IMage Analysis Tool (IMAT)

xprepro, xpluto, xstructbrowse, xfiles

Jakob L. Laugesen
Susanne M. B. Johansen
Michael B. Frøst

Department of Food Science, Sensory group
The Royal Veterinary and Agricultural University, Denmark

1. December 2004
(doc generated March 16, 2005)

1 Introduction

The program Global Image Analysis (IMAT) works on 1- and 2D images. A global feature of an image is defined as a feature that are independent of the spatial positions of pixel values. A global feature may, however, be dependent on the distances (NOT positions). It is the intention that the software must balance the concept of userfriendly GUIs without being too strict. This is an important problem, since applications of image analysis is very difficult (read impossible) to generalize, while on the other hand many operations are often the same.

The software is optimized for the following items that will be discussed throughout this text.

- Often many images of the same sort (e.g. all images are images of bacteria) are the subject of analysis. Hence, analysis is performed as a looping over images placed in a folder.
- The interesting features in the images are very dependent on the research subject. Hence, a subject/problem specific preprocessing of the images is required. This is most effectively done by including m-file scripts written by the user.
- Any type of global analysis method may be implemented as long as a set of rules for input arguments and output arguments are fulfilled.
- In some cases some post-processing of the output is needed. This is implemented in a similar way as the pre-processing facility.
- Some methods requires a gridsampling. Hence, this is an option.

- Also threshold dependent gridsampling may optionally be an input for the global analysis algorithm.

A typical working process, is to enhance the features in the images by using some preprocessing, followed by the global analysis of all images and finally plotting the resulting outputs. The imat software package may take care of all three parts. The global analysis is taken care of by the `imat` program while pre-processing is handled by user specified m-files and `xprepro` and plotting by `xpluto`, which are plugins to imat. Since the imat setup and its output of the analysis is ordered in structured variables (imati and imato) a structure browser, `xstructbrowse`, may be used for easy working with the GUIs. All programs may be used separately or directly from `imat` and will be described in the appropriate chapters.

1.1 Installation and running

The program must be run under Matlab 6.5 in order to run properly[1]. It has been prepared such that it may be compiled and executed as a stand-alone executable under both windows and unix/linux, however this has not been tested and is not to be recommended.

Installation is done by following the steps:

1. Create a new folder (call it whatever you like) under `C:\MATLABxpx\work` and copy the file `imat-dd-mm-yy.zip` to the new folder and unzip it.
2. Enter the folder named imat-dd-mm-yy and open `imat_instpath.m` and supply the full path to the folder `imat-dd-mm-yy`.
3. In the MATLAB Toolbar go to File/Set Path and select the button Add with subfolders and choose the imat-dd-mm-yy folder in question. Eventually press the Move to bottom button.

Now the program is ready to run from any folder of your computer. Type `imat` in the matlab comand window to run the program.

2 Preparing the imat setup window

The items of the main imat window showed in figure 2 must be filled properly in order to run the program successfully. In this chapter mainly the first section of the window is explained, while the remaining is the major subject for the rest of this user guide. In the first field the full path to the folder with the images is entered or chosen by pushing the Chose folder/image. If the field *Single image* is on then a specific image must be chosen. In the field *Image type* filtering on image types is specified. For images with multiple channels (color images) a *channel* may be chosen, if you are running an analysis method that

[1]It has been tested under Matlab 7.0 for most of the functionality

makes use of for instance channel 1 and 3 you must enter 13 in the field[2]. You may also *resize* the images by a factor. The *Output index* is used for saving of the results. The output is saved in the same folder as the images in a filename with the syntax `imat_<your specified index>.mat`. Also, all information on the setup is stored in this file.

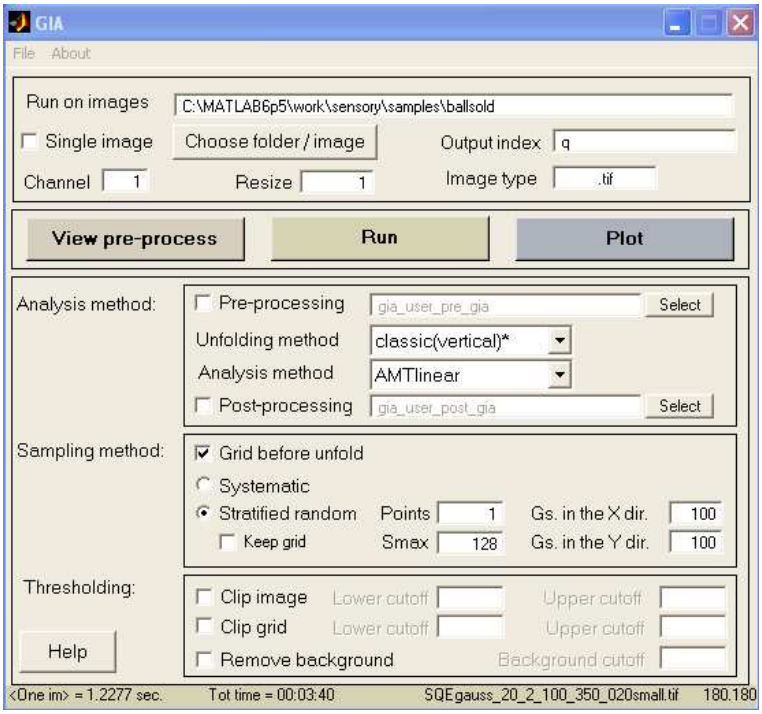


Figure 1: The setup window of imat.

