Protomo User's Guide

version 3.1

Hanspeter Winkler

Revision 1 – 2022-10-31

Contents

1 Introduction **4**

2 Installation **4**
2.1 Requirements . . . 4
2.2 Linux installation . . . 5

3 Concepts and conventions **6**
3.1 Tilt Geometry . . . 6
3.1.1 Coordinate systems . . . 6
3.1.2 Geometric transformations . . . 6
3.1.3 Dual-axis geometry . . . 7
3.2 Tomograms and 3D data sets . . . 8
3.3 Input/output . . . 8
3.3.1 Image file formats . . . 8
3.3.2 Tilt series metadata . . . 8
3.4 Fourier transforms . . . 9
3.5 Specifying parameters . . . 9
3.5.1 Units . . . 9
3.5.2 Command line parameters . . . 9
3.5.3 Parameter files for tilt series alignment . . . 10
3.5.4 Parameter files for subvolume processing . . . 10
3.5.5 Geometry files . . . 11

4 The alignment process **12**
4.1 Initial geometry . . . 12
4.2 Data collection strategies . . . 13
4.3 Common processing parameters . . . 13
4.3.1 General parameters . . . 13
4.3.2 Preprocessing . . . 14
4.3.3 Region of interest . . . 15
4.3.4 Reference construction . . . 16
4.3.5 Fourier space filters . . . 17
4.3.6 Cross-correlation and peak search . . . 18
4.4 Alignment . . . 18
4.5 Geometry refinement . . . 21
4.6 3D reconstruction . . . 21

5 Processing a tilt series **22**
5.1 Protomo commands . . . 22
5.2 Batch-mode processing scripts . . . 24

6 Subvolume processing **26**
6.1 The subvolume processing procedure . . . 26
6.2 Preparing a data set for processing . . . 27
6.3 Subvolume alignment . . . 29
6.4 Subvolume classification . . . 31
6.5 Processing scripts . . . 32

7 Tools **34**
7.1 Image display 34
7.2 Tilt series alignment 35
7.3 Manipulating image stacks 36
7.4 Printing image file metadata 36

8 References **38**

1 Introduction

protomo is a software package used in electron tomography for marker-free alignment of tilt series, 3D reconstruction, averaging, and classification of subvolumes extracted from tomograms. The marker-free alignment is based on cross-correlation methods and projection matching. It also includes the refinement of geometric parameters of the tilt series. 3D reconstruction is carried out by weighted back-projection with general weighting functions that allow varying tilt angle increments (Winkler and Taylor, 2006). Multivariate statistical methods are applied to subvolumes extracted from the tomograms Winkler and Taylor (1999) to characterize the variability of structural motifs found in the tomograms. The software was originally developed for thin sections of insect flight muscle and paracrystalline protein arrays (Taylor et al., 1997), but has since been successfully applied to various other specimens in cryo-electron tomography Winkler (2007). The newest version of the software also supports the alignment of dual-axis tilt series Winkler and Taylor (2013).

2 Installation

2.1 Requirements

Recent versions of the following software packages should be installed. These packages are not included in the *protomo* distribution.

- TIFF library (www.libtiff.org) (mandatory)
- Fourier transform libraries, optional but recommended. If no additional libraries are installed, the routines from FFTPACK (www.netlib.org/fftpack) are used.
  - FFTW2 (www.fftw.org); only version 2 is supported, which must be compiled for single precision with type prefix (configure options: –enable-shared –enable-float –enable-type-prefix).
  - djbfft (cr.yp.to/djbfft.html)
- MINPACK, a library for solving nonlinear equations and nonlinear least squares problems (www.netlib.org/minpack).
- LAPACK, Linear Algebra Package (www.netlib.org/lapack); the Lapack and Blas libraries must be compiled with a default integer size of 4 bytes (32-bit platforms) or 8 bytes (64-bit platforms), respectively. With the GNU Fortran compiler, use the option -fdefault-integer-8 on 64-bit platforms.
- GTK+, the GIMP toolkit (www.gtk.org)
- GtkGLExt, OpenGL Extension to GTK+ (gtkglext.sourceforge.net)
- GNU plotutils (www.gnu.org/software/plotutils)
- A postscript viewer
  - GNU "gv" (www.gnu.org/software/gv)
  - ImageMagick's "display" (www.imagemagick.org)

Note: If some of the above packages or libraries are not present at runtime (Fourier transforms, display related libraries, etc.), the corresponding functionality will be disabled. Tilt series alignment will not be affected.

2.2 Linux installation

Currently, *protomo* is available for Linux on 32-bit and 64-bit architectures (Intel i686, AMD64, and ARMV8). The software can be installed anywhere in the file system, including a user's home directory. The programs do not need superuser privileges to run, however "root" may be required to install the software in directories that are typically writable for root only, such as `/usr/local`. The software is unpacked by typing the following commands at the shell prompt:

```
cd /usr/local
tar -xjf /path/to/protomo-3.x.y.tar.bz2
```

where `/path/to/` is the absolute path where the downloaded tar file was stored, and `x.y` is the release number. The command in this example will create a new subdirectory `protomo-3.x.y` in `/usr/local`. The shell script `setup.sh` must be adapted to match the installation location: the environment variable `I3ROOT` must point to the top installation directory, in our case `/usr/local/protomo-3.x.y`, and the variable `I3DEPLIB` must point to the location of the third-party libraries (Fourier transforms, etc.), that are distributed or installed separately. If these libraries are already provided by the Linux distribution, it is recommended to make a symbolic link in the `I3DEPLIB` directory that points to the actual system library, as the file names or version numbers may be different from what the *protomo* software was linked against.

Other variables in the setup script should not be changed. The script must be "sourced" before the first *protomo* session or program invocation in the current shell. Alternatively, the script can be executed at login time by putting it in the shell startup files `.profile, .bash_profile`, `.bashrc`, or similar. To make sure that the correct versions of executables and libraries are loaded, the environment variables `PATH` and `LD_LIBRARY_PATH` should be examined. Library dependencies that will be loaded at runtime can be printed with the Linux system utility "`ldd`". If the output shows lines containing "not found", the corresponding package is either not installed or not in the library search path. When *protomo* or a command line program exits with the error message "undefined or invalid library path", the environment variable `I3ROOT` is most likely pointing to an incorrect directory or is not set at all.

When `setup.sh` is sourced without any arguments, `PATH` and `LD_LIBRARY_PATH` are reset and the current values are lost. This is primarily useful for debugging purposes to avoid interference with settings for other software. If previous settings of these variables are to be preserved, the script should be called as follows:

```
. setup.sh ${PATH} ${LD_LIBRARY_PATH}
```

Note, that the script is invoked with the dot operator on the command line, which executes it in the current shell instead of a sub-shell. Modifications to the environment variables are thus preserved in the current shell, which would otherwise be lost when executed in a sub-shell.

3 Concepts and conventions

3.1 Tilt Geometry

3.1.1 Coordinate systems

We define a Cartesian coordinate system $S = \{ O_s, \boldsymbol{s_1}, \boldsymbol{s_2}, \boldsymbol{s_3} \}$ which is fixed with respect to the specimen, a coordinate system $M = \{ O_m, \boldsymbol{m_1}, \boldsymbol{m_2}, \boldsymbol{m_3} \}$ which is the microscope coordinate system, and finally, for each projection image, a coordinate system $P^i = \{ O_p^i, \boldsymbol{p_1^i}, \boldsymbol{p_2^i} \}$, where $i = 1 \ldots n$, and n is the number of images in the tilt series. P^i is defined by the pixel raster, and the origin O_p^i is usually located at the bottom left of the image. We assume that the two origins O_m and O_s are identical, and the two planes spanned by the vector pairs $\boldsymbol{m_1}$, $\boldsymbol{m_2}$ and $\boldsymbol{p_1^i}$, $\boldsymbol{p_2^i}$ (the projection plane) are parallel. $\boldsymbol{m_3}$is the projection direction. The projection of O_s onto the image plane is denoted as O^i. O_s is implicitly defined by choosing one of the projection images as a reference image, usually the projection of the untilted specimen.

The tilt azimuth angle ψ measured anti-clockwise from the $\boldsymbol{m_1}$ axis, indicates the azimuthal direction of the tilt axis $\boldsymbol{t}$. An optional elevation angle φ can also be specified (not shown in Figure 1). A non-zero elevation angle means that the tilt axis is not perpendicular to the electron beam direction. In the figure, it would rotate the axis $\boldsymbol{t}$ out of the red plane. The angles ϑ^i are the specimen tilt angles corresponding to the i^{th} image. The in-plane rotation angles α^i align the projected tilt axes $\boldsymbol{t^i}$ in each of the images to a common axis. The green plane corresponds to the grid supporting the specimen, which in its untilted state coincides with the red plane. The rotation R_0 from the green plane to the yellow plane is the specimen orientation relative to the grid (see Figure 1). It is specified by three Euler angles: ψ_0, a rotation about the $\boldsymbol{s_3}$-axis, followed by a rotation ϑ_0 about the new $\boldsymbol{s_1}$-axis, and finally a rotation φ_0 about the new $\boldsymbol{s_3}$-axis.

3.1.2 Geometric transformations

All geometric transformations are specified as transformations of coordinate axes. The total transformation for a particular image consists of two rotations, a rotation about the tilt axis, followed by a rotation related to the specimen orientation which is the same for all images in the tilt series:

$$R^i = R_0 \, R_t^i$$

R_0 is the rotation matrix associated with the transformation of plane B to plane A, and R_t^i with the transformation from plane C to plane B, respectively. R_t^i is the rotation matrix of a rotation about the tilt axis $\boldsymbol{t}$ with the angle ϑ^i, for the i^{th} projection. The matrix R^i is the transformation of the coordinate system M to S:

$$\boldsymbol{s_j} = \sum_{k=1}^{3} r_{jk}^i \, \boldsymbol{m_k} \; ; \quad j = 1 \ldots 3$$

In the software, on the command line or in parameter files, the matrix is specified by listing its elements r_{jk}^i in the order of increasing indices, starting with the first row, then the second row, etc.. Note, that the transformations also require an origin which can be defined anywhere in an image. By default it is the center of the image.

