

Amira Deconvolution Package 1.0

An Amira extension for enhancing the quality of optical sections

Copyright Information

©1999-2002 Indeed - Visual Concepts GmbH, Berlin, Germany

©1999-2002 TGS Template Graphics Software, Inc., USA

All rights reserved.

Amira and the Amira Deconvolution Package are developed by Indeed - Visual Concepts GmbH (www.indeed3d.com) and are marketed and distributed exclusively by Template Graphics Software Inc. (www.tgs.com).

This document contains proprietary information of Indeed - Visual Concepts GmbH, and is protected by German and international law. The contents of this document may not be translated, copied, or duplicated in any form, in whole or in part, without the express written permission of Indeed - Visual Concepts GmbH.

The information contained in this document is subject to change without notice. Neither Indeed - Visual Concepts GmbH nor their employees shall be responsible for incidental or consequential damages resulting from the use of this material or liable for technical or editorial omissions or errors made herein.

Contents

1	Amira Deconvolution Package	5
1.1	General remarks about image deconvolution	5
1.2	Data aquisition and sampling rates	6
1.3	Standard Deconvolution Tutorial	7
1.4	Blind Deconvolution Tutorial	13
1.5	Bead Extraction Tutorial	16
1.6	Performance issues and multi-processing	19
2	Module Reference	21
2.1	BeadExtract	21
2.2	Convolution	22
2.3	CorrectZDrop	23
2.4	DataPreprocess	24
2.5	Deconvolution	25
2.6	FourierTransform	27
2.7	PSFGen	28
3	Examples	30
	Index	

Chapter 1

Amira Deconvolution Package

The Amira Deconvolution Package provides powerful algorithms for improving the quality of microscopic images recorded by 3D widefield and confocal microscopes. Two different methods are supported, namely a so-called non-blind and a blind deconvolution method, both based on iterative maximum-likelihood image restoration. In the first case a measured or computed point spread function (PSF) is required. In the second case the PSF is estimated along with the data itself.

The documentation of the Amira Deconvolution Package is separated into the following parts:

- *General remarks about image deconvolution*
- *Data acquisition and sampling rates*
- *Standard deconvolution tutorial*
- *Blind deconvolution tutorial*
- *Bead extraction tutorial*
- *Performance issues and multi-processing*

The following modules are part of the Amira Deconvolution Package:

- *BeadExtract* - obtain a PSF from a bead measurement
- *Convolution* - convolve two 3D images
- *CorrectZDrop* - corrects attenuation in z-direction
- *DataPreprocess* - background and flatfield correction
- *Deconvolution* - the actual deconvolution front-end
- *FourierTransform* - computes FFT and power spectrum
- *PSFGenerate* - calculates a theoretical PSF

1.1 General remarks about image deconvolution

Deconvolution is a technique for removing out-of-focus light in a series of images recorded via optical sectioning microscopy. Intended to investigate 3D biological objects, optical sectioning microscopy works by creating multiple images (optical sections) of a fluorescing object, each with a different focus plane. However, besides the in-focus structures the images usually also contain out-of-focus light from other parts of the object, causing haze and severe axial blur. This is even the case for a confocal laser scanning microscope, where most of the out-of-focus light is removed from the image by a pinhole system. Mathe-

matically, the image produced by any microscopic system can be described as the convolution of the ideal unblurred image of the specimen and the microscope's so-called point spread function (PSF), i.e., the image of an ideal point light source. With the inverse of this process, called deconvolution, a deblurred image of the specimen can be obtained, provided the point spread function is known or at least, can be estimated.

The Amira deconvolution package mainly provides two variants of a powerful iterative maximum-likelihood image restoration algorithm, namely a non-blind one and a blind one. The difference between them is that in the first case a measured or computed point spread function is used, while in the second case the PSF is estimated along with the data itself. Maximum-likelihood image restoration can be considered as the de-facto standard for deconvolution of 3D optical sections. Although computationally quite expensive, the method is able to significantly enhance image quality. At the same time it is very robust and insensitive with respect to noise artifacts. However, it should be noted that, although rejecting most of the out-of-focus light, by no means all of it is rejected. Therefore, some noticeable haze remains in the images. Also, the images retain a substantial axial smearing in z-direction, which cannot be removed by any deconvolution algorithm.

At the first sight one may wonder why both a non-blind and a blind deconvolution algorithm are provided in Amira, although blind deconvolution seems to be more general since the PSF is calculated automatically. One answer is that blind deconvolution is computationally even more expensive than non-blind iterative maximum-likelihood image restoration. The other answer is that in a blind deconvolution algorithm a meaningful estimate of the PSF can only be computed if severe constraints are imposed. For example, a trivial solution of the blind deconvolution problem would be a deblurred image which is identical to the input image and a PSF with the shape of an ideal delta peak. Obviously, this solution isn't useful at all. Therefore, if for example confocal data is to be deconvolved the algorithm fits the actual PSF in such a way that it looks like a possible measured PSF of a confocal microscope. More precisely, the fit is constraint to be in agreement with the experimental parameters (the refractive index of the medium, the numerical aperture of the objective and the voxel-sizes). Sometimes, this can lead to wrong results, for example when the confocal pinhole aperture of the microscope wasn't stopped down sufficiently during confocal image acquisition, in which case the microscope actually didn't behave like a true confocal microscope. As a matter of fact you should try which approach provides the best results for your own image data, blind deconvolution or non-blind deconvolution with either a measured or an automatically computed PSF.

1.2 Data acquisition and sampling rates

In order to obtain best quality when deconvolving microscopic images some fundamental guidelines should be obeyed during image acquisition. Good results may be obtained even if some of these guidelines are not followed exactly, but in general the chances to get satisfactory results improve if they are. Below we discuss the most important recommendations.

Adjusting the scanned image volume

The region of interest should be centered in the middle of the image volume, as the optics of the microscope has usually the least aberrations in this region and it helps to avoid possible boundary artifacts, which can arise during the deconvolution procedure. Especially for widefield data it is important to record a sufficiently large (preferably empty) region below and above the actual sample. Ideally, this region should be as large as the sample itself. For example, if the sample covers 100 micrometer in z-direction, the scanned image volume should range from 50 micrometer below the sample to 50 micrometer above it.

Choosing the right sampling rate

The sampling rate is determined by the pixel sizes in x and y direction as well as the distance between two subsequent optical sections, both measured in micrometers. Generally speaking, image deconvolution works best if the data is apparently oversampled, i.e., if the pixel or optical section spacing is smaller than required. The maximal required sampling distance (Nyquist sampling) to avoid ambiguities in the data can be obtained from considerations in Fourier-space yielding

$$d_{xy} = \frac{\lambda}{4 NA},$$

where λ denotes the wavelength and NA is the numerical aperture of the microscope. Similar considerations yield for the maximal distance between adjacent imaged planes:

$$d_z = \frac{\lambda}{2 n (1 - \cos(\alpha))},$$

where n denotes the refractive index of the object medium and α the aperture half angle as determined by $NA = n \sin(\alpha)$.

For a confocal microscope both the in-plane sampling distance and the axial sampling distance need to be in theory approximately 2 times smaller. However, this requirement is far too strict for most practical cases and even in the widefield case, approximately fulfilling the above requirements is often sufficient.

The total number of optical sections is obtained by dividing the height of the image volume by the sampling distance d_z . It should be mentioned that deconvolution also works if the sampling distances are not matched rigorously, but matching them improves the chances to get good results. In general oversampling the object is less harmful than undersampling it, with one exception: In case of confocal data the sampling distance d_{xy} should not be much smaller than indicated, if the blind deconvolution algorithm or the non-blind deconvolution algorithm together with a theoretically computed confocal PSF are used. Otherwise the unconstrained Maximum Likelihood algorithm and the predominant noise in the data might lead to unsatisfactory results.

Black level and saturation

Before grabbing images from the microscope's camera the light level should be adjusted in such a way that saturated pixels, either black or white ones, are avoided. Saturated pixels are pixels which are clamped to either black or white, because their actual intensity values are outside the range of representable intensities. In any case, saturation means a loss of information and thus prevents proper post-processing or deconvolution. At the same time an extensively high blackground level should be avoided, since it decreases the dynamic range of the imaging system and the deconvolutions perform worse. This means that empty regions not showing any fluorescence should appear almost black. A background level close to zero is especially important when bead measurements are performed in order to extract an experimental point spread function. Details are discussed in a separate tutorial about *bead extraction*.

1.3 Standard Deconvolution Tutorial

This tutorial explains how 3D image data sets can be deconvolved in Amira. It is assumed that the reader is already familiar with the basic concepts of Amira itself. If this is not the case, it is strongly recommended to work through the *standard Amira tutorials* first. In this section the following topics are covered:

1. Prerequisites for deconvolution
2. Resampling a measured PSF

3. Deconvolving an image data set
4. Calculating a theoretical PSF

As an example we are going to use a confocal test data set (*polytrichum.am*) provided with the Amira deconvolution package. The data file is located in the directory *Amira-2.3/data/deconv*.

- Load the data set *polytrichum.am*.
- Visualize it for example using a *ProjectionView* module.

The data set shows four chloroplasts in a spore of the moss *polytrichum* commune.

Prerequisites for deconvolution

Besides the image data itself for the standard non-blind deconvolution algorithm also a so-called *point spread function* (PSF) is required. The PSF is the image of a single point source, or approximately, the image of a single fluorescing sub-resolution sphere. PSF images can either be computed from theory (see below) or they can be obtained from measurements. In the latter case tiny so-called *beads* are recorded under the same conditions as the actual object. This means that the same objective lens, the same dye and wavelength, and the same immersion medium are used. Typically, the images of multiple beads are averaged to obtain an estimate of a single PSF. Amira provides a module called *BeadExtract* facilitating this process. The use of this module is discussed in a *separate tutorial* about bead extraction. At this point let us simply load a measured PSF from a file.

