: 0.13
quadpreda (QUADSERCH prediction range, first unbending): 7

Common Image Processing Data

Upper Resolution Limit: 4.0
Lower Resolution Limit: 200.0
ALAT (Z-dimension of unit cell to reconstruct): 200.0
RADLIM: 35.0 35.0 0.0

Switches for Algorithm Selection

treunt: Treat as non-tilted image?: ● Yes ○ No
TTF correction before unbending ?: ○ Yes ● No
Treat SpotScan image ?: ○ Yes ● No

Program Technical Data

Keep Large Temporary Files ?: ○ Yes ● No
Resolution max for CTFAPPLY plot: 3.0

```
U1_QVAL                         8729.7
U1_IQs       64 23 8 6 0 2 1 4 11
```

```
SCRATCH/autoBR2000.map.mrc
PS/BR2000-quadserch1a.ps
PS/BR2000-quadserch1b.ps
PS/BR2000-ccunbend1.ps
SCRATCH/corBR2000.mrc
FFTIR/corBR2000.unbend1.fft.mrc
PS/BR2000.unbend1.nolimit.ps
PS/BR2000.unbend1.limit.ps
```

```
FFTIR/corBR2000.unbend1.fft.mrc
NX, NY, NZ:         2000    2000     1
Mode:                                4
Origin x,y,z:          0       0     0
Sample x,y,z:       2000    2000     1
CellA:              2000    2000     1
CellB:                90      90    90
Basis Vectors:         1       2     3
Min, Max, Mean:   0.0013 1.3e+05   3.1
Space Group:                         0
Symmetry Bytes:                      0
Phase Origin:          0       0     0
RMS:                                 0
Labels:                              4
```

```
====== FFTTRANS - to calculate cross-correlation map ==========================
====== QUADSERCH - to search cross-correlation map for peaks ==================
====== with IPASS=1 to find first ERROR field ================================
====== QUADSERCH - to search cross-correlation map for peaks ==================
====== with IPASS=3 to read in first ERROR field and generate better ERROUT ====
====== CCUNBEND - to unbend the original image ===============================
====== TAPEREDGE - to eliminate transform stripes ============================
====== FFTRANS - to calculate FFT from image after unbending ==================
====== MMBOX - no resolution limitation for data extraction ==================
====== MMBOX - resolution limitation for diagnostics =========================
maska=13, boxa1=100: QVal= 8729.7 ... IQ stat = 64 23 8 6 0 2 1 4 11
====== Back in 2dx_unbend1.com ===============================================
====== CTFAPPLY - to produce a first CTF overview ============================
====== UNBEND1 ended. ========================================================
#############################2dx_unbend1 finished.##############################
```

```
QVal     8729.7   Theor. Ma¢64631.2

IQ-1          QVal1       8729.7
64  23  8   6   0   2   1   4   11

IQ-2          QVal2       8922.9
63  23  9   5   2   3   2   2   10

           Defoc. Latt. SpSplit Merge
TLTAXIS     -     -10.9    -      -
TLTANG      -      -9.8    -      -
TLTAXA      -     -14.1    -      -
TAXA        -     -13.9    -      -
TANGL       -      -9.8    -      -
```

Fig. 4. The *2dx_image* graphical user interface. For a description see text. The bottom left pane displays either the image file header information, as in this example, or the image thumbnail preview.