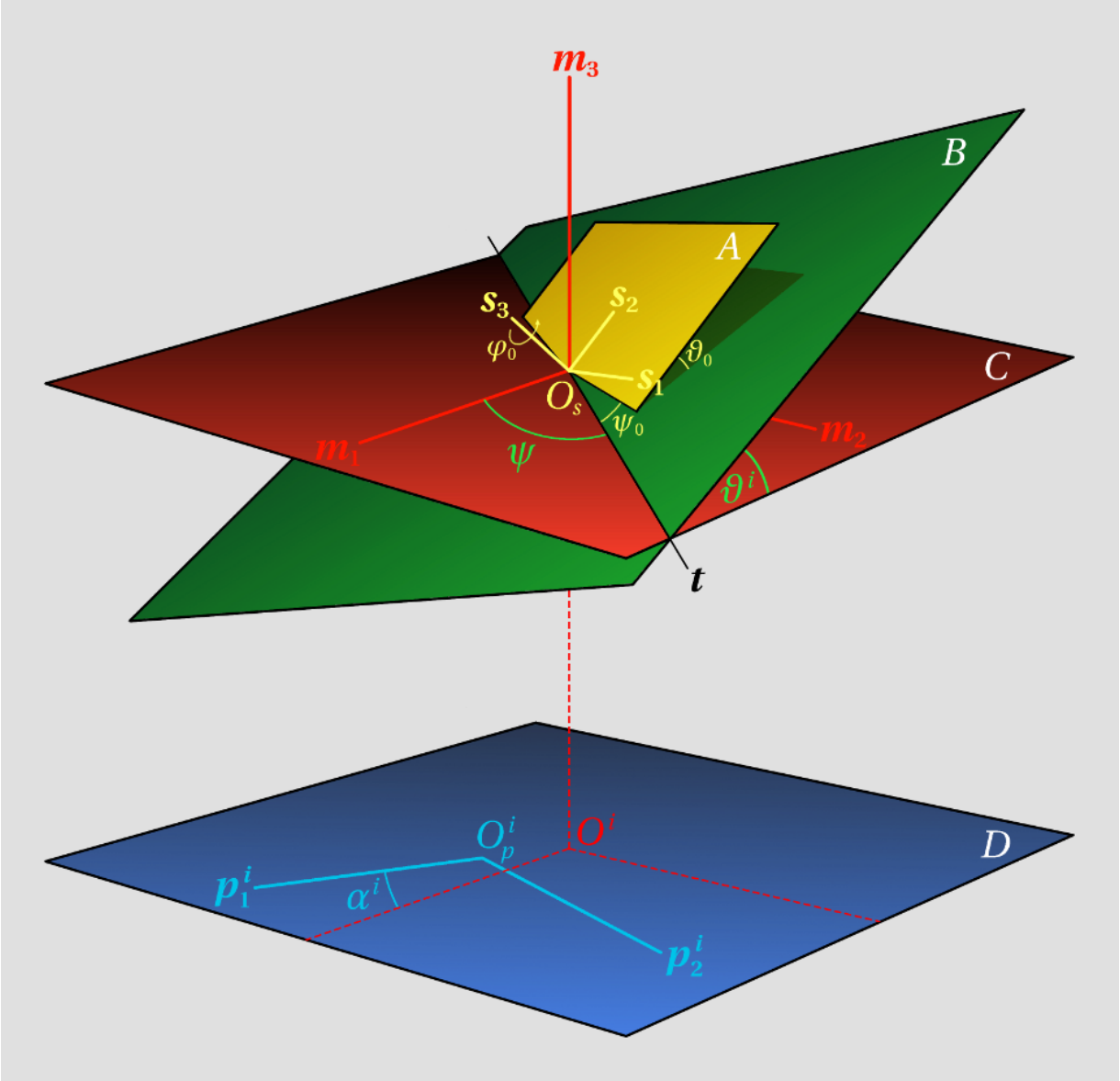

Figure 1: *Tilt geometry*

3.1.3 Dual-axis geometry

For dual-axis tilt series we define separate tilt azimuth angles ψ^k and elevation angles φ^k for each tilt axis $\boldsymbol{t}^k$. Also, we also assign a separate orientation R_0^k to the group of images that belong to tilt axis $\boldsymbol{t}^k$. In addition, for data collection schemes that involve the reversal of the tilt stage motion, subgroups can be formed for each tilt axis, for example a group for images with tilt angles 0...+60° and a second group for the images with tilt angles 0...-60°. This takes into account that the orientation could be slightly different for the two groups due to mechanical hysteresis effects.

3.2 Tomograms and 3D data sets

Similar conventions apply for units and geometry as described above. When manipulating tomograms and volumes extracted from tomograms, the associated geometric transformations are specified as transformations of the coordinate axes. We define the tomogram coordinate system as $S = \{ O_s, \boldsymbol{s_1}, \boldsymbol{s_2}, \boldsymbol{s_3} \}$ and the coordinate system that is related to the structural motif with index i in the tomogram with $V^i = \{ O_v^i, \boldsymbol{v_1^i}, \boldsymbol{v_2^i}\, \boldsymbol{v_3^i}, \}$. The motifs V^i may have arbitrary orientations within the tomogram. To average similar or identical motifs, linear transformations are associated with each motif which define a translation and rotation of S to V^i. Reinterpolating the data with these transformations produces subvolumes with the motifs oriented in the same way with respect to the voxel raster of the subvolume data. With all structural motifs oriented in the same way and aligned to a common origin we are then able to average the motifs.

The linear transformations of the motifs are specified with three origin coordinates relative to S followed by the matrix elements of the rotation matrix R_v^i. If a rotation is expressed with Euler angles, the z-x-z convention is used: the first rotation is about the z-axis, the second rotation about the new x-axis, and the third rotation about the new z-axis after the second rotation.

3.3 Input/output

3.3.1 Image file formats

protomo supports the following file formats: CCP4, EM, FFF, IMAGIC, MRC, SPIDER, SUPRIM, and TIFF. It can detect and read these formats transparently. The automatic format detection is based on the file contents rather than the file name suffixes. Map files are written in an internal format unless otherwise specified, and contain additional metadata required for further processing. Other files and diagnostic output files are written in the FFF format Winkler (2007). While the default output file format can be changed to any of the supported formats, it is not recommended to do so, because not all of the image formats can store the necessary metadata required by the software. In particular, a coordinate system must be associated with each image, and coordinates must be specified relative to this coordinate system in input data files and on the command line. The use of formats that implicitly assume coordinates 0,0 or 1,1 for the bottom left pixel in an image may result in unexpected behavior. In the recommended formats, the coordinate origin is recorded in the header, and cropping images, for instance, with *protomo* utilities preserves this information.

3.3.2 Tilt series metadata

The geometry information (see below, section "Geometry files") and alignment information is stored in a binary database for each tilt series. The geometry metadata files have a file name suffix of "`.i3t`" and are generated the first time a tilt series is created by importing a text file. All subsequent operations access the geometry metadata file only. Alignment and geometry re-evaluation modify their contents to keep track of the geometric parameters and the alignment status during processing. For diagnostic purposes the data can be re-exported in text form at any stage of processing.

3.4 Fourier transforms

Several Fourier transform algorithms are available:

- FFTPACK (www.netlib.org/fftpack), Fortran subroutines for complex and real sequences, developed by Paul Swarztrauber.
- GSL FFT, (www.gnu.org/software/gsl), part of the GNU scientific library. This is a reimplementation of FFTPACK. In *protomo*, it performs slightly worse than the original FFTPACK version and is therefore not recommended.
- FFTW (www.fftw.org), the supposedly "Fastest Fourier Transform in the West".
- *djbfft* (cr.yp.to/djbfft.html), an extremely fast FFT implementation that provides powers of two complex and real FFTs. In *protomo*, these routines run faster than FFTW.

Fourier transform modules are loaded during program startup. A fall-back sequence can be defined to select the optimal algorithm from the installed modules (the default sequence is *djbfft*, FFTW, FFTPACK). During invocation of a Fourier transforms routine, the sequence is searched for the requested routine, and if a particular transform type or transform size is missing or not available in the implementation of the algorithm, the next module in the defined sequence is selected. In the default sequence for instance, calls to a routine with a transform length that is a power of two would use *djbfft*, and calls with other transform lengths would skip *djbfft* and use the FFTW algorithm.

3.5 Specifying parameters

3.5.1 Units

The basic length unit for real space images is the pixel. All parameters pertaining to real space images are specified with pixels as the unit. In Fourier space, a different convention is used. There, a pixel does not correspond directly to a fixed spatial frequency because the spatial frequency assigned to a Fourier space pixel by the Discrete Fourier Transform algorithm depends on the number of samples in the real space image. To make the parameter specification for Fourier space images independent of the corresponding real space image size, the Fourier space unit is defined as one reciprocal real space pixel for parameters pertaining to Fourier space images. With this definition, the coordinates in Fourier space images lie always in the range from -0.5 to +0.5. Note that, if in the following "pixels" is specified as a unit, it always refers to real space pixels.

3.5.2 Command line parameters

Programs that accept command line parameters are invoked at the shell prompt like other Unix commands by specifying the name of the executable first, followed by options and then file name(s). Options are prefixed by a dash ("`-`") and can have zero or more parameters, which are all separated by spaces. To indicate the end of the option part on the command line, an optional double dash ("`--`") is used before the first file name. This avoids problems with parsing the command line if a file name starts with a number.