- Load the data set *polytrichum-psf.am*.
- Use the *ProjectionView* module to visualize it.

The PSF appears as a bright spot located in the middle of the image volume (Figure 1.1). It is important that the PSF is exactly centered. Otherwise, the deconvolved data set will be shifted with respect to the original image. Also, it is important that the PSF fades out to black at the boundaries. If this is not the case, the black level of the PSF image needs to be adjusted using the *Arithmetic* module. Finally, neither the PSF nor the image to be deconvolved should exhibit *intensity attenuation artifacts*, i.e., image slices with decreased average intensity due to excessive light absorption in other slices. If such artifacts are present, they can be removed using the *CorrectZDrop* module.

Resampling a measured PSF

Next, select both, the PSF and the image data. You'll notice that the voxel sizes of both objects are not the same. It is recommended to adjust different voxel sizes of PSF and image data prior to deconvolution using the *Resample* module. The deconvolution module itself also accounts for different voxel sizes, but it does so by using point sampling with trilinear interpolation. This is ok as long as the voxel size of the PSF is larger than that of the image data. However, in our case the voxel size of the PSF is smaller than that of the image data, i.e., the resolution of the PSF is higher. Using the *Resample* module provides slightly more accurate results here, since all samples will be filtered correctly using a Lanczos kernel.

- Connect a *Resample* module to *polytrichum-psf.am*.
- Connect the *Reference* port of the *Resample* module to the image data set *polytrichum.am*
- In the *Mode* port of the *Resample* module, choose *voxel size* (see Figure 1.2).
- Resample the PSF by pressing the *DoIt* button.

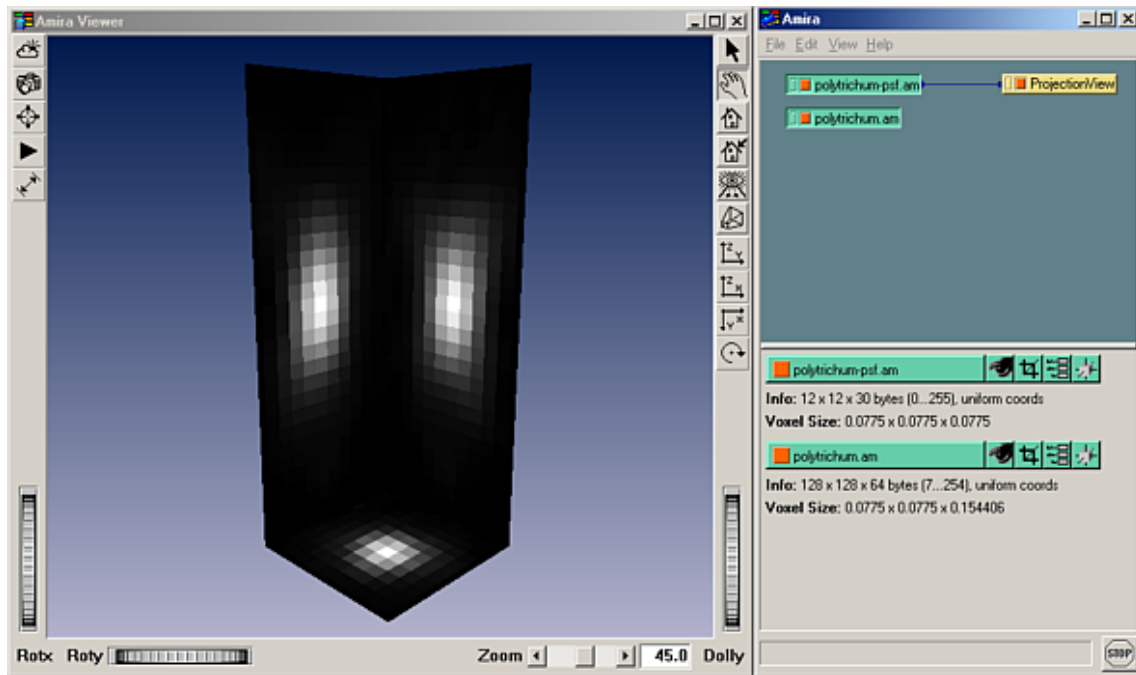


Figure 1.1: Maximum intensity projection of *polytrichum-psf.am*

The *voxel size* option means that the PSF will be resampled on a grid with exactly the same voxel size as the image data set, which is connected to the *Reference* port. While the original PSF had a resolution of 12 x 12 x 30 voxels, the resampled one only has 12 x 12 x 16 voxels. However, the extent of a single voxel in z-direction is bigger now.

Deconvolving an image data set

After a suitable PSF has been obtained we are ready for deconvolving the image data set. This can be done by attaching a *Deconvolution* module to the image data.

- Connect a *Deconvolution* module to *polytrichum.am*.
- Connect the *Kernel* port of the *Deconvolution* module to the resampled PSF *polytrichum-psf.Resampled*.

Once the deconvolution module is connected to its two input objects, some additional parameters need to be adjusted (for a detailed discussion of these parameters see also the *reference documentation* of the *Deconvolution* module itself). Figure 1.3 shows these settings:

Border width: For deconvolution the image data has to be enlarged by a guardband region. Otherwise boundary artifacts can occur, i.e., information from one side of the data can be passed to the other. There is no need to make the border bigger than the size of the PSF. However, if the data set is dark at the boundaries a smaller border width is sufficient. In our case, let us choose the border values 0, 0, and 8 in the x, y, and z direction.

Iterations: The number of iterations of the deconvolution algorithm. Let us choose a value of 20 here.

Initial estimate: Specifies the initial estimate of the deconvolution algorithm. If *const* is chosen a constant image is used initially. This is the most robust choice, yielding good results even if the input data is very

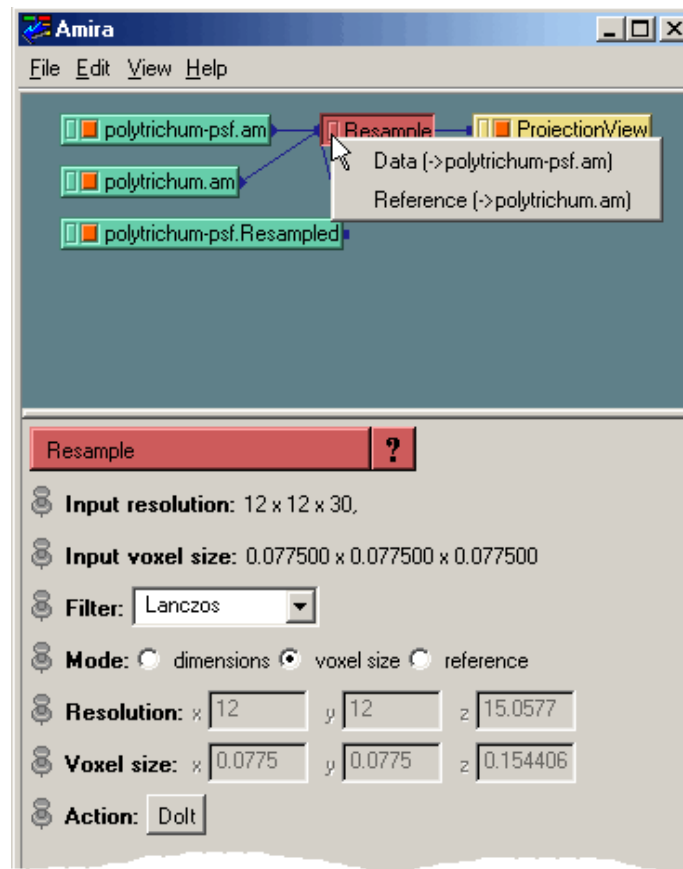


Figure 1.2: Resampling a PSF using the *Resample* module.

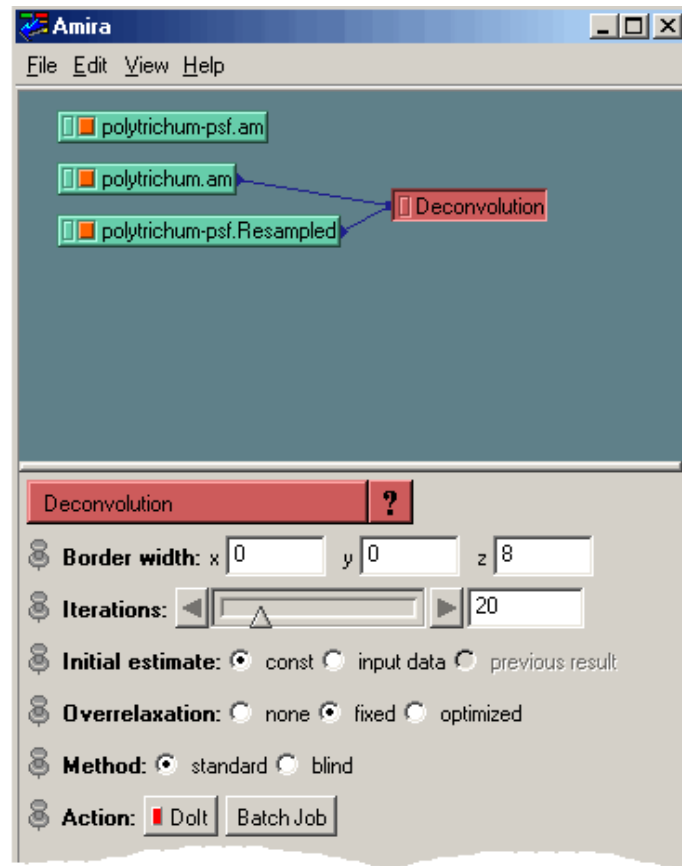


Figure 1.3: Deconvolution module attached to *polytrichum.am*.

noisy. We keep this option here.