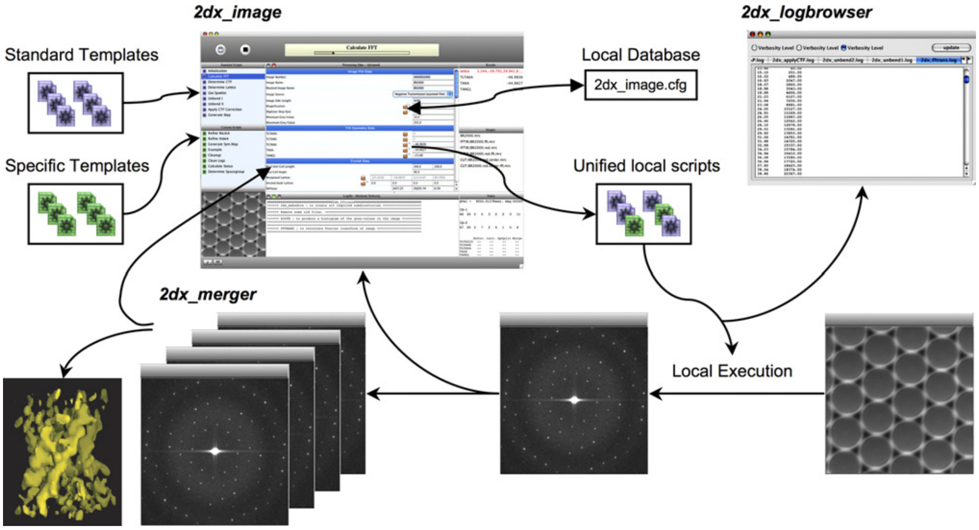


Fig. 5. The *2dx_image* workflow. *2dx_image* maintains a local database (2dx_image.cfg), with which script templates are translated into local executable scripts. These scripts can be launched, with their output and processing results then channeled back into *2dx_image*.

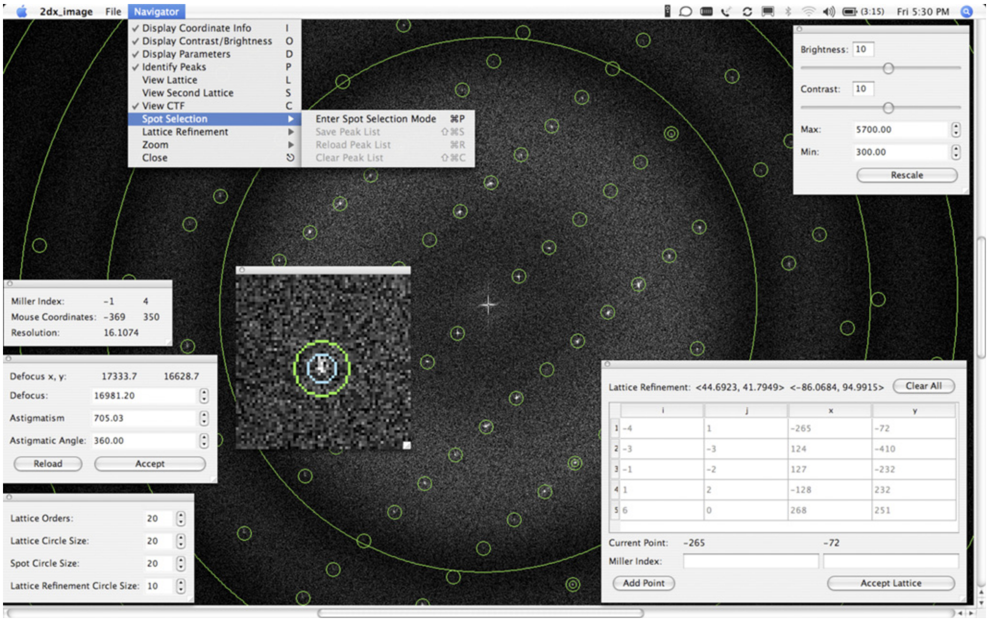
database, creates an executable c-shell command file in the local directory, and launches that command file as an independent child process (Fig. 5). The progress of the executing job is monitored and graphically displayed in the top-banner of the *2dx_image* program, which also allows the user to halt the running task. Output of the running job is displayed in the lower central pane of the *2dx_image* GUI, entitled "Logfile". Display of the job output has one of three verbosity levels, with the user being able to switch between levels by selecting one of three buttons on the top banner of that pane. Double-clicking this top banner launches the *2dx_logbrowser*, which allows the user to browse all available logfiles, each with the choice of three verbosity levels.