3.5.3 Parameter files for tilt series alignment

Parameter files for tilt series alignment are simple text files in a free format. Keywords and numbers are separated by "white space", i. e. any number of consecutive, non-printing characters such as spaces, tabs, new line, etc. Parameters for a particular application are specified in sections which consist of a section name followed by the section body which is enclosed in braces. For instance,

```
tiltseries { sampling: 2 }
```

is a section that declares a parameter "`sampling`". A section can contain nested subsections, and parameters with the same name declared within different subsections represent distinct values. Parameters are grouped in subsections according to various processing functions. The *protomo* software uses parameters defined in the section named "`tiltseries`" only and ignores all other sections. When read, the entire parameter file is parsed though, and all section and parameter specifications must therefore be syntactically correct. Comments can be included as follows and they can be nested and span multiple lines:

```
(* this is a comment *)
```

Parameters are declared with a parameter name, followed by a colon, followed by the parameter value. The parameter values can be of type boolean, integer, floating point and string. Boolean values are designated by the keywords `true` or `false`, strings are enclosed in double quotes. To define a parameter that consists of two or more values, the comma separated items are enclosed in braces. Variables are defined with the text sequence "variable name", "equal sign", "value", and simple arithmetic operations can be performed: addition (+), subtraction (-), multiplication (*), and division (/).

In the following example, a window is defined and reduced in size according to the specified sampling factor, so that it covers the same area in the original image when the sampling factor is changed:

```
tiltseries {
  N = { 512, 512 } (* window size *)
  S = 2            (* sampling factor *)
  suffix: ".tiff"
  sampling: S
  binning: true
  window { size: N / S }
}
```

If parameters need to be specified elsewhere on a command line, the corresponding section names in the parameter file are concatenated and separated by dots. The top level name (`tiltseries`) can be omitted in *protomo* applications. The name of the size parameter in the above example would therefore be specified as "`window.size`".

3.5.4 Parameter files for subvolume processing

Parameter files for subvolume processing are shell scripts that are sourced during execution. Parameters are environment variables that are exported in the shell. The file format must therefore conform to the shell syntax. In particular, for variable definitions, no blanks should be inserted between the variable name, the equal sign, and the assigned value.

3.5.5 Geometry files

Like parameter files, the geometry files are text files using a free format, i. e. keywords, identifiers and numbers can be separated by any amount of white space. All definitions are enclosed by keywords as follows:

```
TILT SERIES identifier
  definitions
END
```

where *identifier* is an alphanumeric text string (without quotes), which defines a name for the tilt series. *definitions* is a list of subsections. Each subsection can be either a tilt axis definition, a specimen orientation definition, a reference definition, or an image parameter definition. The following notation applies in the definitions below: keywords are specified in upper case, words in italic are substituted by the pertaining parameter, and text in angle brackets is optional. Parameters that are not explicitly specified are assumed to be zero.

```
AXIS
  TILT AZIMUTH angle
  < TILT ELEVATION angle >

ORIENTATION
  < PSI angle >
  < THETA angle >
  < PHI angle >

REFERENCE IMAGE number
```

Angles are specified in degrees. *number* is a non-negative integer number. The `AXIS` subsection defines the two angles ψ and φ, and the `ORIENTATION` subsection the rotation R_0. The `REFERENCE IMAGE` specification defines the image used as a coordinate reference. If omitted, the image with a tilt angle closest to 0° is automatically selected. These subsections are followed by multiple `IMAGE` subsections defined below.

An image parameter definition starts with the keyword `IMAGE` followed by an image number and a parameter definition:

```
IMAGE number
```

The valid parameter definitions are listed below:

```
FILE name     or    FILE name[index]
ORIGIN [ x y ]
TILT ANGLE theta
ROTATION alpha
SCALE factor
```

Image file names are constructed as explained in the next section by prepending a file path and appending a suffix to the file identifier *name*. The first character of a file identifier must be a letter and cannot be a number. Slashes are permitted in a file identifier; the corresponding raw images must then be located in the appropriate subdirectory. This can be used as a workaround if the file identifiers start with a number. Image files are either separate 2D images or image stacks stored in a "pseudo-3D" image. The first variant of the file identifier definition without brackets

assumes a 2D image, the second one an image stack, for which *index* within the brackets refers to the section number in the image stack (the pseudo z-coordinate).

The common origin of the tilt series in the image coordinate systems is defined with the `ORIGIN` keyword. x and y are the coordinates of point O^i in the image, where the index i corresponds to *number* defined with the `IMAGE` keyword. The tilt angle *theta* and the in-plane rotation angle *alpha* are denoted as ϑ^i and α^i in Figure 1. *factor* is an isotropic scale factor that modifies the sampling of the image. If unspecified, it is assumed to be 1.

For dual-axis tilt series, the `AXIS` subsection is repeated and the images listed after the second axis definition form a group with their own parameters. Each axis group can be further subdivided by repeated `ORIENTATION` subsections. Orientation groups are evaluated separately in the geometry refinement. Axis groups are treated differently in the alignment according to certain alignment options.

4 The alignment process

4.1 Initial geometry

For each image, an image file name, coordinate origin, tilt angle, and optionally an in-plane rotation must be defined in the geometry file. The image file name may include a section number, if the input data is in the form of a pseudo-3D image. The origin is specified relative to the pixel raster of the raw image and takes into account the origin recorded in the image file header. Tilt angles must be obtained from the data collection software. In *protomo*, the tilt angles are kept constant and it is assumed that the increments between the angles are accurate, but not the absolute values. A possible offset to the true angles is accounted for by the specimen orientation angles (cf. Fig. 1).

The initial geometry file can be created with a text editor, or more conveniently with the shell script `tomomaketlt.sh` that takes a simple text file as input. This shell script can be used as a reference implementation for converting data from the output of vendor-specific data collection software to the format of the geometry file in *protomo*. The text file contains a list of items, one line per image, in the following format:

Column	Data type	Description
1	integer	Sequence number
2	string	File identifier with optional section index
3	real	Tilt angle ϑ^i
4,5	real	x- and y-coordinate of common origin O_p^i
6	real	in-plane rotation α^i (optional)

Table 1: *Data file with initial geometry*

Angles are specified in degrees, origin coordinates in pixels. The script is called with the following command line arguments:

```
tomomaketlt.sh datafile identifier azimuth [orientation] [number]
```

datafile is the name of the text file in the format described above, *identifier* is an alphanumeric character string, and *azimuth* is the tilt axis azimuth ψ^k. The argument *orientation,* the angle φ_0, is optional. The numbering of the images starts with zero by default and the value can be changed with the optional argument *number.* If the data collection software stores two images at 0° tilt, one of the images should be removed. This can be done by deleting it from *datafile*, or later with the `exclude` parameter in the parameter file (Table 2).

The sequence number indicates the order of data collection. For the first collected image, it should be set to a value of 1. Subsequently, the number should be increased by 1 every time the tilt stage movement has been reversed. Images with the same sequence number belong to the same image group and appear as such in the output file (see below under "data collection strategies").

For dual-axis tilt series, the rotation angle of the specimen between the two partial tilt series must also be supplied. This is accomplished with the orientation angle φ_0. Note, that the tilt azimuth is the same for both parts, because it is defined relative to the microscope coordinate system. The rotation angle can easily be measured by comparing the images at 0° tilt. For the subsequent alignment, the second image at 0° tilt should be excluded because it received a higher radiation dose and it would also double the contribution of that angle in the final map. An automated procedure to create the initial geometry file for dual-axis tilt series is described in the *protomo* tutorial.

4.2 Data collection strategies

When data are collected in a single pass from negative to positive angles (or vice versa) then a single rotation specified with three Euler angles is sufficient to account for the specimen orientation. However, if data are collected in two passes, from 0° to the most negative tilt angle and subsequently from 0° to the most positive angle (or vice versa), then mechanical backlash may occur when the tilt stage reverses its movement between the two passes. As a consequence, images collected at nominally identical tilt angles may have slightly different true tilt angles, and this difference is compensated by creating two sets of parameters for the orientation angles. For other collection schemes a new set of orientation angles should be defined every time the tilt stage movement is reversed.

4.3 Common processing parameters

4.3.1 General parameters

The parameters described in this section control the input and output of image data, diagnostic terminal output, and the sampling of the input images (Table 2). They are specified within a top level section called `tiltseries`. When a tilt series is processed, the raw images are first located by file name. File names are constructed from the identifiers specified in the geometry file by appending a suffix defined in the parameter file. If a path list was also defined, the files are searched in the directories given in that list. The images are first preprocessed if preprocessing is enabled by the boolean flag `preprocessing`. The preprocessed images are stored in a cache file. If the parameter `cachedir` is defined, the cache file will be created in that directory, otherwise the current working directory is used. If binning is requested and the sampling factor is greater than or equal to two, the raw or preprocessed images are binned and stored in an additional cache

file. The binning factor is the sampling factor truncated to an integer value. All cache files can be deleted anytime and will be automatically regenerated when needed.

Other output files are created in the directory defined by the parameter `outdir` or the current working directory if the parameter is undefined. The selection and exclusion parameters have the effect of ignoring images in the tilt series without having to remove them explicitly. The format of a selection specification is a comma-separated list of image numbers or image number ranges. The parameter is a character string and must therefore be enclosed in double quotes. By default, all images are being selected. If both selection and exclusion parameters are present, selection takes place before exclusion.