Overrelaxation: Overrelaxation is a technique to speed up the convergence of the iterative deconvolution process. In most cases the best compromise between speed and quality is *fixed* overrelaxation. Therefore we keep this choice also.

Method: Selects between standard (non/blind) and blind deconvolution. Lets specify *standard* here.

The actual deconvolution process is started by pressing the *DoIt* button. Please press this button now. The deconvolution should take about 1-2 minutes on a modern computer. During the deconvolution the progress bar informs you about the status of the operation. Also, after every iteration a message is printed in the Amira console window indicating the amount of change of the data. If the change seems to be small enough you can terminate the deconvolution procedure by pressing the stop button. However, note that the stop button is evaluated only once between two consecutive iterations.

When deconvolution is finished a new data set called *polytrichum.deconv* appears in the object pool. You might take a look at the deconvolved data by moving the *ProjectionView* connection line from *polytrichum.am* to *polytrichum.deconv*.

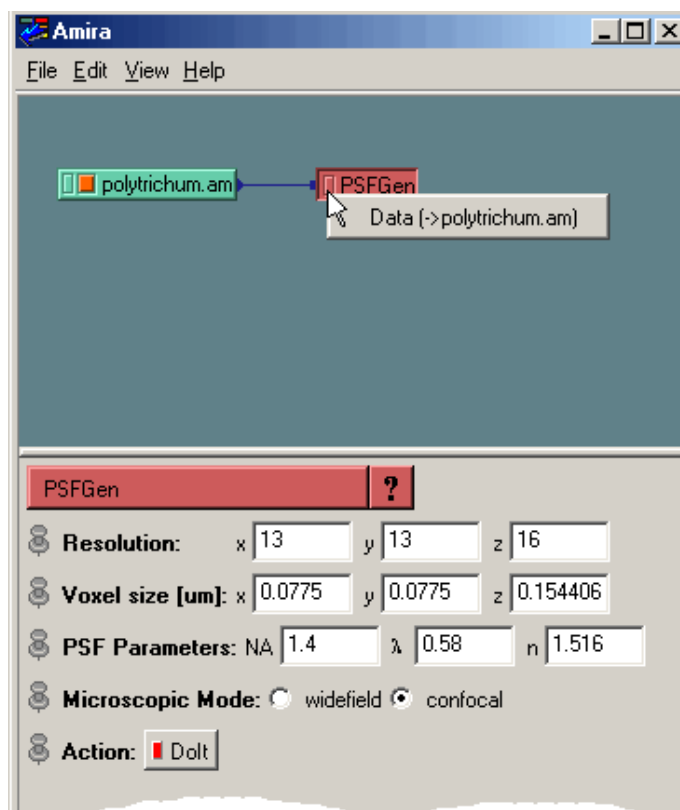


Figure 1.4: The PSFGen module calculates theoretical PSFs.

Calculating a theoretical PSF

Sometimes bead measurements are difficult to perform, so that an experimental PSF cannot easily be obtained. In such cases a theoretical PSF can be used instead. Amira provides the module *PSFGenerate*, allowing you to calculate theoretical PSFs. The module can be created by selecting *PSFGenerate* from the *Edit Create* menu of the Amira main window.

Once the module is created again some parameters have to be entered. The *resolution* and the *voxel size* can be most easily specified by connecting the *Data* port of the *PSGGenerate* module to the image data set to be convolved. In our case please connect this port to *polytrichum.am*.

In order to generate a PSF you also need to know the numerical aperture of the microscope objective, the wavelength of the emitted light (to be entered in micrometers!), and the refractive index of the immersion medium. In our test example these values are NA=1.4, lambda=0.58, and n=1.516 (oil medium). Also, change the microscopic mode from widefield to confocal.

After pressing the *DoIt* button the computed PSF appears as an icon labelled *PSF* in the object pool. You can compare the theoretical PSF with the measured one using the *OrthoSlice* module. You'll notice that the measured PSF appears to be slightly wider. This is a common observation in many experiments.

Once you have computed a theoretical PSF you can perform non-blind deconvolution as described above. However, for convenience the *Deconvolution* module is also able to compute a theoretical PSF by itself. You can check this by disconnecting the *Kernel* port of the *Deconvolution* module. If no input is present at this port, additional input fields are shown, allowing you to enter the same parameters numerical aperture,

wavelength, refractive index, and microscopic mode as in *PSFGenerate*. After these parameters have been entered the deconvolution process again can be started by pressing the *DoIt* button.

Note, that any previous result connected to the *Deconvolution* module will be overwritten when starting the deconvolution process again. Therefore, be sure to disconnect a previous result if you want to compare deconvolution with different input PSFs.

1.4 Blind Deconvolution Tutorial

This tutorial explains how blind deconvolution can be performed in Amira. At the same time it describes how deconvolution jobs can be processed using the Amira job queue. Like in the previous tutorial it is assumed that the reader is already familiar with the basic concepts of Amira itself. If not, we recommend to work through the *standard Amira tutorials* first.

A blind deconvolution example

Let us start by loading a raw image data set first.

- Load the file *alphalobe.am* from the directory *Amira-2.3/data/deconv*.
- Visualize the data set by attaching a *ProjectionView* module to it.

The data set has been recorded using a widefield microscope. It shows a stained neuron, the alpha-neuron, from a part of the brain of a bee called the optical lobe. Compared to the confocal data set used in the standard deconvolution tutorial *alphalobe.am* is much bigger. It has a resolution of 248 x 248 x 256 voxels with a uniform voxel size of 1 micrometer. In the xy-plane of the projection view the structure of the neuron can be clearly identified. However, the contrast of the image is quite poor because there is a significant amount of out-of-focus light or haze present. With Amira's blind deconvolution algorithm we can enhance the image data without needing to know an explicit PSF in advance.

- Attach a deconvolution module to *alphalobe.am*.
- Adjust the parameters like shown in Figure 1.5.

The individual parameters have the following meaning:

Border width: Like for standard non-blind deconvolution the image data has to be enlarged by a guardband region. Otherwise boundary artifacts can occur, i.e., information from one side of the data can be passed to the other. In our case we only provide a small guardband region of 8 voxels in x- and y-direction. In z-direction we don't provide any border because there are sufficiently many empty slices below and above the actual neuron. The resulting size of the data arrays on which the computations are performed then is 256 x 256 x 256, which is nice because it is a power of two and thus the fast Fourier-transforms, the computationally most expensive part of the deconvolution algorithm, can be executed somewhat faster.

Iterations: We choose a value of 25 here. Depending on the data, usually at least 10 iterations are required. With overrelaxation being enabled (see below), results usually don't improve much after 40 iterations.

Initial estimate: Specifies the initial estimate of the deconvolution algorithm. Since there is not much noise present in the original *alphalobe* images it is safe to choose *input data* here. This causes the algorithm to converge even faster.

Overrelaxation: Overrelaxation is a technique to speed up the convergence of the iterative deconvolution process. We enable overrelaxation by choosing the *fixed* toggle.

Method: We chose *blind* here in order to select the blind deconvolution algorithm.

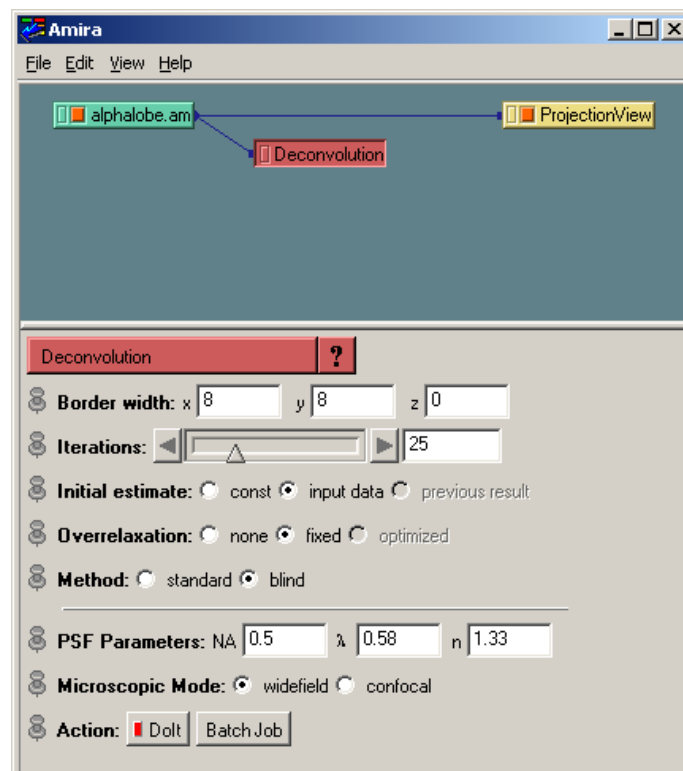


Figure 1.5: Parameters for blind deconvolution.