Running jobs can signal to the *2dx_image* GUI the names of important image-files, by including the label "# IMAGE:" in the log-output. Image files on the hard-drive that are flagged in this way are listed in the panel on the center right of the *2dx_image* GUI, entitled "Images". For example, the log-file entry "# IMAGE: SCRATCH/corBR2000.mrc" would add that MRC-format file to the list of image files in that pane. Currently, both MRC-format and PostScript format files are viewable. Their format is recognized by the ending of the file name. Selecting one of these image files in the *2dx_image* GUI launches the creation of a thumbnail preview of that image, which is displayed in the lower left pane of the GUI. Alternatively, the user can switch from the thumbnail view to a header-view, by selecting the button at the lower left end of the GUI, entitled "*i*". Double-clicking an image name or the thumbnail preview launches a full-screen image browser for that image (Fig. 6). This browser displays images and Fourier transformations, and also allows the user to manually adjust or edit, and save the reciprocal lattice in a Fourier transform, the Fourier spotlist, and the defocus.

During the execution of jobs, determined or refined parameters can be returned to the *2dx_image* database, by writing them into a results file. A script "2dx_allspace.com" for example can output a determined space group and phase origins by creating a file named "2dx_allspace.results", which should contain entries of the form "*set SYM* = *p3*" (Fig. 3). *2dx_image* will then interpret the results file and update the database accordingly. Selection of the "lock" icon located next to the entered values in the central pane will prevent a running script from updating specific parameters in the database. This would, for example, be useful if a user spent time and energy to manually determine the reciprocal lattice of a difficult Fourier transformation, and did not want the automatic lattice determination routine to overwrite the manually fine-tuned lattice vectors.

The bottom right pane of the *2dx_image* GUI displays the processing "Status" for the current image. This pane summarizes the most important parameters of the current image-processing job, which are maintained in a file named "2dx_image.status". These parameters include the quality value of the entire processing (*QVal*, see below), the refined theoretical magnification (for comparison with the nominal magnification, to indicate possible errors in pixel size, magnification or lattice vector dimensions), the statistic of IQ-values as defined by R. Henderson et al. (Henderson and Unwin, 1975; Henderson et al., 1990), as well as the five parameters describing the tilt geometry of the sample and the crystal, as determined by four different methods. The data in this "Status" pane informs the user about the status of the processing of this image, and indicates possible errors in the processing. Discrepancies in the tilt geometry between values determined from defocus variations across the image, distortions of the reciprocal lattice, spot-splitting due to the tilted transfer function (Henderson et al., 1990) and those refined during merging can be identified here.

Most of the fields, labels and names in the *2dx_image* GUI have a context-sensitive right-mouse-click activated



Brightness: 10

Contrast: 10

Max: 5700.00

Min: 300.00

Miller Index: -1 4

Mouse Coordinates: -369 350

Resolution: 16.1074

Defocus x, y: 17333.7 16628.7

Defocus: 16981.20

Astigmatism 705.03

Astigmatic Angle: 360.00

Lattice Orders: 20

Lattice Circle Size: 20

Spot Circle Size: 20

Lattice Refinement Circle Size: 10

Lattice Refinement: <44.6923, 41.7949> <-86.0684, 94.9915>

	i	j	x	y
1	-4	1	-265	-72
2	-3	-3	124	-410
3	-1	-2	127	-232
4	1	2	-128	232
5	6	0	268	251

Current Point: -265 -72

Miller Index:

Fig. 6. The *2dx_image* full-screen browser, here displaying a Fourier transformation of an image. A pull-down menu allows activating various panels. The Coordinate Info panel displays the current mouse coordinates, the corresponding resolution, and the Miller indices of the closest lattice spot. The Contrast/Brightness panel allows adjusting the display parameters. The Parameters panel allows defining the dimensions of the different symbols. The spots in the current spotlist are displayed when the "Peaks" option is selected. The current Lattice and Second Lattice can be displayed, as well as the Thon rings of the contrast transfer function (CTF), which is defined by the given defocus and astigmatism values. A Spot Selection mode and a Lattice Refinement mode allow manually editing or refining the spotlist and lattice vectors. The entire display can be zoomed up or down. Following the excellent development in the MRC program Ximdisp.exe, a mouse-activated local zoom window can be produced with the mouse. Determined values for the spotlist, the lattice vectors or the defocus values are automatically transferred back into the *2dx_image* GUI.

help function. Right-mouse-clicking a variable name in the parameter pane, for example, produces a window with a short explanation of that variable, its purpose and the units for the value, as well as an Internet link to the documentation in the corresponding web page on the *2dx.org* server.