4.3.2 Preprocessing

The preprocessing is carried out in one or two passes. In each pass, image statistics are first computed from the whole image or a smaller region thereof. A region is defined by specifying a border of fixed width. From the selected region, a linear density gradient can optionally be subtracted and/or the densities can be thresholded below and above a specified multiple of the standard deviation, or alternatively, below and above absolute density values. Additionally, a median or Gaussian filter can be applied. The preprocessing parameters are specified in a section called `preprocess` (Table 3). If a subsection `mask` is present within the `preprocess` section, then the preprocessing is carried out as a two-pass operation. The preprocessing parameters in the inner `mask` subsection apply to the first pass, the parameters in the outer `preprocess` section to the second pass. The first pass generates a binary mask that indicates which pixels are to be set to a locally computed mean value in the second pass. If a parameter `grow` is supplied, contiguous areas of selected pixels in the binary image are enlarged at the perimeter by the

Parameter name	Parameter type	Parameter description
prefix	string	Prefix for input and output files, with the exception of raw image files, which are specified in the geometry file.
suffix	string	Suffix for raw image files.
pathlist	string	Colon-separated list of directories to search for raw image files.
cachedir	string	Directory where cache files are stored.
outdir	string	Directory where other output files are stored.
preprocessing	bool	Enable/disable preprocessing of raw image files.
binning	bool	Enable/disable binning of raw image files.
sampling	real ≥ 1	Sampling factor.
select	selection	Select specific images in the tilt series.
exclude	selection	Exclude images from the tilt series.
logging	bool	Enable diagnostic terminal output.

Table 2: *General parameters*

number of pixels specified. The binary mask is applied in the second pass in addition to the other requested operations. Two-pass preprocessing is primarily used to remove density outliers. Preprocessing can be turned off without removing the `preprocess` section from the parameter file (see parameter `preprocessing` under general parameters).

4.3.3 Region of interest

Three sections are relevant for the tilt series alignment: `window`, `reference`, and `align`. The first step in the alignment procedure is the extraction of windows from the source images according to the parameters specified in the `window` section (Table 4). The source images are raw, preprocessed,

Parameter name	Parameter type	Parameter description
border	int > 0	Width of border to exclude from image statistics calculation.
clip	real, real	Lower and upper threshold, specified in multiples of the standard deviation.
thr	real, real	Lower and upper threshold, specified as density values.
gradient	bool	Enable linear gradient subtraction.
iter	bool	Iterate gradient subtraction once.
filter	string	"median" or "gauss"
kernel	int, int	Filter window size.
radius	real, real	Widths of the Gaussian function.
grow	int > 0	Grow the selected regions in the binary mask by the specified number of pixels.
logging	bool	Enable diagnostic terminal output.

Table 3: *Preprocessing parameters*

Parameter name	Parameter type	Parameter description
size	int, int > 0	Size of extracted and resampled window.
area	real	Fraction of extracted area that must lie within the source image.
mask	section	See table 5
lowpass	section	See table 5
highpass	section	See table 5
gaussian	section	See table 5

Table 4: *Window parameters*

or binned images, depending on the preprocessing and resampling parameters. The resampling is carried out with linear interpolation and uses the geometric parameters stored in the geometry metadata file. The size of the extracted window after resampling is the size specified with the `size` parameter in the `window` section. Choosing the window size must take into account the selected sampling factor. If the sampling factor is greater than one, then the corresponding area in the raw image is larger than the window area. Consequently, the window size should be reduced when the sampling factor is increased (or vice versa), so that the resampled windows cover equivalent areas relative to the raw image.

If binning is selected, the window is extracted from the cached, binned data, otherwise it is extracted from the preprocessed or raw data. In the former case, the sampling factor used in the linear interpolation is automatically adjusted to take into account the binning factor, so that the overall sampling factor remains as specified by the `sampling` parameter. If the resampled area does not lie completely within the source image, an error is generated and the alignment is terminated. This condition occurs when the region of interest lies close to the edge of a raw image, or when the region shifts by large amounts due to poor tracking during data collection. The error condition can be relaxed by specifying the minimal fraction of the extracted area that must lie within the source image. The pertaining parameter is called `area`, with a default value of 0.95. Extracted areas not covered by the source image are filled with zeros.

4.3.4 Reference construction

The alignment reference construction is controlled by the sections `window` and `reference`. The size of the area that will be used for alignment is specified in the `window` section, whereas the parameters for the reference construction algorithm are specified in the `reference` section (Table 6). The selection and exclusion parameters within the `reference` section define which images in the tilt series contribute to the reference construction. The format of these parameters is the same as explained in the section "General options".

If no back-projection options are present in the `reference` section, the back-projection body parameter is zero, or the section is absent, the reference image is a single image which is selected from the already aligned images. The selection criterion is the smallest tilt angle difference between the selected image and the image being aligned. The selected image that will become the reference is extracted, resampled, masked, and Fourier transformed, as described in the previous sections. Parameters that apply to this step are specified in the `window` section, including the mask parameters, which are contained in a `mask` subsection (a different mask is applied to the extracted images that are cross-correlated with the reference image, see below). Rectangular and

Parameter name	Parameter type	Parameter description
width	real, real ≥ 0	Rectangular mask widths.
diameter	real, real ≥ 0	Ellipsoidal mask, principal axes.
sigma	real, real ≥ 0	Gaussian mask widths.
apodization	real, real ≥ 0	Apodization for rectangular and ellipsoidal masks.
inv	bool	Mask interior instead of exterior region.

Table 5: *Mask parameters*

ellipsoidal masks should generally be apodized to avoid ripple effects when Fourier transforms are involved.

The presence of a `reference` section with the parameter `body` indicates that the reference is to be calculated by re-projecting a preliminary back-projection map, based on already aligned images. The new reference is constructed by updating stored back-projection data which was computed and continuously updated during the previous cycles. Images that have been aligned in the current cycle but not yet been merged into the back-projection data are prepared for merging as described above for a single-image reference. Updating the back-projection data includes updating weighting functions and computing re-projections of a preliminary map. The re-projection is carried out with the geometric parameters of the image that will be aligned, and the re-projected image becomes the new reference image in the alignment.

4.3.5 Fourier space filters

The operation following the reference construction is the computation of a filtered cross-correlation function. For this purpose, high-pass and/or low-pass filters are applied to the reference image produced by the selected construction algorithm. Since the cross-correlation function is computed via multiplication of Fourier transforms, these multiplicative filters could be applied to either one of the operands, or to the result, which is the transform of the cross-correlation function. For efficiency reasons, the filters are applied to the reference transform in *protomo*, because the reference transform is computed only once, whereas the image transforms may have to be evaluated multiple times with varying parameters in the area matching algorithm.

Low-pass and high-pass filters are specified with the same identifiers as the real space masks, but only the `diameter` and `apodization` parameters make sense in the context of a Fourier space filter. The units for Fourier space filter limits are reciprocal pixels. Note, that the filter limit parameters are the lengths of the principal axes ("diameter") of an elliptical or circular region, not the spatial frequency of the cutoff, i. e. a value of 1 would filter at the Nyquist frequency, whereas the Nyquist frequency itself has a value of 0.5 in our definition. If such a transform that has been filtered at the Nyquist frequency, is displayed, the filter would appear as an ellipse or circle inscribed in a rectangular or square image of the transform.

Additionally, a Gaussian filter may be applied. The width of the Gaussian is specified with the parameter `sigma`. It functions similar to the `diameter` parameter for low- and high-pass filters. A value of 1 for `sigma` would produce a Gaussian weighting of the transform that has a weight of $1/e$ on a circle with diameter 1, i. e. the Nyquist frequency would be weighted down by that factor.

Parameter name	Parameter type	Parameter description
body	real	Back-projection body size.
slab	bool	Adjust back-projection body size for a slab-like specimen.
select	selection	Select images of the tilt series for reference computation.
exclude	selection	Exclude images from reference computation.

Table 6: *Reference and reconstruction parameters*

4.3.6 Cross-correlation and peak search

The image that is cross-correlated with the reference image can also be masked before its Fourier transform is computed. This mask is specified as a `mask` subsection (Table 5) within the `align` section. Several cross-correlation methods are available: the conventional cross-correlation "`xcf`", the mutual correlation "`mcf`", phase only correlation "`pcf`", and phase doubled correlation "`dbl`". For diagnostic purposes, an image stack with the cross-correlation functions can be written to a file. The `size` parameter for the diagnostic output specifies the image window that is extracted and written to the file. This parameter does not affect the cross-correlation calculation nor the peak search described below, which are always based on the image window size defined in the `window` section. The cross-correlation specific parameters are enclosed in a `correlation` subsection (Table 7).

In order to find the location of maximal correlation, the whole cross-correlation function is searched by default. An optional `radius` parameter restricts the search for the highest correlation peak to a smaller ellipsoidal or circular region. The two required values for this option define the principal axes of the search region. If a parameter `cmdiameter` is also present, a refined value (i. e. the center of mass) of the peak position is calculated in a second step. The calculation is performed within a region that is centered at the maximum that was obtained by the first search. The region for the center of mass calculation has the dimensions defined by the `cmdiameter` values. The peak search parameters are specified within a `peaksearch` subsection (Table 8).

4.4 Alignment

A coarse alignment can be carried out interactively with the graphical tool described in section "Standalone graphical tools". It is useful for the correction of single misaligned images under visual control, or for an initial alignment of a raw tilt series when large shifts occurred during data collection. The tool offers options for a manual or automatic alignment of individual images, or the alignment of an entire tilt series. The automatic alignment function performs a simple, sequential cross-correlation between neighboring image pairs. Differences in the foreshortening of the images due to the different tilt angles are accounted for in the cross-correlation computation. Area matching and back-projection references are not implemented in the graphical tool.