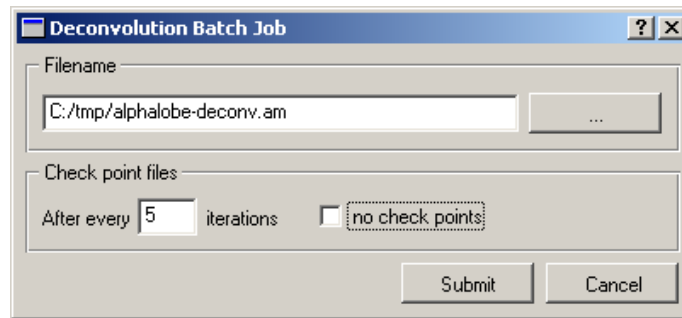


Figure 1.6: Dialog for submitting a deconvolution job.

PSF Parameters: For *alpha-lobes.am* the numerical aperture is 0.5, the wavelength is 0.58 micrometers, and the refractive index is 1.33 (water). These parameters are required in order to apply certain constraints to the estimated point spread function. They are also used in order to compute an initial PSF. If a data set would be connected to the *Kernel* port of the deconvolution module, this data set would be used as the initial PSF with the given PSF parameters still acting as constraints. For example, you could provide a measured PSF and let it be fitted to the actual data by the deconvolution algorithm.

Microscopic mode: *alpha-lobes.am* is a widefield data set, so select this option here.

Submitting a deconvolution job

After all parameters have been entered, the deconvolution process can be started. On a modern computer blind deconvolution of our test data set roughly takes about 20 minutes. Especially, if you want to deconvolve multiple data sets at once it is inconvenient to do this in an interactive session. Therefore multiple deconvolution jobs can be submitted to the Amira job queue and then, for example, processed over night. This works as follows:

- Press the *Batch Job* button of the *Action* port. A dialog as shown in Figure 1.6 pops up.
- In the dialog choose a file name under which you want to save the deconvolved data set, e.g. `C:/Temp/alpha-lobes-deconv.am`.
- Modify the text field, so that check point files are written after every 5 iterations.

Check point files are used to store intermediate results. With the above settings the deconvolved data is written into a file after every 5 iterations. Check point files are called like the final result, but a consecutive number is inserted just before the file name suffix. For example, if the result file name is `C:/Temp/alpha-lobes-deconv.am` the check point files are called `C:/Temp/alpha-lobes-deconv-0005.am`, `C:/Temp/alpha-lobes-deconv-0010.am` and so on. Now we are ready to actually submit the batch job.

- Press the *Submit* button of the deconvolution dialog. After a few seconds the Amira batch job dialog appears, compare Figure 1.7.
- Select the deconvolution job and press the *Start* button.

You now have to wait about 20 minutes until the deconvolution job is finished. Once the job queue has been started you can quit Amira. The batch jobs will be continued automatically. If Amira is still running when the deconvolution job exits then the result will be loaded automatically in Amira. Otherwise you have to restart Amira and load the deconvolved data set manually.

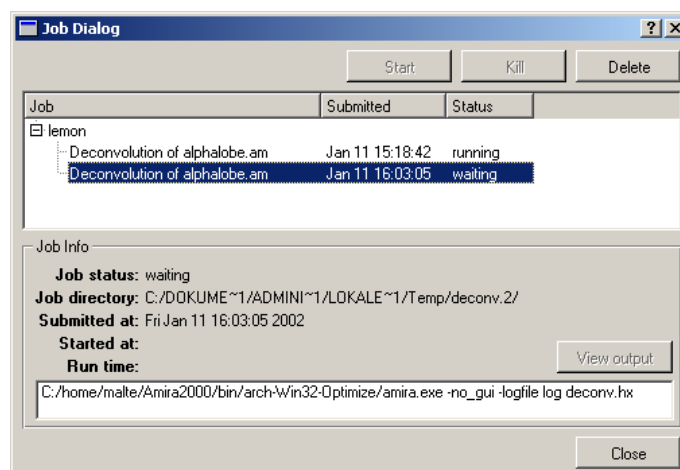


Figure 1.7: The Amira job dialog showing a pending deconvolution job.

1.5 Bead Extraction Tutorial

Non-blind deconvolution is a powerful and robust method for enhancing the quality of 3D microscopic images. However, the method requires that the image of the point spread function (PSF) responsible for image blurring is provided. As stated in the *standard deconvolution tutorial*, the PSF can either be calculated theoretically or it can be obtained from a bead measurement. Amira provides a special-purpose module called *BeadExtract* which facilitates the extraction of PSF images from one or multiple bead measurements. In this tutorial the use of the module shall be explained. The following topics are covered:

1. Bead measurements
2. Projection View and Projection View Cursor
3. Resampling and averaging the beads

Bead measurements

The PSF is the image of a single point source recorded under the same conditions as the actual specimen. It can be approximated by the image of a fluorescing sub-resolution microsphere, a so-called bead. Performing good bead measurements requires some practice and expertise. In order to obtain good results the following hints should be obeyed:

1. Use the right beads. It is important that the bead size is smaller than approximately 1/2 full width at half maximum (FWHM) of the PSF. Good sources for obtaining beads suitable for PSF measurements are Molecular Probes (<http://www.probes.com/>) or Polysciences (<http://www.polysciences.com/>).
2. The beads must be solid. Besides solid beads there are also beads with the shape of a spherical shell, allowing to check the focus plane of a microscope. Such beads cannot be used as a source for PSSF generation in the current version of Amira.
3. Don't record clusters of multiple beads. Sometimes multiple beads may glue together, appearing as a single big bright spot. Computing a PSF from such a spot obviously leads to wrong results.
4. Note that beads are not resistant to a variety of embedding media. In particular beads will be destroyed in xylene based embedding media such as Permount (Fisher Scientific) and methyl salicylate (frequently used to clear up the tissue). As a substitute you might use immersion oil instead which

has a similar refractive index as e.g. methyl salicylate.

5. The sample and the beads should always be imaged as close to the coverslip as possible. When it is not possible to attach the sample to the coverslip, the beads should also be imaged in a comparable depth, embedded in the same mounting medium. Imaging the beads with better quality than the sample will yield a slightly blurred deconvolution result. However, when the PSF used for deconvolution is too wide, artefacts can arise during deconvolution.

The objective lense should always be selected according to the mounting medium, i.e. if the sample is attached to the slide and embedded in a buffer of refractive index close to water, a severe loss of image quality can be expected when using an oil-immersion objective without a correction collar. Deconvolution of properly imaged data will allways be superior to deconvolution of data suffering from aberrations.

6. Problems occur if the mounting medium remains liquid. In this case the sample distribution may not be permanent. If your specimen is to be embedded in water you can try to immerse the beads in an agarose gel of moderate concentration instead. Attaching the small beads to the coverslip (for example by letting them dry) is often also sufficient for immobilisation.

An example of an image data set containing multiple beads is provided in the file *beads.am* in the directory *Amira-2.3/data/deconv*.

- Load the data set *beads.am*.
- Visualize the data using a *ProjectionView* module.

Projection View and Projection View Cursor

The bead data set contains five different beads which can be clearly seen in the three orthogonal planes of the *ProjectionView* module. In order to obtain a single PSF we first want to select several “good” beads. These beads are then resampled and averaged, thus yielding the final PSF. A bead can be considered as “good” if it is clearly visible and if it is not superimposed by other beads (even when defocused),

Selecting “good” beads is an interactive process. It is most easily accomplished using the *ProjectionView*’s *Cursor* module. This module allows you to select a point in 3D space by clicking on one of the three planes of the *ProjectionView* module. The third coordinate is automatically set by looking for the voxel with the highest intensity. Points selected with the *Cursor* module can be stored in a *LandmarkSet* data object.

- Attach a *Cursor* module to the *ProjectionView* module.
- Click on any bead on one of the three planes.
- Store the current cursor position in a *LandmarkSet* object by pressing the *Add* button.
- Select and add some other beads too.

The landmarks need not to be located exactly at the center of a bead. The exact center positions can be fitted automatically later on.

You can remove incorrect bead positions from the landmark set by invoking the landmark set editor. In order to activate the editor select the landmark set object and press the green button with the disk-shaped symbol. If you want to add additional bead positions to an existing landmark set object, make sure that the *master port* of the landmark set object is connected to the *Cursor* module. Otherwise, a new landmark set object will be created.

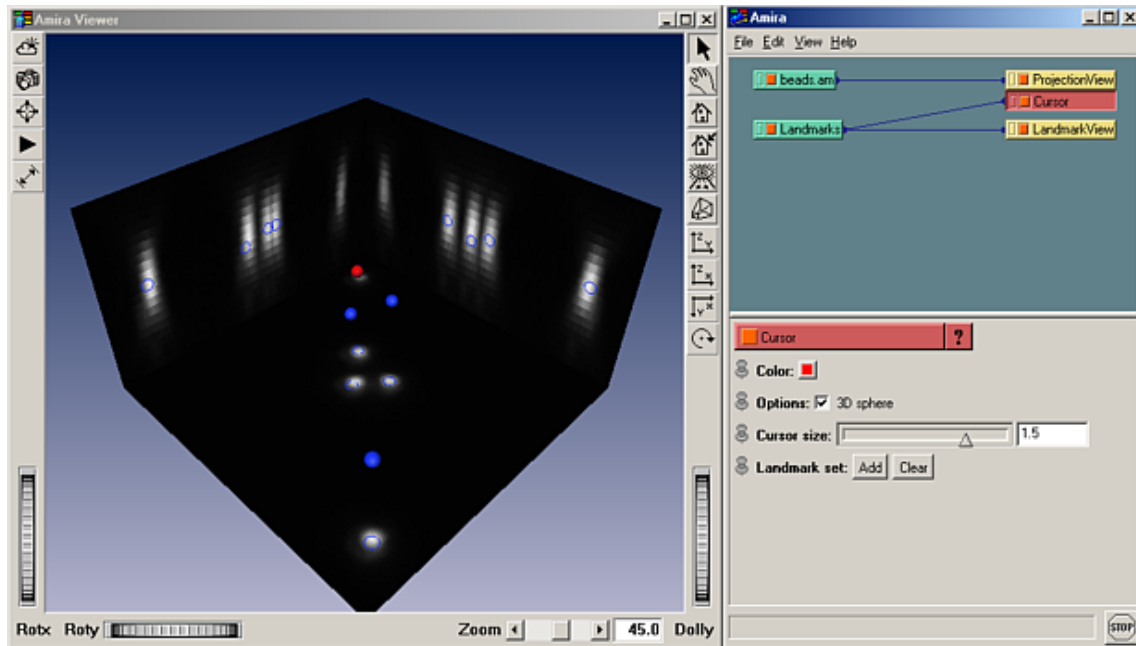


Figure 1.8: Individual beads can be interactively identified using a *Cursor* module.