Important: If there is no analysis method chosen and if the field pre-processing is on then a pre-processing of all the images is performed and is **saved with new filenames** having the syntax `<picture filenames w.o. extension>_<your specified index>.<the specified image type>`. For this reason it is not possible to run an analysis without having specified an index. If a new filename already exist it will be overwitten!

imat Setups may be saved and loaded, using the *File/Save* and *Load*.

3 Global features of images

Before presenting the analysis methods included in the software at hand, the reader and user should take the time to understand what information is. Most of the methods included concerns global information. To distinguish between local and global information figure 2 may be helpful. The difference between figures 2a and 2c are (almost) completely of pure local nature, since one of the

[2]Currently, none of the methods implemented make use of more than one channel.

objects are changes, while all other objects are unchanged both in size, shape and position.

Contrary, figures 2a and 2d is an example of a global difference, since the distribution of the objects are different. Strictly, the difference is also local, since the images are distributions of objects. Hence, a generically description would show that practically the everything in the two images are of local nature. Obviously, this can not be very interesting[3].

Information embedded in images, may be defined mathematically by calculating the geometrical relation between pixels in the space. This holds both for local and global information. The point at which the two kind of informations differs lies in wether the releations between the points also includes the actual positions of the points, i.e. it is local if not only relation between points are stored but also the positions of the points are stored. Figure 3 describes the principle of global information extraction. The point labelled 1 is an initial chosen point. This point alone only suplies one a local information, namely its position. By choosing additionally one point (point 2), points can be related to each other, by their distance s_{12} and the projected distances Δx_{12} and Δy_{12}. These three variables are global informations. However, they are not independent since $s_{12} = \sqrt{(\Delta x_{12})^2 + (\Delta y_{12})^2}$, hence, only two of them are relevant. Similar arguments can be made for a third point etc. Clearly, the more points that are related to each other, the more information on the nature of the image is gained.

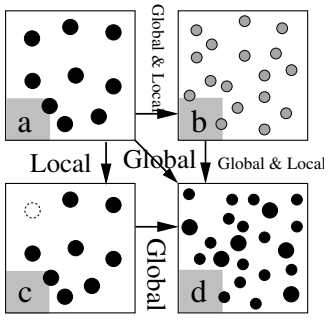


Figure 2: Relations between local and global information or characteristics.

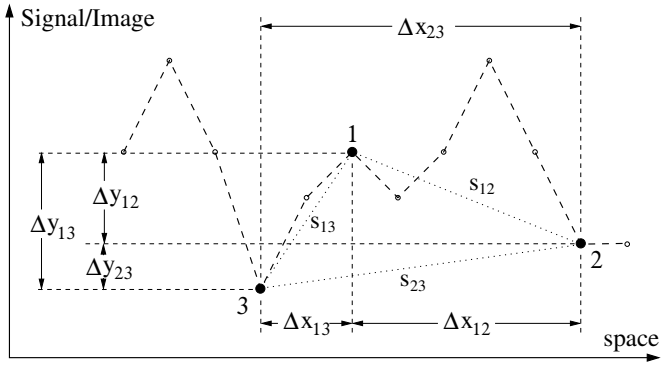


Figure 3: A mathematical piece of information.

4 Pre-processing and unfolding of images

4.1 User specified pre-processing

Often images must be pre-processed in order for the global features of the images to be more visible or enhanced. Since this topic depends very much

[3]Extreme analogy: It will never be interesting to describe the difference between a tiger and a cop of tea, since this is completely obvious and seems to be nonsens.

on the images, it is left to the user to supply the proper matlab commands in the editable file `imat_prepro.m`. The file may be edited by pressing the button View pre-processing whereby a new window appear and then press Opn m-file. The header of the m-file looks like

It is supplied by three arguments

X	The image
background	The background of the image

The output arguments of the m-file is

X	The image
background	The background of the image
N	The number of rows of X
M	The number of columns of X

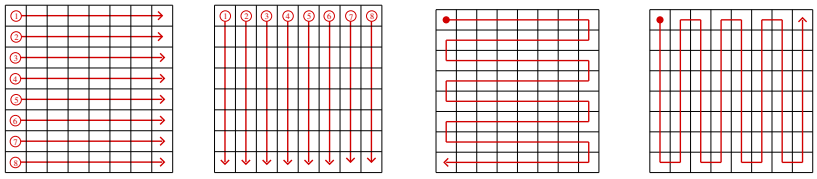
In the folder `xfiles` that came with the package one might find some usefull pre-processing algorithms. Recommended is also the DIPUM package by Gonzales *et al.* [1]. Note: That the effect of your supplied pre-processing can be explored by the `xprepro` window, see also the chapter on this facility.