Parameter name	Parameter type	Parameter description
mode	string	Correlation mode: "`xcf`", "`mcf`", "`pcf`", or "`dbl`".
size	int, int > 0	Size of output image with correlation peaks.

Table 7: *Cross-correlation parameters*

Parameter name	Parameter type	Parameter description
radius	real, real > 0	Defines peak search region.
cmdiameter	real, real > 0	Size of region for center of mass calculation.

Table 8: *Peak search parameters*

The batch-mode alignment provides options for grid search alignment as well as for area matching. Both operate on the entire tilt series. Grid search and area matching start with the image defined as the coordinate reference in the geometry parameter file. This image is chosen as the first alignment reference in the iterative procedure, and the neighboring images at the next lower and higher tilt angle are aligned to this reference. The images are extracted and resampled according to the stored geometry parameters unless certain parameters are set (cf. Table 10). For instance, the `estimate` parameter takes the correction values of the already aligned neighbor image as an estimate for the current image. This is usually most effective in the first cycle, when the correction is still unknown (the stored values are initialized to zero). After extraction and resampling, a mask is applied to the images. The image mask is specified in the `align` section and can be different from the mask applied to the images involved in the reference construction.

The grid search alignment mode is selected when a subsection `gridsearch` is present in the `align` section (Table 9) and the `step` parameter of the grid search is non-zero. Otherwise, area matching is performed. After successful completion, the aligned images are marked for inclusion in the reference. For a simple sequential alignment no further steps are necessary to proceed to the next iteration. For a back-projection reference, the weighting functions must be updated at this point and re-projections of the updated preliminary map be computed that serve as the new references. The new references are then used to align the next two neighbors in the tilt series. The above process is repeated until all images are aligned, or one of the termination criteria listed in Table 10 is met. Termination parameters are useful to restrict the alignment to a subset of images, which reduces the computational effort especially in the initial cycles. Rather than attempting to align the whole tilt series based on an inaccurate geometry estimate, it is preferable to re-evaluate the geometry with a subset of aligned images at this point, and proceed to a new cycle.

Dual-axis or multiple-axis tilt series can be aligned in two ways, either separately or simultaneously. For the first variant, the images are grouped by tilt axis first and the groups are aligned and merged one after another. For the second variant, for which the parameter `startangle` must be set, all images are aligned and merged in the order of increasing tilt angle magnitude, alternating between the axis groups. If `startangle` is greater than zero, the first few images with a tilt angle magnitude less than the value of `startangle` are aligned according to the first variant. After the alignment of these images is complete, the alignment mode is switched to the second variant for the remaining images.

Area matching optimizes a four parameter correction per image, whereas the grid search determines the optimal in-plane rotation only. In both cases, translational alignment is always derived

Parameter name	Parameter type	Parameter description
step	real >= 0	Defines the step size of the angular increment for the rotational grid search. If this value is set to 0, then area matching instead of a grid search is performed and all grid search parameters are ignored.
limit	real >= 0	The grid points lie in the range from -limit to +limit. If limit is set to 0, then only a translational alignment is performed.

Table 9: *Grid search parameters*

from the location of the correlation peak. Area matching computes a correction C^i to the input transformation R^i with an optimization algorithm, and the resulting new transformation is:

$$A^i = R^i C^i$$

A^i is the total transformation applied to a particular image that produces the best alignment with respect to the generated reference. To interpret the correction matrix, we compute the singular value decomposition:

$$C^i = U S V^T$$

The singular values of the diagonal matrix S indicate a stretch/compression correction in two orthogonal directions with an orientation angle given by the orthogonal matrix V. The matrix U yields the correction of the in-plane rotation. Since a grid search only computes the in-plane correction, a geometry refinement is not possible, because only one of the four necessary correction parameters per image are known.

Parameter name	Parameter type	Parameter description
gridsearch	section	If present, use a grid search algorithm instead of area matching. See table 9
estimate	bool	Estimate geometric parameters instead of using stored values from previous cycle.
norotations	bool	Set in-plane rotations to zero instead of using stored values.
area	real	Fraction of area that must be covered by source image.
mask	section	Mask parameters for aligned image, see table 5
correlation	section	See table 7.
peaksearch	section	See table 8.
translimit	real	Discard alignment and keep original geometric parameters if translational shift exceeds specified value.
maxcorrection	real	Terminate alignment if correction exceeds specified value.
maxshift	real	Terminate alignment if translational shift exceeds specified value.
startangle	real	Start alternating dual-axis alignment at specified tilt angle.
maxtilt	real	Align images only up to $\|angle\| \leq maxtilt$.
include	selection	Include images in the alignment.
exclude	selection	Exclude images from alignment.
logging	bool	Enable diagnostic terminal output.

Table 10: *Alignment parameters*

4.5 Geometry refinement

The geometry refinement tries to reduce matrix C_f^i to a unit matrix in the equation $A^i = R_f^i C_f^i$ by varying the alignment transformations R_f^i. The matrices A^i were obtained by area matching, and are constant here. After refinement, R_f^i define the new transformations. The parameters included in the refinement can be selected individually. In the first few cycles it is usually best to select azimuth, rotation, and orientation only. In later cycles, the scale refinement and possibly the elevation refinement may be enabled. In dual-axis tilt series, including the scale factor in the refinement helps to compensate magnification differences between the two parts of the series. Note, that if images have been explicitly excluded with the selection/exclusion parameters in an alignment cycle, a subsequent re-evaluation of the geometry is based only on the newly aligned images. This could result in undesirable changes in the geometric parameters if the images exhibit poor alignment.

4.6 3D reconstruction

3D reconstruction is carried out by weighted back-projection with general weighting functions. The back-projection parameters have the same meaning as in the reference construction section (cf. Table 6). The size of the map is set in the `map` section, and is independent of the window size used during alignment. The sampling factor (cf. 2) is also taken into account. If it is greater than or equal to 2, and binning is also selected, the binned raw images are used for the reconstruction. A low-pass filter can be applied to the source images in the map calculation to filter out the signal beyond the resolution limit. The filter is applied before back-projection takes place, so back-projection artifacts may still be present in the computed tomogram at higher spatial frequencies. To remove the artifacts, a 3D low-pass filter should be applied to the computed tomogram.

Parameter name	Parameter type	Parameter description
orientation	bool	Include orientation angles in refinement.
azimuth	bool	Include tilt azimuth in refinement.
elevation	bool	Include tilt axis elevation in refinement.
rotation	bool	Include in-plane rotations in refinement.
scale	bool	Include scale factors (magnification) in refinement.
include	selection	Select images by image number.
exclude	selection	Exclude images.
logging	bool	Enable diagnostic terminal output.
loglevel	int > 0	Increase verbosity of diagnostic output.

Table 11: *Geometry refinement parameters*

Parameter name	Parameter type	Parameter description
size	int, int, int > 0	Size of computed map.
identifier		See table 6
lowpass	section	See table 5.
logging	bool	Enable diagnostic terminal output.

Table 12: *Map computation parameters*

5 Processing a tilt series

Tilt series alignment is carried out with the program *protomo*. Due to the large number of parameters required for the alignment, the parameters for *protomo* operations are not passed to the processing routines on the command line, but are rather imported from a text file. Likewise, geometry information and tilt series metadata are read from a text file initially and maintained in a binary geometry metadata file thereafter. The *protomo* program operates either interactively, or alternatively in batch-mode, when a command file name is specified on the shell command line.

5.1 Protomo commands

The program *protomo* is invoked from the shell command line as described below:

`protomo [-log] [-param` *`file`*`] [-tlt` *`file`*`] [`*`commandfile`*`]`

If the option `-log` is specified, the commands that are executed are logged to standard error. *`file`* is a parameter file name or tilt geometry file name, respectively. If *`commandfile`* is given, commands are read from that file instead of the terminal.

A tilt series must first be initialized with a geometry file as input which is either given on the shell command line or as a *protomo* command. When a new tilt series is opened, a binary metadata file with suffix "`.i3t`" is created that contains all information including the geometry that is read from the geometry text file. At this point, the raw images are preprocessed and stored in a cache file.

After initialization, a typical alignment session first reads a parameter file, specified either on the shell command line or as a *protomo* command. All geometric parameters are now read from the binary metadata file (the geometry text file is no longer needed). Results of the alignment are stored in the metadata file. If an alignment command is interrupted, it can be restarted and will resume with the first unaligned image pair. After the alignment is completed, the geometry is usually re-evaluated ("fitted"). This can only be done when area matching is used and not for a grid search alignment, since a grid search does not provide the required correction information. It is not necessary to align all images of the tilt series to compute updated geometry parameters. This can save execution time during the first few cycles, if the estimate for the tilt axis azimuth is not very accurate in the beginning. The re-evaluation can be carried out multiple times with different parameters. The updated parameters are stored in a temporary location and must be explicitly saved, otherwise they will be lost if the session terminates at this point.

Maps can be computed at any stage of alignment. If the map is to be used later for subvolume processing, it is recommended to write it in the default file format which will save additional metadata such as sampling and tilt geometry information, for instance. If this information is missing, certain features of the subvolume processing will not be available.

A list of all *protomo* commands is shown below. Optional arguments appear in brackets. If the file path is not specified for a command, the file name is generated automatically from the file identification and cycle number.