Resampling and averaging the beads

Now we are ready to extract and average the individual beads. This is done by means of the *BeadExtract* module.

- Connect a *BeadExtract* module to the *Landmarks* object.
- Make sure, that the *Data* port of *BeadExtract* is connected to the bead data set *beads.am*. If the landmarks are still connected to the beads via the *Cursor* and *ProjectionView* modules, the connection is established automatically.

The *BeadExtract* module provides two buttons called *Adjust centers* and *Estimate size*, which should be invoked in a preprocessing step before the beads are actually extracted.

The first button (*Adjust centers*) modifies the landmark positions so that they are precisely located in the center of gravity of the individual beads.

The second button (*Estimate size*) computes an estimate for the number of voxels of the PSF image to be generated. This button is only active if no PSF image is connected as a result to *BeadExtract*. If there is a result object, the resolution and voxel sizes of the result are used and the port becomes insensitive.

Any of the actions of the two preprocessing buttons can be undone using the *Undo* button. This can be necessary for example if two beads are too close so that no correct center position could be computed. In general, overlaps between neighbouring beads should be avoided. Small overlaps might be tolerated because during resampling intensities are weighted according to the influence of surrounding beads.

- Perform the preprocessing steps *Adjust centers* and *Estimate size*.
- Compute a resampled and averaged PSF by pressing the *Exrtract beads* button.

The data type of the resulting PSF will be *float*, irrespectively of the data type of the input image. The individual beads will be weighted on a per-voxel basis and added to the result. No normalization will be

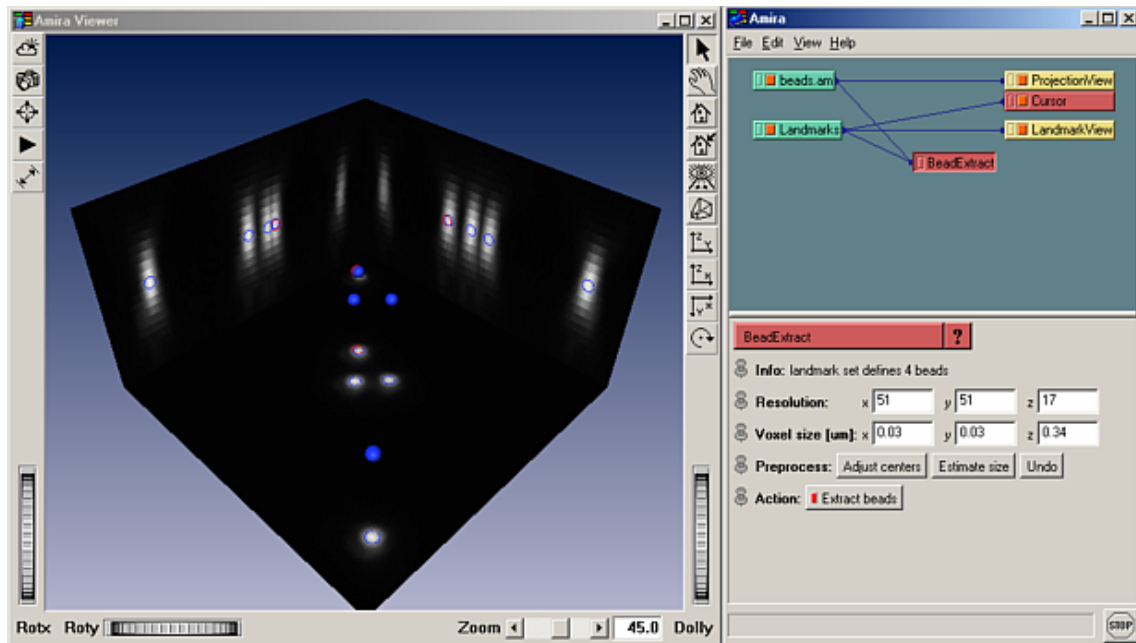


Figure 1.9: The *BeadExtract* module resamples and averages multiple beads.

performed afterwards. You may investigate the resulting PSF image using any of the standard visualization modules. In Figure 1.10 a volume rendering of the resulting PSF is shown.

In some cases you may want to average multiple beads recorded in different input data sets. This can be easily achieved by creating a *Landmarks* object for each input data set. For the first input data set extract the beads as described above. For the other input data set also use the *BeadExtract* module. However, make sure that the PSF obtained from the first input data set is connected as a result object to *BeadExtract* before pressing the *DoIt* button. In order to use an existing PSF as a result object connect the *Master* port of the PSF to *BeadExtract* (once this is done the *Resolution* and *Voxel size* ports of *BeadExtract* become insensitive, see above). If an existing result is used new beads simply will be added into the existing data set. Therefore datasets should be scaled in intensity according to their quality prior to bead extraction and summation to obtain a suitable weighting of the individual extracted beads in the final result.

1.6 Performance issues and multi-processing

Iterative maximum-likelihood deconvolution essentially is the most powerful and most robust technique for the restoration of 3D optical sections. However, it is also computationally very demanding. It can take several minutes (sometime even hours) to process large 3D data sets. This is not due an improper implementation but due to the algorithm itself. Both, the blind and the non-blind variant of the method, rely heavily on fast Fourier-transforms in order to efficiently compute convolutions. If you want to improve performance, try to adjust the size of your data volumes so that the number of voxels plus the border width is a power of two. Sometimes, it is worth to enlarge the border width a little bit in order to get a power of two. Although the algorithm works with data of any size, powers of two can be transformed somewhat faster.

Another issue is memory consumption. Internally, several copies of the data set need to be allocated by the deconvolution algorithm. These copies should all fit into memory at the same time (a specific variant of the algorithm suitable for working under low memory conditions will be provided in a later version). Besides

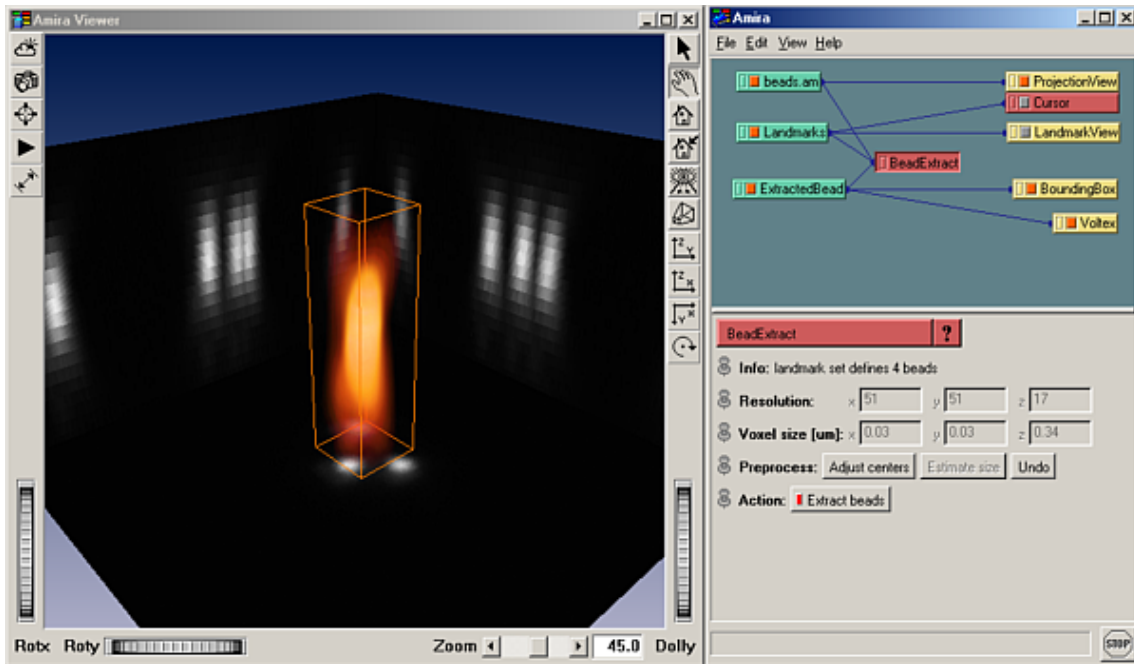


Figure 1.10: The final PSF visualized using a Vortex module.

the input data itself the following number of working arrays are required by the different methods:

- 3 working arrays for the non-blind algorithm with no or with fixed overrelaxation
- 5 working arrays for the non-blind algorithm with optimized overrelaxation
- 5 working arrays for the blind algorithm

The number of voxels of a working array is the product of the number of voxels of the input data set plus the border with along each statial dimension. The primitive data type of a working array is a 4-byte floating point number. For example, if the number of voxels of the input data set plus the border width is $256 \times 256 \times 256$ (as for the *alphalobe.am* data set in the blind deconvolution tutorial), each working array will be about 64 MB, irrespectively of the primitive data type of the input data set. Therefore at least 192 MB ($3 \times 4 \times 256 \times 256 \times 256$ bytes) are required for non-blind deconvolution with fixed overrelaxation, and 320 MB ($5 \times 4 \times 256 \times 256 \times 256$ bytes) for blind deconvolution. Keep this in mind when configuring the computer on which to perform deconvolution! However, also note that for most platforms it usually doesn't make sense to have more than 1.5 GB of main memory. For more memory a 64-bit operating system is required. On a Windows PC, it is strongly recommended not to use Windows 98 or Windows ME if you have more than 512 MB of main memory.