4. Scripting conventions

The entire *2dx_image* construction is kept as user-adjustable and flexible as possible. The *2dx_image* database named "2dx_image.cfg", for example, is kept in a self-explanatory, editable text format. A user can easily add or delete variables, define their format (e.g., float, integer, pull-down menu, Boolean switch, etc.), and define the corresponding help information and web-page link. The standard and custom scripts can be modified, extended or replaced by other scripts that might launch other user-defined software. The format for reporting data to the *2dx_image* database (*.results) and for updating the status window is self-explanatory and easy to implement into existing software/scripts (see also Fig. 3).

5. Implemented algorithms

In the current state of *2dx_image* we have provided a collection of standard scripts for the processing of 2D crystal images, as we use them in our laboratory—most of which are based on the *MRC programs*. We also added functions for automatic lattice determination (Zeng et al., 2006), spot-list determination, and crystal masking, as well as for the determination of the tilt geometry (using *ctffind2*; Grigorieff, 1998). The need for the determination of the optimal reference patch location is eliminated by choosing a one pixel diameter Fourier mask in the first unbending round (*unbend1*): The resulting reference map is of low quality, but shows no deviation over the entire map. The reference patch can therefore be chosen in the center of that map. Further unbending rounds (*unbend2*) with wider Fourier masks will then retrieve the structure's underlying signal, while keeping the reference location in the center of the image.

Additional scripts are available in the lower left pane in the *2dx_image* GUI, entitled "Specific Scripts". For example, the script "Determine Spacegroup" allows the determination of the symmetry space group and/or phase origin for a given symmetry (using *allspace*; Valpuesta et al., 1994).

6. The quality value QVal

The scripts calculate a single, one-dimensional, value *QVal* that attempts to describe the quality of the entire image processing. While the IQ-values that were introduced by R. Henderson (Henderson and Unwin, 1975) are defined as a function of the intensity ratio between a specific reciprocal spot and its local background, the *QVal* addresses the entire image processing phase. *QVal* is calculated by an empirical formula that combines different performance measures and indicators into a single number. The library function *qval(nIQ,rscamax)* resides centrally in the file .../2dx/mrc/lib/2dx_func.for, and allows user fine-tuning of the formula used in calculating the QVAL, when a different function is desired. We currently employ the formula

$$QVAL = R * \{IQ1 * 17.5 + IQ2 * 12 + IQ3 * 8 + IQ4 * 5.2 + IQ5 * 3.3 + IQ6 * 2 + IQ7\}/500.0$$

where *IQ1* to *IQ7* denotes the number of IQ-values of the categories 1 to 7, and *R* denotes the height of the central pixel of the averaged Fourier peak profiles, as calculated, for example, by the MRC program *mmbox*. Division by the calibration factor 500.0 is done to allow easier display. This empirically derived formula corresponds to a more-than-linear weighting of the calculated diffraction power.

The *QVal* can also be reduced in the form of "penalty" points, if for example discrepancies between tilt geometries determined by different methods are encountered, or when other image processing parameters or results appear "suspicious".