Command	Description
align	Align the tilt series with area matching or optimize translations and in-plane rotations with a grid search.
area [to *path*]	Determine the maximal usable area and write a map to a file with name *path*.
close	Close the current tilt series.
corr [to *path*]	Write correction data to a text file with name *path*.
cycle [*N*]	Set the cycle number to the number *N* of a previous alignment and restore the geometric parameters from that cycle. Note, that this discards all results from cycles greater than *N*. If *N* is omitted, print the current cycle number.
euler [*E1 E2 E3*]	Set orientation angles to the real values *E1*, *E2*, and *E3*, or print the angles if no values are specified.
filter *N* [to *path*]	Compute a filtered version of the aligned area of image number *N* and write it to a file.
fit	Re-evaluate geometry parameters.
geometry [aligned|fitted] [to *path*]	Write geometry to a text file.
help	Print a short descriptions of all commands.
image *N* [to *path*]	Compute the aligned area by resampling image number *N* and write it to a file.
map [to *path*] [format *FMT*]	Compute a tomogram and write it to a file. Optionally use file format *FMT* (not recommended).
open [*prefix*]	Open a tilt series.
origin [*X Y Z*]	Set the map origin, or print the origin if no values are specified. Setting the map origin shifts the region of interest without changing the origins of the individual images.
param *PAR* [*VAL*]	Set the parameter *PAR* to value *VAL*, or print the parameter value if *VAL* is not specified.
paramfile *path*	Read a parameter file.

plot	Display a graph of the stretch/compression factors.
preproc *N* [to *path*]	Preprocess image number *N* and write it to a file.
print [param | status]	Print tilt series metadata.
quit	Quit the program.
refine	Refine alignments.
restart	Realign already aligned images instead of skipping them.
tiltfile *path*	Read geometry from a text file.
transform *N* [to *path*]	Compute the transform of the aligned area of image number *N*.
unalign	Discard the alignment of the current cycle.
update	Replace the current geometry with the re-evaluated geometry and start a new cycle.
usefitted	Replace the current geometry with the re-evaluated geometry without starting a new cycle.

Table 13: *Protomo commands*

5.2 Batch-mode processing scripts

The scripts described below simplify the alignment process by running the program *protomo* with the appropriate commands. They keep track of the alignment status and save intermediate results in a subdirectory of the current working directory. Each tilt series has its own subdirectory and must be initialized first to create the subdirectory and initial data files.

Typically, a good alignment strategy is to start with binned images, reducing the binning factor gradually. First, a coarse alignment is carried out that aligns the images translationally and computes more accurate geometric parameters, in particular for the tilt axis azimuth which usually tends to deviate from the true value in the initial estimate. The coarse alignment is run with a large binning factor to speed up the process. Subsequently, the alignment is refined at the same binning factor with area matching. For the following area matching cycles, the binning factor can then be reduced gradually.

The processing scripts do not produce any terminal output (or output to standard output or standard error). Diagnostic messages and error messages are written to a log file that is initially created in the current working directory. If a script terminates abnormally, the log file is left in place where it can be examined. On successful completion, it is moved to a subdirectory of the particular tilt series.

Alignment initialization

```
tomoinitialize.sh ident param tltdir
```

ident is an alphanumeric string from which directory and file names are derived. *param* is a parameter file, and *tltdir* a directory where the initial tilt geometry files are located. The

name of the geometry file must consist of the string *ident* with the suffix "`_initial.tlt`" appended. All parameters for the first coarse alignment run should be set in the parameter file *param*, especially an estimate for the specimen thickness, and the size of a preliminary map. The script creates a new subdirectory with the name *ident* and copies the relevant files into it, then preprocesses the images and stores the preprocessed data in a cache file. It also saves log files and other data in a subdirectory called "`history`".

Coarse alignment

`tomoaligninitial.sh` *ident trans match*

ident is the identification string of the tilt series, *trans* and *match* the number of iterations of translational alignment and area matching, respectively. The parameters that are used are those specified during initialization. The script first aligns each image to its nearest neighbors. After *trans* iterations, it switches the alignment mode to area matching and runs *match* iterations of area matching, followed by a geometry refinement. At the end, it computes a preliminary map. Results are placed in the subdirectory specified in the parameter file and in the subdirectory "`history`".

Refinement

`tomoalign.sh` *ident param match* `[update] [map]`

ident is the identification string of the tilt series, *param* the name of a parameter file, and *match* the number of iterations of area matching. If the keyword `update` is given, the window size is set to the maximal size estimated during the previous cycle. The keyword `map` enables the computation of a map after the alignment iterations are completed. Results are placed in the same locations as for the coarse alignment.

Map computation

`tomomap.sh` *ident param* `[body` *b*`] [size` *x y z*`] [orient` *e1 e2 e3*`]`

ident is the identification string of the tilt series, *param* the name of a parameter file. The optional parameters `body` and `size` override the values in the parameter file. If `orient` is specified, the Euler angles *e1*, *e2*, *e3* replace the orientation angles of the stored tilt geometry. Specifically, for values of 0 0 0, the projection direction for the nominally untilted specimen is set to the map z-axis, and the specimen may therefore appear slightly tilted in the computed map if it was not lying flat on the grid.

6 Subvolume processing

Tomograms produced by the *protomo* software contain all the necessary metadata required for subvolume processing, including the tilt geometry, when written in the default file format. The tilt geometry is only needed if missing wedge compensation is applied to determine the direction of the tilt axis in the tomogram and the subvolumes. Two options of generating the initial datasets from the tomograms are supported. The first one is to extract the subvolumes from the tomogram at the given positions and write them to a new file as a stack of subvolumes. The second one is to link the positions to the tomogram without copying the data. The former option is useful if the subvolume positions in the tomogram are separated by large distances and the extraction produces a smaller stack file. For further processing, only the stack files need to be present. The latter option avoids copying and duplicating the image data if the data reduction is not significant. In this case, both the stack file and the raw tomograms are needed because the subvolumes must be extracted from the tomograms on the fly during processing.

Normally, a separate stack is generated from each tomogram, and for subsequent processing, all stacks are combined into a single dataset. The combination of individual stacks into a single set creates a new file with references to the original stacks, and no image data will be copied to the new file at this point, to avoid duplication of the data. Therefore, for further processing, the original stacks, or tomograms, if the second option above was chosen, must still be present in addition to the newly created file.

The stacks whether produced by extracting or linking the data should only be manipulated with the programs described below in order not to lose the metadata required for processing. Each stack contains the original positions where the volumes were extracted from the tomograms, additional shifts if an alignment was carried out with the data, and rotation matrices. The rotation matrices are either computed by the alignment procedure or can be supplied initially when a new stack is created. Other metadata that may be stored in the stack file by a particular operation includes tilt geometry, cross-correlation coefficients, and class memberships, to name a few.

6.1 The subvolume processing procedure

The general procedure is shown schematically in Figure 2. It consists of alternating alignment and classification operations. To speed up the whole process, the procedure is started with small subvolumes at low resolution. Depending on the source data that is available, the small subvolumes are extracted either from maps computed at lower resolution than the raw tilt data (sampling factor > 1), or from maps that were binned. If subvolume stacks instead of maps are available, these can be used directly or after a binning operation.

In the initial cycle, the references for the alignment are usually derived from a global average. Cycles following the initial cycle typically use class averages. If the reference file provided as input is an image stack, or if multiple images are provided, a multireference alignment (MRA) is carried out, in which each subvolume is cross-correlated with all references, and the alignment with the highest cross-correlation coefficient is selected. The alignment is followed by multivariate statistical analysis (MSA), classification, and the generation of class averages and new references. A more detailed discussion of subvolume processing can be found in Taylor et al. (2006).

The binning factor can easily be changed for the entire dataset. If the dataset was derived from unbinned data, all the original source files must be accessible, and the subvolumes are

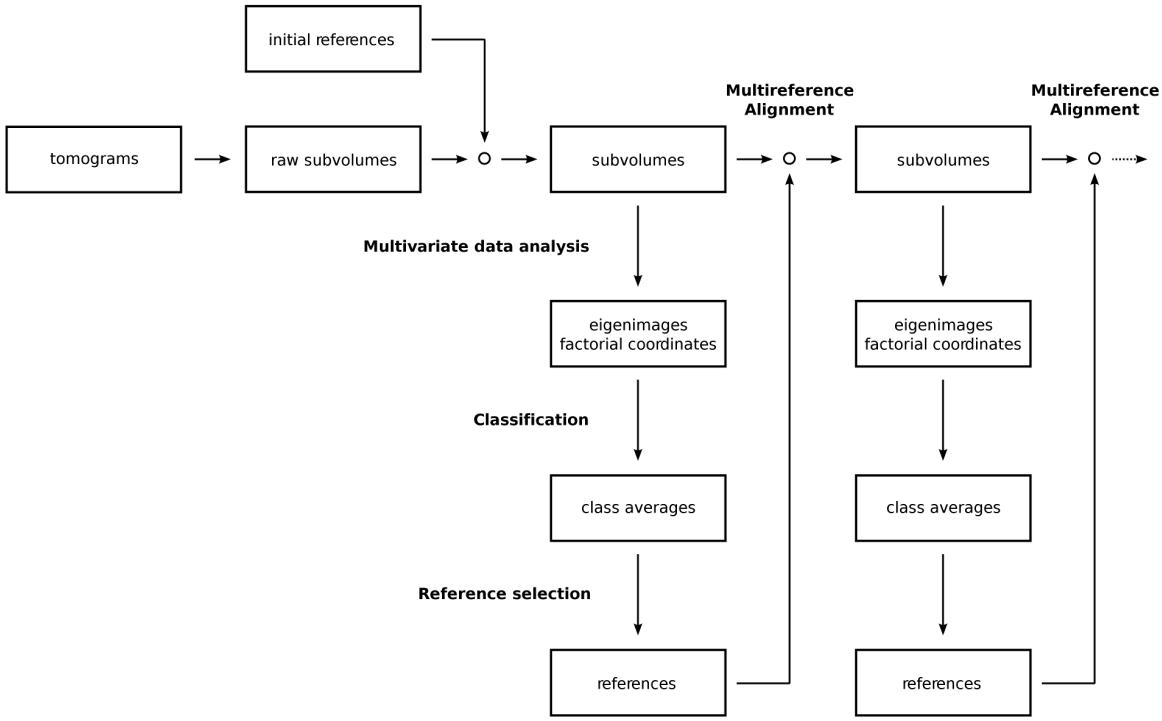

Figure 2: *Subvolume processing procedure*

re-extracted from the appropriate files. Center positions and alignment data are automatically adjusted according to the new binning factor. The resulting stack is then used as the initial stack for a new round of processing.