Finally, it should be mentioned that the deconvolution algorithm can make use of a multi-processor CPU board. Although you don't get twice the performance on a dual-processor PC, a speed-up of almost 1.5 can be achieved. On default, Amira uses as many processors as there are on the computer. If for some reason you want to use less processors you can set the enironment variable `AMIRA_DECONV_NUM_THREADS` to the number of processors you actually want to use simultaneously.

Chapter 2

Module Reference

2.1 BeadExtract

This module is part of the Amira deconvolution package. It can be used to resample and average the image of one or multiple beads, i.e., fluorescing sub-resolution microspheres, thereby obtaining an approximation of a point spread function (PSF) required for non-blind deconvolution.

The module must be connected to the image data set containing the measured beads as well as to a landmark set indicating the center positions of all beads to be resampled and averaged. The whole process of obtaining a point spread function from a bead measurement is described in a separate *tutorial on bead extraction*. Please refer to this tutorial for more information on how to use the module.

Connections

Data

[required]

Connection to a uniform image data set containing measured beads.

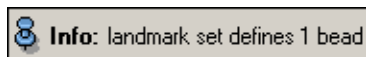
Landmarks

[required]

Connection to a landmark set indicating the center positions of the beads to be resampled and averaged.

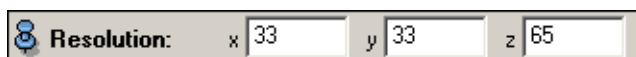
Ports

Info



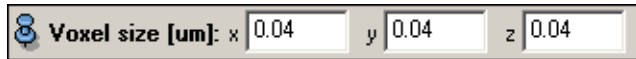
Displays the number of beads to be resampled and averaged.

Resolution



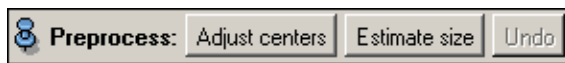
Specifies the number of voxels of the final PSF image to be generated. If a PSF image is connected as a result object to this module, the port becomes insensitive and the number of voxels of the result object are used.

Voxel size [um]



Specifies the voxel size of the final PSF image to be generated in micrometers. If a PSF image is connected as a result to this module, the port becomes insensitive and the voxel size of the result object is used.

Preprocess

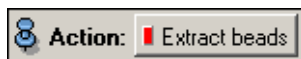


Adjust centers: If this button is pressed the landmark positions of the data set connected to port *Landmarks* are shifted to the center of gravity of the corresponding bead. It is required that the initial landmark positions are inside the bead images and that neighbouring beads do not overlap significantly. Otherwise incorrect center positions may be computed.

Estimate size: If this button is pressed the desired number of voxels of the final PSF image specified in port *Resolution* are computed automatically. Before this is done the beads' center positions already should have been adjusted. The estimate is computed by determining the extent of the biggest spot around any landmark. Again, it is required that neighbouring beads do not overlap significantly. The *Estimate size* button becomes insensitive if a result object is connected to the module. In this case always the size of the existing PSF image will be used.

Undo: Undos the effect of any of the previous two buttons. For example, if wrong center positions have been computed after pressing *Adjust centers*, the original landmark positions can be restored using the *Undo* button.

Action



Actually extracts the beads with the center of the resulting dataset corresponding to the current positions of the landmarks. The image volume around every landmark is extracted and resampled using a Lanczos filter (compare the *Resample* module). The resampled bead images are then added to the result object. The final PSF is not normalized.

2.2 Convolution

This module convolves two uniform 3D data objects with each other by Fourier transforming the two inputs, multiplying them, and then transforming them back. The module is part of the Amira *deconvolution package*. It can be used for example to verify the results of image deconvolution.

The bounding box and voxel sizes of the input data set and the convolution kernel are ignored by this module. Use the *Resample* module to make sure that the resolution of both inputs is identical.

Connections

Data

[required]

The data set to be convolved.

Kernel

[required]

The convolution kernel.

Ports

Border width

A port control for 'Border width' with three input fields: 'x' with value 12, 'y' with value 12, and 'z' with value 30.

Defines the size of the border region. A border region is necessary if the input data set doesn't fade out to black at the boundaries.

Options

A port control for 'Options' with two checked checkboxes: 'normalize kernel' and 'add noise'.

If *normalize kernel* is selected, the integral of the convolution kernel (connected to port *Kernel*) will be normalized to one. If this is not the case the intensities of the convolved data set will be scaled by the actual integral.

If *apply noise* is selected, the values of the convolved data set will be multiplied by random numbers uniformly distributed around 1 (white noise).

Noise level

A port control for 'Noise level' with a slider and a value field showing 0.

This port will only be shown if the *apply noise* option of the *Options* port has been selected. It specifies the amount of noise applied to the output, i.e., the range of the random numbers around 1, by which the result is multiplied.

Action

A port control for 'Action' with a button labeled 'Do it'.

Actually convolves the input data sets.

2.3 CorrectZDrop

This module lets you fix artifacts in 3D microscopic images caused by light absorption in other slices. If such artifacts are present, the average intensity in lower slices seems to be decreased. This so-called *z-drop* or *intensity attenuation* can be corrected automatically by fitting an exponential curve to the average intensities in each slices, or manually by providing a user-defined formula.

Connections

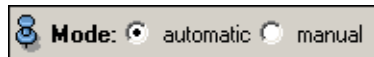
Data

[required]

The image data exhibiting a z-drop artifact. Scalar fields with uniform or stacked coordinates as well as *multi-channel fields* are supported.

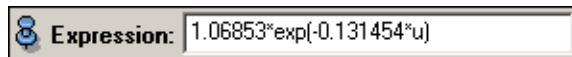
Ports

Mode



Lets you select between automatic mode and manual mode.

Expression



This port is only available if manual mode has been selected. It provides a text field where you can enter a formula specifying a factor used to multiply the intensity values in each slice. Within the formula the variable u specifies the slices. u will take the value 0 for the first slice and 1 for the last slice. For the other slices it takes intermediate values depending on the actual slice location (this makes support of stacked coordinates easy). In automatic mode the following formula $a * \exp(b * u)$ will be used, where a and b are fitted automatically. If you first perform an automatic z-drop correction and then switch to manual mode, the fitted exponential will be displayed in the port's text field.

Action



Starts z-drop correction.

2.4 DataPreprocess

This module is part of the Amira *deconvolution package*. It can be used to apply both a background and a flatfield correction to a raw 3D image stack.

For the background correction a single background image has to be provided at the *background* port of the module. The background image should be a nearly black image recorded with the camera's shutter closed. This image is subtracted from all slices of the 3D input data set, thus compensating for any dark current of the camera's CCD detector.

For the flatfield correction a single flatfield image has to be provided at the *flatfield* port of the module. The flatfield image should be an unfocused almost white image taken from a drop of homogeneously fluorescing dye. The intensities of the 3D input image are then scaled according to the normalized intensities of the flatfield images. The input image gets brighter at pixels where the flatfield is dark and vice versa. In this way non-uniform sensitivity of the camera's CCD detector is compensated. If both a flatfield and a background image are present, the background is subtracted from the flatfield too.

Connections

Data

[required]

The raw 3D image stack to be corrected. Any regular scalar field with uniform coordinates is supported.

Background

[optional]

An optional 2D background image with the same number of voxels in x and y direction as the 3D input image. If an input is present at this port a background correction is performed (see above).

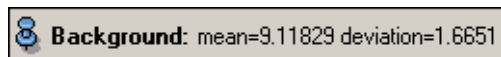
Flatfield

[optional]

An optional 2D flatfield image with the same number of voxels in x and y direction as the input image. If an input is present at this port a flatfield correction is performed (see above).

Ports

Background



Displays the mean value and the standard deviation of the background image, if such an image is present. Only the first slice is considered.

Flatfield



Displays the mean value and the standard deviation of the flatfield image, if such an image is present. Only the first slice is considered.

Action



Starts the computations and produces a corrected output data set.

2.5 Deconvolution

This module is the front-end for deconvolving 3D microscopic images. Two different iterative maximum-likelihood image restoration algorithms are provided, a non-blind one and a blind one. For a general description of the deconvolution process please refer to the *tutorials* provided with the Amira deconvolution package.

The resulting deconvolved data set will be stored in the object pool. If no input PSF is specified or if the blind deconvolution algorithm has been selected, also the estimated PSF will be stored in the object pool.

Connections

Data

[required]

The data set to be deconvolved.

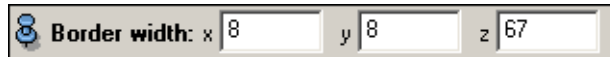
Kernel

[optional]

The point spread function (PSF) to used for deconvolution. If no PSF is specified, an estimated PSF is calculated automatically based on the numerical aperture of the microscope, the wavelength of the emitted light, and the refractive index. If a blind deconvolution is to be performed, an input PSF (if connected) will be used as an initial estimate.