The *QVal* can then be used by the user or by the *2dx_manager* program to judge the reliability or usability of an image for inclusion in the merging process. The *QVal* can also be used by the *2dx_image* program to automatically refine the image-processing task. The "Specific Scripts" "*Refine Parameters Unbend I*" and "*Refine Parameters Unbend II*" optimize the *QVal* during a systematic variation of parameters, and automatically determine the parameter combination that results in the highest *QVal*. Sensitive parameters like the Fourier mask radius (e.g., *maska*) or the diameter of the reference patch for cross-correlation in the unbending procedure (e.g., *boxa1*) can be systematically tested, and the optimized parameter setting can then automatically be saved and used in future processing. The parameter refinement scripts are computationally intensive, and can be applied to one representative image. The identified optimal parameters can then be saved as future default parameters for the processing of other images in the same tilt-angle group. Fig. 7 displays the result of a refinement of *maskb1*, for which the *QVal* was calculated for values between 1 and 30. Attention should be paid to exclude unrealistically small reference sizes, which can produce high *QVals* due to noise correlation. This "overfitting" of the unbending procedure would produce good IQ statistics and a high *QVal*, but does not improve the resolution of the image processing (see also Grigorieff, 2000). The single *QVal*-based refinement strategy can easily be adapted by the user to refine parameters for other scripts and/or programs, such as those that use *bsoft*, SPIDER, *IPLT* or other *MRC* programs (Heymann, 2001; Frank et al., 1996; Philippsen et al., 2003; Crowther et al., 1970).

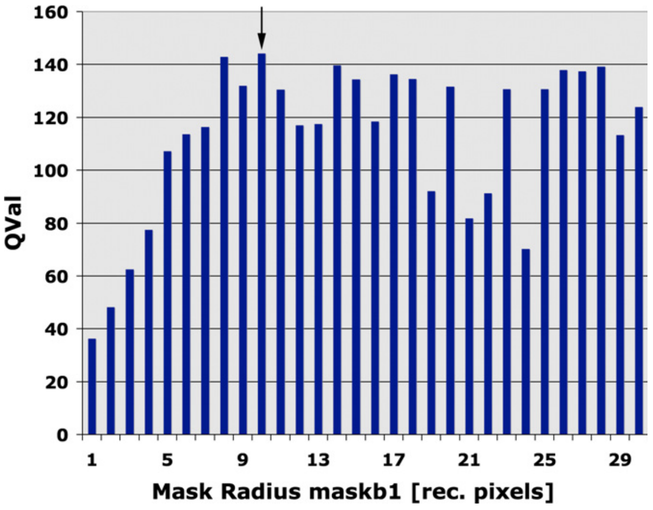


Fig. 7. The *QVal*-based parameter refinement. A search for the best parameter for *maskb1* resulted, in this example, in an optimal *QVal* value with *maskb1* = 10.

7. Conclusions

2dx is a user-friendly software system for electron crystallography. In its current state the components *2dx_image* and *2dx_logbrowser* allow the processing of 2D crystal images. Future development for electron diffraction pattern evaluation and 3D merging is under way. *2dx* is currently employed to run the "*MRC programs*" (Crowther et al., 1970), but can be used in conjunction with other systems. While the focus of *2dx* lies on user-friendliness, user-guidance, transparency, processing efficiency, and automation, we have implemented routines for the automatic determination of the crystal lattice, determination of the tilt geometry, spot-list creation, and crystal masking. Most of these implementations are based on the excellent developments of others (Henderson and Unwin, 1975; Henderson et al., 1990; Grigorieff, 1999; Philippsen et al., 2003; Heymann, 2001) and our aim has been to merge these into a user-friendly and efficient software system. Contributions in the form of additional user scripts or suggestions for additional functions are most welcome.

Acknowledgments

This work was supported by the NSF, grant number MCB-0447860 and by the NIH, Grant No. U54-GM074929. We wish to thank Per Bullough, Anchi Cheng, Andreas Engel, Bob Glaeser, Niko Grigorieff, Richard Henderson, Werner Kühlbrandt, Kaoru Mitsuoka, Ansgar Philippsen, Sriram Subramaniam, Vinzenz Unger, Jannet Vonck, and Tom Walz, who generously shared their knowledge and experience in image processing, and we thank Bob Glaeser and Rena Hill for comments on the manuscript. Development of some of the underlying scripts was started in the laboratories of Jacques Dubochet in Lausanne, and Andreas Engel in Basel, Switzerland.