6.2 Preparing a data set for processing

A subvolume stack is generated from a tomogram and a list of particle positions with the program *tomoprepare*. The subvolumes can either be copied to the new stack file ("extracted"), or the positions can be recorded in the stack file with a reference to the original map ("linked") without actually copying the data. The former option generates an independent stack, whereas the latter option requires both the stack and original map to be present during processing, but may save disk space by not duplicating the data. For faster processing and reduced disk space usage, the tomograms may be binned before creating the stacks.

The lists of particle positions are plain text files. Each line contains the data for one particle: space-separated coordinates, optionally followed by space-separated matrix elements that define the particle coordinate system, usually in the form of a rotation matrix. Lists for multiple tomograms must be stored in separate files. By default, the rotations are ignored when the subvolumes are extracted and not recorded unless explicitly enabled in the extraction command. If recording is enabled and no rotation is provided in the list, a predefined constant rotation is used when the particles are extracted which must be specified beforehand.

Note, that the extraction does not interpolate the data, which means that the extracted subvolumes are stored with the same orientation as in the original map. If rotations were recorded

during extraction, they are not used at this point. They are applied later during processing, so that only a single interpolation of the data has to be carried out.

A required parameter for the stack generation is the size of the extracted subvolumes. All subvolumes should normally lie within the boundaries of the tomogram. By default, the program *tomoprepare* produces an error if a subvolume is not completely contained within the source tomogram, for instance if it lies close to the edge of the tomogram. It may be desirable to still use such subvolumes if the overlap is large enough and contains the relevant area of interest of the investigated structure. The minimum fraction of overlap can be redefined with the command "`area`" to allow such subvolumes to be included in the stack. In order to discard subvolumes with insufficient overlap without producing an error, the command "`set exclude`" must be used. To discard duplicate positions, use "`set duplicate`", and to eliminate positions that are too close to each other, set an appropriate exclusion radius with the command "`radius`".

Once the subvolume stacks have been created, they are combined into a single dataset. All the individual stacks are opened one by one with the command "`attach`" which merges the stacks into the dataset. Previously merged and saved datasets may also be attached at this point. No image data is copied when the stacks are merged or when the resulting dataset is saved to a new file. Only references to the original stacks are recorded, which means, that all the original stacks must be accessible when the dataset is being processed. If different window sizes are encountered while creating the dataset, the resulting window size of the dataset is the smallest size encountered. This includes the size specified with the command "`window`".

Other operations provided by the program *tomoprepare* include the binning of maps, stacks, and datasets. The following table lists all commands:

Command	Description
attach *path*	Open stack file *path* and add it to the data set.
area A	Minimum overlap area or volume: $0 < A \leq 1$.
binning B	Set binning factor B.
close	Close the data set.
cut X Y Z to *path* [clip|pad]	Extract a region centered at X Y Z from the map, creating a new map named *path* that is either clipped, or padded to the window size.
extract *pos* to *path* [with rotations]	Extract subvolumes from the map, centered at positions in file *pos*, writing the subvolume stack including the image data to *path*. Rotations in *pos* are discarded by default unless specified otherwise.
geometry *path*	Read tilt geometry file *path*.
help	Print brief command descriptions.
map *path*	Open map file named *path*. Closes previously opened file.
mapdir *pathlist*	Specify path list to search for the map file.

positions *pos* to *path* [with rotations]	Create subvolumes from the map, centered at positions in file *pos*, writing the subvolume stack excluding image data to *path*, linking positions to original map. Rotations in *pos* are discarded by default unless specified otherwise.
print	Print currently set parameters.
radius R	Exclusion radius R applied during stack creation.
rebin B [from *name*] [to *dir*]	Change binning factor of data set to B, optionally using input file *name* and writing to directory *dir*.
recalc	Recalculate parameters.
remap S [from *name*] [to *dir*]	Re-extract dataset with sampling factor S from existing maps, optionally using input file *name* and writing to directory *dir*.
reset	Reset parameters and close all files.
rotation $a_{11}a_{12}a_{13}a_{21}a_{22}a_{23}a_{31}a_{32}a_{33}$	Specify default rotation matrix.
save *path*	Save data set to file named *path*.
search *pathlist*	Path list to search for stack files.
set [log|exclude|duplicate|original]	Set processing flags: "log" to print diagnostic messages, "exclude" to skip subvolumes with insufficient overlap, "duplicate" to skip duplicate positions, "original" to use original map files in commands rebin and remap.
window X Y Z	Specify window size.
writemap to *path*	Write map to file *path* with current binning factor.
writestack from *stk* to *path*	Write stack *stk* to file *path* with current binning factor.
quit [discard]	Exit the program.

Table 14: *tomoprepare commands*

6.3 Subvolume alignment

Processing is normally carried out with the scripts listed in table 17 below. The scripts call two programs, *tomoprocess* (Table 15) for alignment related operations, and *tomoclass* (Table 16) for multivariate statistical analysis and classification. In *tomoprocess*, parameters such as image masks and Fourier space filters need to be specified before the reference files are opened with the command "`reference`" and before the reference is generated with "`makereference`". The same applies for running the alignment. First, the dataset is opened with the command "`dataset`", then alignment parameters and options are set, and finally the alignment is started with the command "`align`". *tomoprocess* also provides two additional commands: "`average`" to produce averages of the dataset or individual classes, and "`resample`" to write aligned subvolumes to an

image stack. For diagnostic purposes the generated masks, filters, and references may also be written to files with the commands "`writemask`", "`writefilter`", and "`writereference`".

Command	Description
align to *path* [log]	Run alignment with previously set parameters and write results to file *path*.
area A	Minimum overlap area or volume: $0 < A \leq 1$.
average [even|odd] to *path* [log]	Compute averages and write results to file *path*.
binning B	Set binning factor B.
classes C	Select classification with C classes for averaging.
classmembers *path*	Write number of class members to text file *path*.
clear	Clear all mask parameters.
corr-mode xcf|mcf|pcf|dbl	Set correlation mode.
corr-path *path*	Write correlation peak images to file *path*.
dataset *path* [,*path*] ...	Define and open dataset specified by a list of file names.
elliptic X Y Z [apod x y z]	Define ellipsoidal real space mask, optionally apodized.
gaussian X Y Z	Define Gaussian real space mask.
gauss-weight X Y Z	Define Gaussian Fourier space weighting function.
help	Print brief command descriptions.
high-pass X Y Z [apod x y z]	Define high-pass filter, optionally apodized.
low-pass X Y Z [apod x y z]	Define low-pass filter, optionally apodized.
limit R [R]	Define grid search limits.
makereference [to *path*] [log]	Create alignment reference with previously set parameters and opened reference files, and optionally write it to file *path*.
molecular *path*	Define a molecular mask, read from file *path*.
openreference *path*	Open and use reference previously written to file *path*.
peakradius X Y Z	Define peak search range.
peakcmradius X Y Z	Define range for center of mass calculation in peak search.
polar	Perform spin search in polar coordinates.
print	Print current parameters.
rectangular X Y Z [apod x y z]	Define rectangular real space mask, optionally apodized.
reference *path* [,*path*] ...	Define and open reference files specified by a list of file names.

resample to *path* [log]	Write resampled subvolumes to file *path*.
reset	Reset all parameters.
search *pathlist*	Path list to search for datasets.
select ccc|ref|class *S*	Subvolume selection.
set [log|exclude|wedge-comp]	Set processing flags: "log" to print diagnostic messages, "exclude" to skip subvolumes with insufficient overlap, "wedge-comp" to enable missing wedge compensation.
steps *S* [*S*]	Define grid search steps.
wedge-comp	Enable missing wedge compensation.
window *X Y Z*	Set subvolume window size.
writefilter to *path*	Write Fourier space filter to file *path*.
writemask to *path*	Write real space mask to file *path*.
writereference to *path*	Write reference images to file *path*.
quit	Exit the program.

Table 15: *tomoprocess commands*

6.4 Subvolume classification

tomoclass provides two main operations: the first one is to compute factorial coordinates with a singular value decomposition of the data matrix generated from the aligned subvolumes (command "`svd`"), and the second one is hierarchical ascendant classification, using the factorial coordinates (command "`hac`") computed by "`svd`". A data reduction step can be enabled by setting a binning factor for generating the data matrix. The binning takes place before the subvolumes are resampled to the aligned orientations when generating the matrix for singular value decomposition. Note, that parameters for masks and filters need also be adjusted to the new dimensions of the binned subvolumes.