Ports

Border width


 A control panel for border width. It features a blue icon on the left, followed by the text "Border width:". To the right are three input fields labeled "x", "y", and "z". The "x" field contains the value "8", the "y" field contains "8", and the "z" field contains "67".

Defines the size of the border region. A border region is necessary if the input data set doesn't fade out to black at the boundaries. For performance reasons it might advisable to choose values such that the sum of the size of the input data set and the border width results in a power of two. For example, if the data set consists of 118 slices, a border width of 10 slices in z direction would be a good choice.

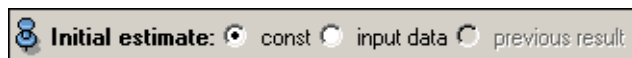
In case of widefield data it is sometimes advisable to have the border width in z direction exactly as large as the data set itself. If this is the case, the border region will be initialized by mirroring the values from the actual data volume. Otherwise, the values of the first slice and of the last slice will be interpolated linearly.

Iterations


 A control panel for iterations. It features a blue icon on the left, followed by the text "Iterations:". To the right is a slider control with a triangular handle in the middle. Further right is a numeric input field containing the value "60".

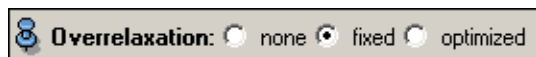
Specifies the number of iterations of the deconvolution procedure.

Initial estimate


 A control panel for the initial estimate. It features a blue icon on the left, followed by the text "Initial estimate:". To the right are three radio button options: "const", "input data", and "previous result". The "const" option is selected.

Specifies the initial estimate of the deconvolution algorithm. If *const* is chosen a constant image is used initially. Often, this yields smoother results than the second option, namely *input data*. The third option (*previous result*) is only available if the input data set has already been deconvolved previously. Use this option if you want to apply some additional deconvolution iterations.

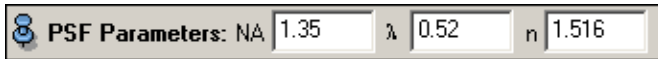
Overrelaxation


 A control panel for overrelaxation. It features a blue icon on the left, followed by the text "Overrelaxation:". To the right are three radio button options: "none", "fixed", and "optimized". The "fixed" option is selected.

Overrelaxation is a technique to speed up the convergence of the iterative deconvolution process. In most cases *fixed* overrelaxation is a good choice. For non-blind deconvolution also an *optimized* overrelaxation technique is available. This method further accelerates convergence but is more memory and time consuming.

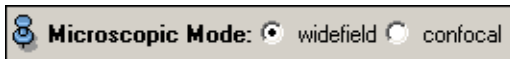
Method

This port specifies whether standard (non-blind) or blind deconvolution should be used.

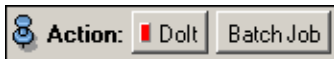
PSF Parameters

Parameters for calculating the point spread function. This port will only be shown if standard (non-blind) deconvolution has been selected and no input PSF was specified, or if blind deconvolution has been selected, in which case these parameters act as constraints.

NA denotes the numerical aperture of the microscope. *lambda* denotes the wavelength of the emitted light in micrometers with the voxel sizes also being interpreted in micrometers. Finally, *n* denotes the refractive index of the specimen.

Microscopic Mode

Selects whether the input image has been recorded using a widefield microscope or using a confocal microscope. This is important for the PSF generation as well as for the selection of appropriate constraints during blind deconvolution.

Action

The *DoIt* button actually starts the deconvolution process. Since deconvolution is a time consuming operation, it optionally can be performed as a batch job. A batch job can be submitted using the *Batch job* button. If this button is pressed, first a dialog is popped up allowing you to specify the filename of the final deconvolved data set as well as the number of optional check point files (compare Figure 1.6). A check point stores an intermediate result obtained after a certain number of iterations have been performed. The actual deconvolution job is started via the Amira job dialog. The job dialog is popped up automatically after the job has been finally submitted, but this may take a few seconds if there are currently no jobs running.

2.6 FourierTransform

This module compute a discrete forward or backward Fourier transform a scalar input data set with uniform coordinates. Alternatively, the power spectrum, i.e., the squared magnitude of the Fourier transform can be computed.

The origin of the input data set will be ignored by this module. Also, instead of being expressed in wave numbers the bounding box of a Fourier transformed data set will be the same as the bounding box of the input.

Connections

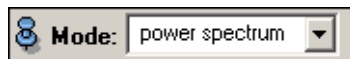
Data

[required]

The input data to be Fourier transformed.

Ports

Mode



Option menu specifying the action to be performed.

If a real scalar field is connected to the module, three options are available, namely *forward*, *forward complex*, and *power spectrum*. If *forward* is selected, the output is stored in a special so-called half-complex format. Such an output can be back-transformed, but not many other operations can be performed on it. If *forward complex* is selected, the output will be a complex-valued scalar field with the same number of voxels as the input object. Finally, if *power-spectrum* is selected, the output is a real-valued scalar field with the same number of voxels as the input object containing the squared magnitude of the Fourier transform.

If a complex scalar field in half-complex format is connected, the only option is *backward*, indicating a backward Fourier-transform.

If an ordinary complex scalar field is connected to the module, the three options *forward*, *backward*, and *power spectrum* are available. The first two options specify a forward and backward Fourier transform, respectively. The output is a complex scalar field with the same resolution as the input. If *power-spectrum* is selected, the output is a real-valued scalar field with the same number of voxels as the input object containing the squared magnitude of the Fourier transform.

Action



Starts the computation.

2.7 PSFGen

This module can be used to compute a point spread function (PSF) for deconvolution of widefield and confocal microscopic image data. PSF computation is based on electromagnetic vector theory. The module is part of the Amira *deconvolution package*. It can be created by selecting it from the *Edit Create* menu of the Amira main window.

Connections

Data

[optional]

A uniform scalar field (usually the image data to be deconvolved) can be connected to this port. The values of the *Resolution* and *Voxel size* ports will be set automatically to the values of the input field then.

Ports

Resolution

Resolution: x 33 y 33 z 65

The number of voxels of the PSF image to be generated.

Voxel size [um]

Voxel size [um]: x 0.04 y 0.04 z 0.04

The voxel size in microns of the PSF image to be generated.

PSF Parameters

PSF Parameters: NA 1.35 λ 0.52 n 1.516

Parameters for calculating the point spread function. *NA* denotes the numerical aperture of the microscope. *lambda* denotes the wavelength (as measured in vacuum) of the emitted light in micrometers. For confocal data the excitation and emission wavelength are assumed to be identical. In this case it might prove useful to compensate by supplying a value between excitation and emission wavelength as parameter. Finally, *n* denotes the refractive index of the specimen.

Microscopic Mode

Microscopic Mode: ☒ widefield ☐ confocal

Selects whether the PSF of a widefield microscope or of a confocal microscope should be computed.

Action

Action:

Actually computes the PSF.

Chapter 3

Examples

Example 1: Confocal Data

The original data set is provided under *Amira-2.3/data/deconv/polytrichum.am*. The images below were created using the *ProjectionView* module.

The properties of the data set are as follows:

- Numerical aperture 1.4
- Wavelength of the emitted light 0.58 micrometers
- Refractive index 1.516 (oil)

Example 2: Widefield Data

The original data set is provided under *Amira-2.3/data/deconv/alphalobe.am*. The images below were created using the *ProjectionView* module.

The properties of the data set are as follows:

- Numerical aperture 0.5
- Wavelength of the emitted light 0.58 micrometers
- Refractive index 1.33 (water)

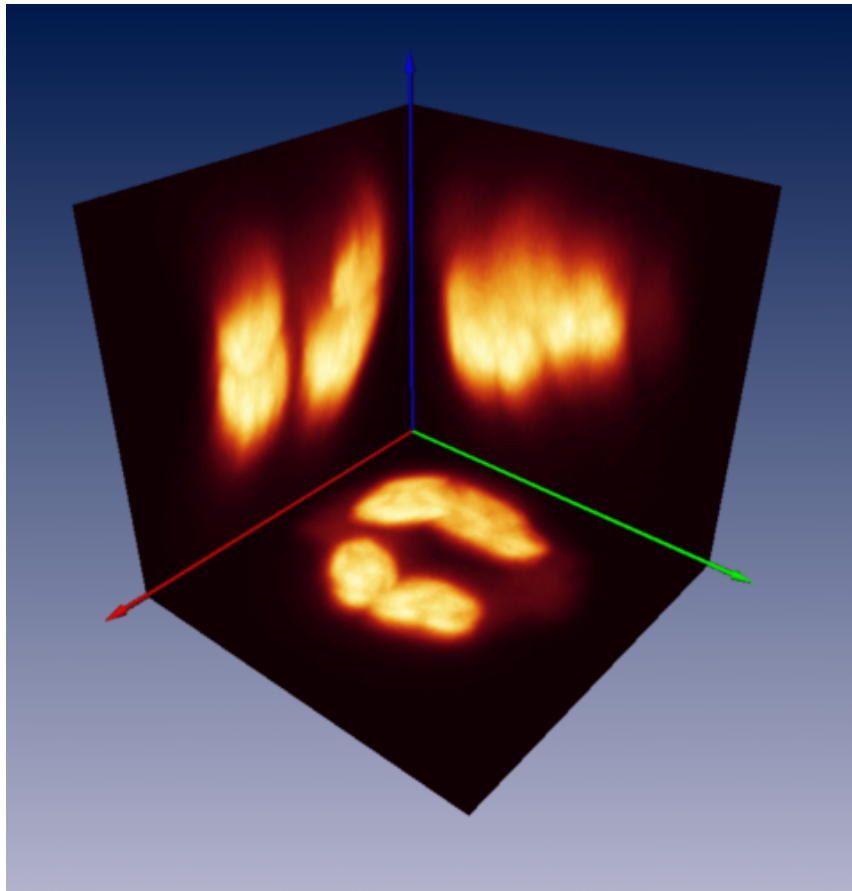


Figure 3.1: *polytrichum.am* before deconvolution.