References

Aller, S.G., Unger, V.M., 2006. Projection structure of the human copper transporter CTR1 at 6-Å resolution reveals a compact trimer with a novel channel-like architecture. Proc. Natl. Acad. Sci. USA 103 (10), 3627–3632.

Crowther, R.A., Amos, L.A., Finch, J.T., De Rosier, D.J., Klug, A., 1970. Three dimensional reconstructions of spherical viruses by fourier synthesis from electron micrographs. Nature 226, 421–425.

Crowther, R.A., Henderson, R., Smith, J.M., 1996. MRC image processing programs. J. Struct. Biol. 116, 9–16.

Downing, K.H., 1991. Spot-scan imaging in transmission electron microscopy. Science 251, 53–59.

Frank, J., Radermacher, M., Penczek, P., Zhu, J., Li, Y., Ladjadj, M., Leith, A., 1996. SPIDER and WEB: processing and visualization of images in 3D electron microscopy and related fields. J. Struct. Biol. 116, 190–199.

Frigo, M., Johnson, S.G., 2005. The design and implementation of FFTW3. Proc. IEEE 93, 231–261.

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.

Gonen, T., Sliz, P., Kistler, J., Cheng, Y., Walz, T., 2004. Aquaporin-0 membrane junctions reveal the structure of a closed water pore. Nature 429, 193–197.

Grigorieff, N., 2000. Resolution measurement in structures derived from single particles. Acta Crystallogr. D Biol. Crystallogr. 56 (Pt 10), 1270–1277.

Grigorieff, N., 1999. Structure of the respiratory NADH:ubiquinone oxidoreductase (complex I). Curr. Opin. Struct. Biol. 9, 476–483.

Grigorieff, N., 1998. Three-dimensional structure of bovine NADH:ubiquinone oxidoreductase (complex I) at 22 Å in ice. J. Mol. Biol. 277, 1033–1046.

Grigorieff, N., Ceska, T.A., Downing, K.H., Baldwin, J.M., Henderson, R., 1996. Electron-crystallographic refinement of the structure of bacteriorhodopsin. J. Mol. Biol. 259, 393–421.

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.

Hardt, S., Wang, B., Schmid, M.F., 1996. A brief description of I.C.E.: the integrated crystallographic environment. J. Struct. Biol. 116, 68–70.

Harries, W.E., Akhavan, D., Miercke, L.J., Khademi, S., Stroud, R.M., 2004. The channel architecture of aquaporin 0 at a 2.2 Å resolution. Proc. Natl. Acad. Sci. USA 101, 14045–14050.

Henderson, R., Baldwin, J.M., Ceska, T.A., Zemlin, F., Beckmann, E., Downing, K.H., 1990. Model for the structure of Bacteriorhodopsin based on high-resolution electron cryo-microscopy. J. Mol. Biol. 213, 899–929.

Henderson, R., Unwin, P.N., 1975. Three-dimensional model of purple membrane obtained by electron microscopy. Nature 257, 28–32.

Heymann, J.B., 2001. Bsoft: image and molecular processing in electron microscopy. J. Struct. Biol. 133, 156–169.

Hirai, T., Heymann, J.A., Shi, D., Sarker, R., Maloney, P.C., Subramaniam, S., 2002. Three-dimensional structure of a bacterial oxalate transporter. Nat. Struct. Biol. 9, 597–600.

Hiroaki, Y., Tani, K., Kamegawa, A., Gyobu, N., Nishikawa, K., Suzuki, H., Walz, T., Sasaki, S., Mitsuoka, K., Kimura, K., Mizoguchi, A., Fujiyoshi, Y., 2006. Implications of the aquaporin-4 structure on array formation and cell adhesion. J. Mol. Biol. 355, 628–639.

Holm, P.J., Bhakat, P., Jegerschold, C., Gyobu, N., Mitsuoka, K., Fujiyoshi, Y., Morgenstern, R., Hebert, H., 2006. Structural basis for detoxification and oxidative stress protection in membranes. J. Mol. Biol. 360 (5), 934–945.