Command	Description
area *A*	Minimum overlap area or volume: $0 < A \leq 1$.
binning *B*	Set binning factor *B*.
classes *MIN MAX STEP*	Set limits (*MIN MAX*) and increment (*STEP*) on number of classes to compute.
clear	Clear all mask parameters.
dataset *path* [,*path*] ...	Define and open dataset specified by a list of file names.
elliptic *X Y Z* [apod *x y z*]	Define ellipsoidal real space mask, optionally apodized.
gaussian *X Y Z*	Define Gaussian real space mask.

gauss-weight *X Y Z*	Define Gaussian Fourier space weighting function.
hac *COO* to *CLS MEM* [log]	Compute hierarchical ascendant classification from factorial coordinates in file *COO*. Write results to text file *CLS* and class memberships to file *MEM*.
hacoptions *HVO HVM*	Specify fraction of high-variance outliers *HVO* and high-variance class members *HVM* to discard.
hacfactors *F*	Define factors to use in the classification. *F* is a comma separated list of factor numbers or factor ranges.
help	Print brief command descriptions.
high-pass *X Y Z* [apod *x y z*]	Define high-pass filter, optionally apodized.
low-pass *X Y Z* [apod *x y z*]	Define low-pass filter, optionally apodized.
msamask [*path*]	Create mask for MSA with previously set parameters, or read mask from file *path*.
molecular *path*	Define a molecular mask, read from file *path*.
print	Print current parameters.
rectangular *X Y Z* [apod *x y z*]	Define rectangular real space mask, optionally apodized.
rowstat *path*	Write matrix row statistics to file *path*.
select ccc|ref|class *S*	Subvolume selection.
set [log|wedge-comp]	Set processing flags: "log" to print diagnostic messages, "wedge-comp" to enable missing wedge compensation.
svd to *SV RSV COO* [log]	Compute singular value decomposition and write singular values to file *SV*, right singular vectors to file *RSV*, and factorial coordinates to file *COO*.
svdfactors *F*	Set maximum number of factors to compute.
threshold *T*	Set density threshold for MSA mask.
variance *path*	Write variance image to file *path*.
wedge-comp	Enable missing wedge compensation.
window *X Y Z*	Set subvolume window size.
quit	Exit the program.

Table 16: *tomoclass commands*

6.5 Processing scripts

The processing scripts are listed below in the order in which they are normally executed (for cycle number *N*). Note, that `subvolreference.sh` must be executed before `subvolalign.sh`,

`subvolmsamask.sh` before `subvolsvd.sh`, and `subvolhac.sh` after `subvolsvd.sh`, ect. The listed order allows the time-consuming parts, starting with `subvolalign.sh` to be run without user intervention.

Before running the initialization for a new cycle, either with `subvolinitial.sh` or `subvolnext.sh`, a parameter file with the name `template-initial.sh` must be created that contains all parameters for the next cycle to be executed. The initial dataset must be prepared as described in an earlier section, and its file name is passed to the script `subvolinitial.sh`. If the initial dataset references other datasets or subvolume stacks, these must be stored in a single directory, which is specified in the parameter file. The initialization scripts create a new subdirectory for the new cycle where all intermediate results are stored, including the input parameters. If a particular operation in the current cycle needs to be rerun with different parameters, the copy of the parameter file in the subdirectory must be edited, not the template file.

For more details about the use of the scripts refer to the *protomo* subvolume tutorial.

Script	Description
subvolinitial.sh S	Initialize the alignment of a new dataset S.
subvolmsamask.sh N	Create MSA mask for visualization.
subvolreference.sh N	Create alignment reference(s).
subvolalign.sh N	Alignment.
subvolsvd.sh N	Calculate factorial coordinates for classification.
subvolhac.sh N	Classification.
subvolclassaverage.sh N	Produce class averages.
subvolclassalign.sh N	Align class averages to each other.
subvolnext.sh N M	Use results of cycle N to initialize the next cycle M.

Table 17: *Subvolume processing scripts*

7 Tools

7.1 Image display

The program "`i3display`" displays 2D or 3D images, and is invoked at the shell prompt as follows:

`i3display [-r` *min max*`]` *image*

`-r` *min max*	density thresholds
image	2D or 3D image file name

The command will display the image with the file name *image*, and if the option `-r` is given, it thresholds the densities below *min* and above *max*. If the option is absent, it first scans the image to find the minimal and maximal densities and uses these values to scale the densities for displaying the image. It recognizes the keys/buttons listed in table 18. Stacks of 3D images cannot be displayed directly. Use the program "`i3montage`" to create a montage first, and display the montage.

Key/button	Action
up arrow	Display next higher section.
down arrow	Display next lower section.
left mouse	Print coordinates.
left mouse + drag	Pan.
middle mouse + drag	Zoom.
shift + left mouse	Pick position.
ctrl + left mouse	Delete position.
L l	Display current z-level markers only.
P p	Toggle marker display on/off.
Q q	Display cross/square markers.

Table 18: *i3display: key/button press actions*

For particle picking, an additional file name *output* is supplied, and optionally another file name *posname* to read positions from a text file if a position list needs to be edited:

`i3display [-r` *min max*`] [-pos` *posname*`]` *image output*

`-r` *min max*	density thresholds
`-pos` *posname*	read position list from file *posname*
image	2D or 3D image file name
output	write positions to text file *output*

Particle coordinates must appear as space separated values in the input file, one set of coordinates per line for each particle. Positions are marked in the following way in the image: yellow markers indicate that the position is located at the currently displayed z-level, green markers indicate a level above, and blue markers a level below the displayed section.

7.2 Tilt series alignment

The program "tomoalign-gui" is used for manual alignment of tilt series, or for displaying and saving animated sequences of tilt series:

`tomoalign-gui [-log] [-zoom` *fac*`] [-r` *min max*`] [-tlt` *geom*`]` *param*

`-log`	enable logging of diagnostic information
`-zoom` *fac*	set zoom factor to *fac*
`-r` *min max*	density thresholds
`-tlt` *geom*	read tilt geometry from file *geom*
param	parameter file name

If the `-log` option is present, diagnostic information is printed to the terminal. Option `-zoom` sets the initial zoom factor, and `-r` sets the density scaling as in the program "`i3display`". If the specified tilt series is a new series and the geometry metadata file does not exist yet, the option `-tlt` is mandatory to specify the geometry file. Key/button actions to display and manipulate the images are listed in table 19.

Key/button	Action
left mouse	Print coordinates.
left mouse + drag	Align image manually.
A a	Align image manually.
I i	Toggle image display.
R r	Reset alignment.
-	Zoom out.
+	Zoom in.

Table 19: *tomoalign: key/button press actions*

On startup, or when the overlay mode is selected from the menu, the program displays two images, a reference image in red, and the image to be aligned in green. The superposition of the two colors results in a more or less grey tone image if the two superimposed images are in register. To align manually, the green image can be dragged to the aligned position while holding down the left mouse button. The automatic alignment function, selected from the menu, performs a cross-correlation alignment. The differences in foreshortening of the images is compensated, but the reference construction scheme used in area matching is not applied, the two images are simply cross-correlated and the displacement calculated according to the correlation maximum.

7.3 Manipulating image stacks

The program "`i3concat`" (see program reference) is used to create image stacks. To separate a stack into individual images use the script "`i3unstack.sh`". The output images are numbered consecutively and an optional file name prefix and/or suffix can be chosen.

`i3unstack.sh` *`inputfile`* [*`prefix`* [*`suffix`*]]

`inputfile`	image stack of 3D images
`prefix`	output file name prefix (optional)
`suffix`	output file name suffix (optional)

7.4 Printing image file metadata

The program "`tomoinfo`" prints metadata stored in image files. It returns silently without error if the requested type of information is absent in the file.

`tomoinfo [option]` *`inputfile`*

`inputfile`	any image file
`option`	specifies type of information to be printed as listed in table 20

Option	Description
`-ali`	Reference numbers to which subvolumes were aligned
`-ccc`	Print cross-correlation coefficients
`-cccstat`	Print statistics of cross-correlation coefficients
`-cls`	Print classifications
`-ctr`	Positions in original map and file references
`-pre`	Saved transformations before alignment
`-ref`	Print file references
`-tlt`	Print tilt geometry
`-trf`	Active transformations
`-vol`	Print area/volume coverage ratios
`-volstat`	Print statistics of area/volume coverage
`-wdg`	Print missing wedge information

Table 20: *tomoinfo options*

The most important type of metadata are the center positions of the subvolumes specified relative to the maps from which the subvolumes were extracted. The associated map is identified with an

index to a file reference metadata entry which identifies the particular map. Positions are listed with the option `-ctr`, the file reference indices and associated file names of the maps are listed with option "`-ref`". Alignment related data consists of the transformations from a coordinate system specified relative to the image files (the pixel raster) to a coordinate system relative to the specimen (i. e. the reference). Applying the transformations to the images in the dataset produces subvolumes in orientations suitable for averaging or classification. An alignment produces a new set of transformations that is stored as the active set, while the former active set is saved. The currently active transformations are listed with option `-trf`, the saved transformations with `-pre`.

8 References

Taylor, K. A., Liu, J. and Winkler, H. (2006). Localization and classification of repetitive structures in electron tomograms of paracrystalline assemblies. In Electron Tomography. Methods for three-dimensional visualization of structures in the cell, (Frank, J., ed.), pp. 417–439. Springer New York.

Taylor, K. A., Tang, J., Cheng, Y. and Winkler, H. (1997). The use of electron tomography for structural analysis of disordered protein arrays. J. Struct. Biol. *120*, 372–386.

Winkler, H. (2007). 3D reconstruction and processing of volumetric data in cryo-electron tomography. J. Struct. Biol. *157*, 126–137.

Winkler, H. and Taylor, K. A. (1999). Multivariate statistical analysis of three-dimensional cross-bridge motifs in insect flight muscle. Ultramicroscopy *77*, 141–152.

Winkler, H. and Taylor, K. A. (2006). Accurate marker-free alignment with simultaneous geometry determination and reconstruction of tilt series in electron tomography. Ultramicroscopy *106*, 240–254.

Winkler, H. and Taylor, K. A. (2013). Marker-free dual-axis tilt series alignment. J. Struct. Biol. *182*, 117–124.

Winkler, H., Zhu, P., Liu, J., Ye, F., Roux, K. H. and Taylor, K. A. (2009). Tomographic subvolume alignment and subvolume classification applied to myosin V and SIV envelope spikes. J. Struct. Biol. *165*, 64–77.