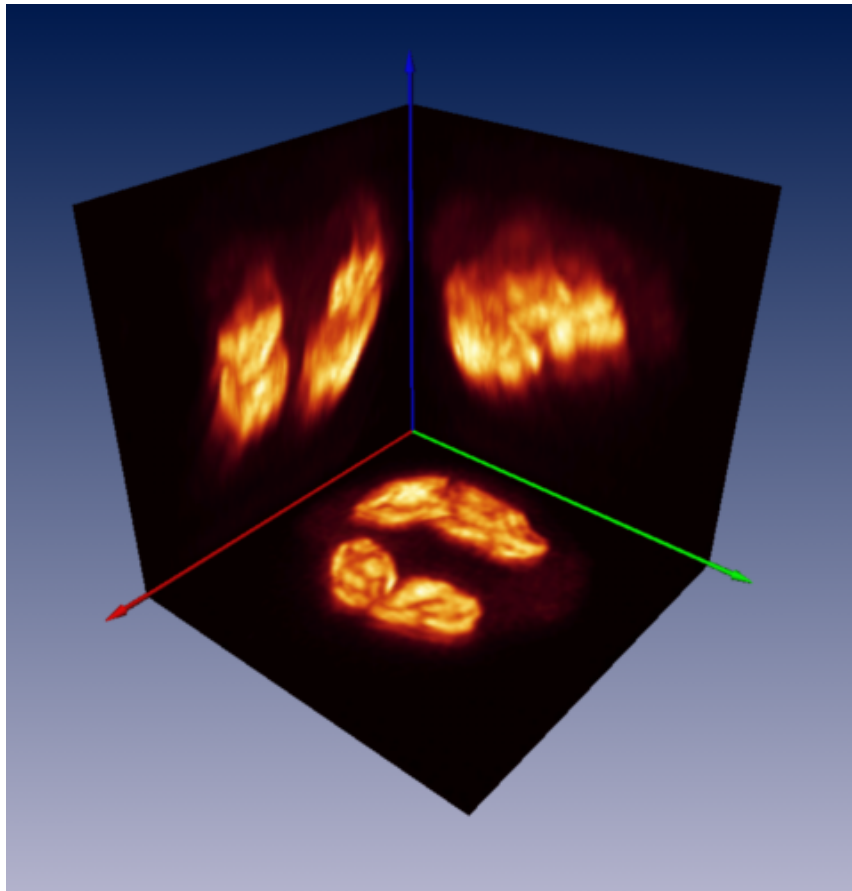


Figure 3.2: *polytrichum.am* after deconvolution.

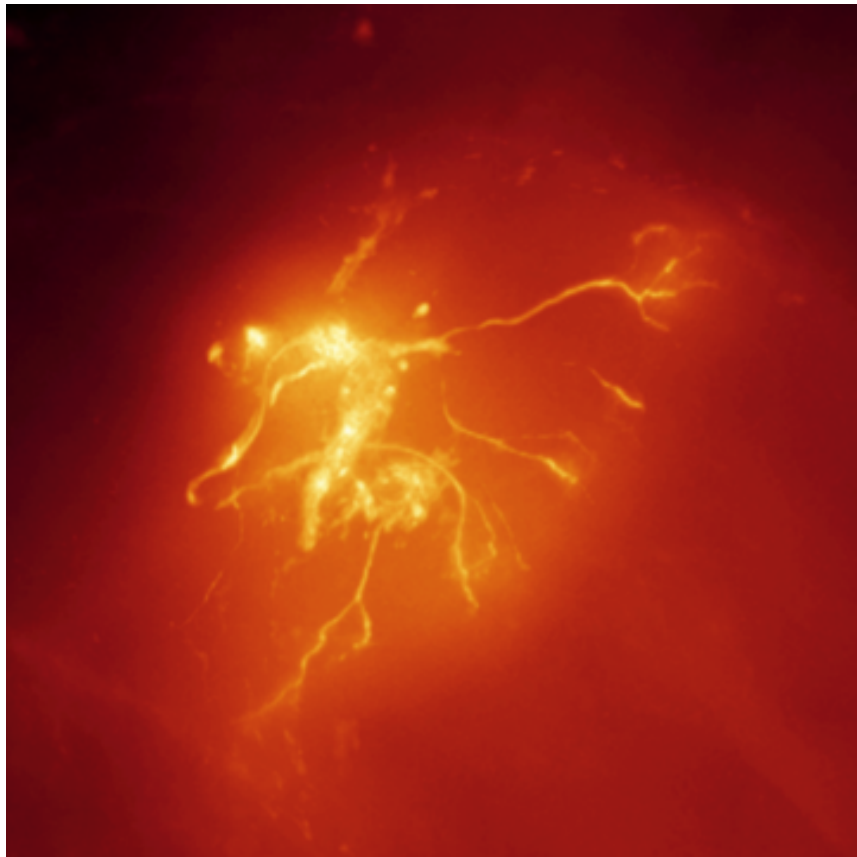


Figure 3.3: XY-maximum intensity projection of *alphasat* before deconvolution.

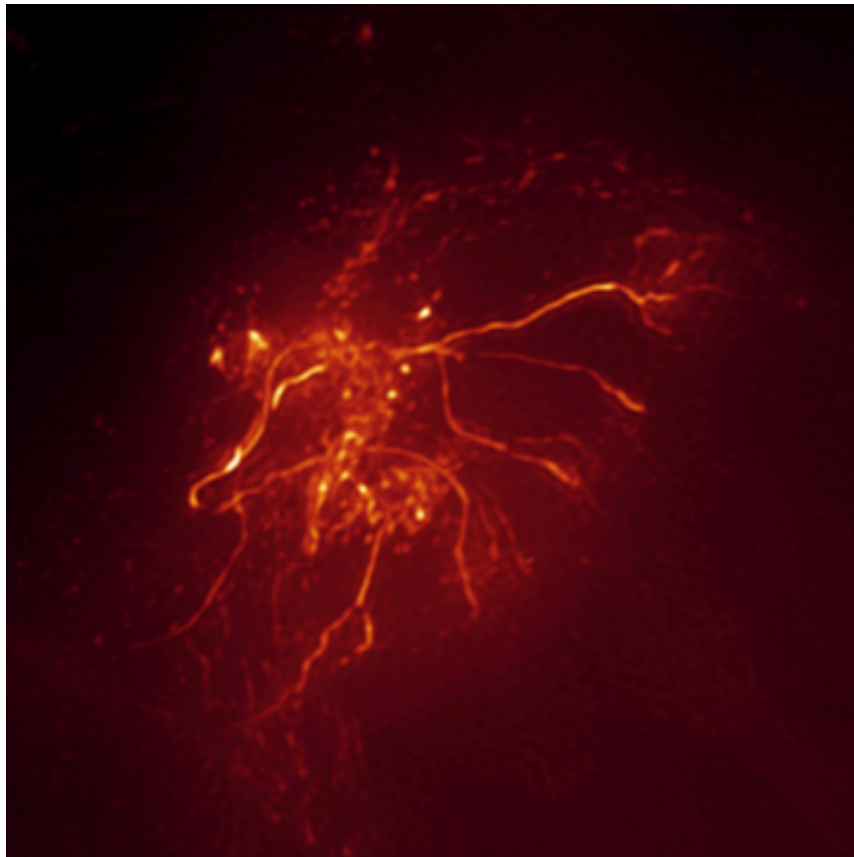


Figure 3.4: XY-maximum intensity projection of *alphalobe.am* after deconvolution.

Index

- abberation, 17
- agarose gel, 17
- axial blur, 5

- background correction, 24
- batch job, 15
- bead extraction, 16
- BeadExtract, 21
- beads, 16
- black level, 7
- border width, 9, 13, 26
- boundary artifacts, 6

- CCD detector, 24
- check point files, 15
- confocal microscope, 6, 7
- Convolution, 22
- CorrectZDrop, 23
- coverslip, 17

- DataPreprocess, 24
- Deconvolution, 25
- deconvolution
 - blind, 6, 13
 - non-blind, 6
 - standard, 7

- embedding medium, 17

- flatfield correction, 24
- Fourier transform, 19
- FourierTransform, 27

- immersion medium, 17
- in-plane sampling, 7
- initial estimate, 11, 13, 26
- intensity attenuation, 8, 23

- job dialog, 15, 27

- Lanczos filter, 8
- landmark set, 21

- maximum-likelihood method, 6, 25
- memory consumption, 20

- microsphere, 16, 21
- multi-processing, 20

- noise, 6, 7, 23
- numerical aperture, 7, 26
- Nyquist sampling, 7

- oil immersion, 17
- optical sectioning microscopy, 5
- out-of-focus light, 5, 13
- overrelaxation, 11, 13, 26
- oversampling, 7

- performance, 19
- power spectrum, 27
- PSF, 6, 8, 21
 - theoretical, 12
- PSFGen, 28

- reampling, 8
- refractive index, 17
- refractive index, 7, 26

- sampling rate, 7
- saturation, 7
- scanned volume, 6

- undersampling, 7

- vector theory, 28

- wavelength, 7, 26
 - emission, 29
 - excitation, 29
- widefield data, 6

- z-drop, 23