Khademi, S., O'Connell 3rd, J., Remis, J., Robles-Colmenares, Y., Miercke, L.J., Stroud, R.M., 2004. Mechanism of ammonia transport by Amt/MEP/Rh: structure of AmtB at 1.35 Å. Science 305, 1587–1594.

Kühlbrandt, W., Wang, D.N., Fujiyoshi, Y., 1994. Atomic model of plant light-harvesting complex by electron crystallography. Nature 367, 614–621.

Kukulski, W., Schenk, A.D., Johanson, U., Braun, T., de Groot, B.L., Fotiadis, D., Kjellbom, P., Engel, A., 2005. The 5 Å structure of heterologously expressed plant aquaporin SoPIP2;1. J. Mol. Biol. 350, 611–616.

Mitsuoka, K., Hirai, T., Murata, K., Miyazawa, A., Kidera, A., Kimura, Y., Fujiyoshi, Y., 1999. The structure of bacteriorhodopsin at 3.0 Å resolution based on electron crystallography: implication of the charge distribution. J. Mol. Biol. 286, 861–882.

Miyazawa, A., Fujiyoshi, Y., Unwin, N., 2003. Structure and gating mechanism of the acetylcholine receptor pore. Nature 424, 949–955.

Murata, K., Mitsuoka, K., Hirai, T., Walz, T., Agre, P., Heymann, J.B., Engel, A., Fujiyoshi, Y., 2000. Structural determinants of water permeation through aquaporin-1. Nature 407, 599–605.

Nogales, E., Wolf, S.G., Downing, K.H., 1998. Structure of the alpha beta tubulin dimer by electron crystallography. Nature 391, 199–203.

Philippsen, 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.

Ren, G., Reddy, V.S., Cheng, A., Melnyk, P., Mitra, A.K., 2001. Visualization of a water-selective pore by electron crystallography in vitreous ice. Proc. Natl. Acad. Sci. USA 98, 1398–1403.

Renault, L., Chou, H.-T., Chiu, P.-L., Hill, R., Zeng, X., Gipson, B., Zhang, Z.-Y., Cheng, A., Unger, V., Stahlberg, H. 2006. Milestones in electron crystallography. J. Comp.-Aided Molec. Design (2006) in press.

Roosild, T.P., Greenwald, J., Vega, M., Castronovo, S., Riek, R., Choe, S., 2005. NMR structure of Mistic, a membrane-integrating protein for membrane protein expression. Science 307, 1317–1321.

Schenk, A.D., Werten, P.J., Scheuring, S., de Groot, B.L., Müller, S.A., Stahlberg, H., Philippsen, A., Engel, A., 2005. The 4.5 Å structure of human AQP2. J. Mol. Biol. 350, 278–289.

Schmid, M.F., Dargahi, R., Tam, M.W., 1993. SPECTRA: a system for processing electron images of crystals. Ultramicroscopy 48, 251–264.

Tate, C.G., Ubarretxena-Belandia, I., Baldwin, J.M., 2003. Conformational changes in the multidrug transporter EmrE associated with substrate binding. J. Mol. Biol. 332, 229–242.

Unwin, P.N., Henderson, R., 1975. Molecular structure determination by electron microscopy of unstained crystalline specimens. J. Mol. Biol. 94, 425–440.

Valpuesta, J.M., Carrascosa, J.L., Henderson, R., 1994. Analysis of electron microscope images and electron diffraction patterns of thin crystals of phi 29 connectors in ice. J. Mol. Biol. 240, 281–287.

Vinothkumar, K.R., Smits, S.H., Kühlbrandt, W., 2005. pH-induced structural change in a sodium/proton antiporter from *Methanococcus jannaschii*. EMBO J. 24, 2720–2729.

Wüthrich, K., 1998. The second decade—into the third millenium. Nat. Struct. Biol. 5 (Suppl.), 492–495.

Zeng, X., Gipson, B., Zhang, Z.Y., and Stahlberg, H. 2006. Automatic lattice determination for two-dimensional crystal images. Manuscript in preparation.