4.2 Unfolding from 2D to 1D

Many methods for global feature extraction works on the unfolded 1D-image. The process of unfolding is in principle part of the preprocessing. However, there is no loss of generality to predefine the unfolding methods and include them as options in the GUI. Four types of unfolding is implemented, which are classic, snake, tilt and spiral. Except from the spiral method, they exist in a horizontal and vertical version. The principles of the methods is illustrated in fig.4.2.

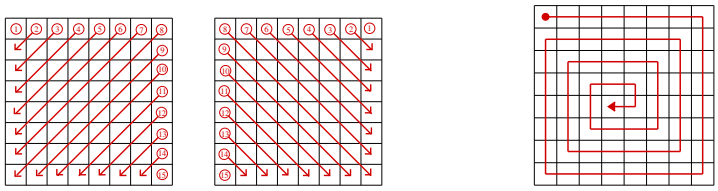
Regardless of method the $M\alpha$ spectra's always have a spike at some special values of the scale s. For an image of the size N and the case where $s = M$ and unfolding path is classic, the points A, B, and C will always be neighbors, see figure 4.2. Unless the pixel value variation within the image is close to random, these points probably have similar colours and hence, the mean angles are ($M\alpha$) close to zero. For this reason the classical unfolding path show a spike downwards at $s = M$. In case of unfolding by using the snake path, the situation becomes different. Here only B and C are close to each other in the 2D image. In the scale-colour representation, this situation is sketched in the lower left-hand side of figure 4.2. The colour of point A is independent of B and C and may thus take any value, i.e. A could be anywhere on the dashed vertical line. As the figure indicate the $M\alpha$ therefore is between zero and 2α due to the fact that $C \approx B$, and hence the spike will be up-wards for this case.

For the special distance $s = M/2$, the situation is nearly opposite of that of the case $s = M$. With the classical unfolding the situation becomes similar



(a) Classic. Horizontal and vertical

(b) Snake. Horizontal and vertical



(c) Tilted. For square images this $is 45^0$ and 135^0

(d) Spiral. Inside out.

Figure 4: Unfolding methods.

(except from one pixel-distance) to that of case $S = M$ with snake as unfold path. Thus, the spike will be upward at $s = M/2$.

It is important to note that the sketched situation in the lower left-hand side, points B and C, might as well be aligned with point A. But the figure illustrates that the angle has the possibility of being larger than the angle in the lower right-hand side of the figure, where it is point A and B that are similar and span the interval $[-\alpha_{max}; \alpha_{max}]$. For case $s = M$ (snake) C and B are similar, and therefore the angle span the broader interval $[-2\alpha_{max}; 2\alpha_{max}]$, that makes the important difference. α_{max} is defined in the figure.

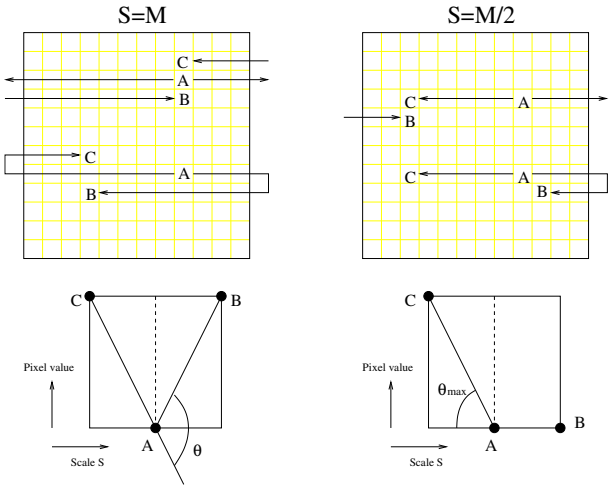


Figure 5: Illustration of the special situations occurring for the classic (upper part) and snake (lower part) unfolding methods.

5 Global information extraction algorithms

In this chapter the algorithms currently implemented in imat is presented and shortly explained. Table 5 gives an overview of possible setting. The label *must* denotes that for the method in question must be associated with the corresponding parameters. *not* means that it is not possible to run the method with the corresponding parameter. The '+' means that you have the choice to use it, but it is not demanded.

Method	Unfolding	pre-pro.	post-pro.	sampling	Clip im.	Clip gr.	Rm.bg
None	must	+	+	must	+	+	+
AMT linear	must	+	+	must	+	+	+
AMT circle	must	+	+	must	+	+	+
AMT circle2	must	+	+	must	+	+	+
AMT 2D	not	+	+	must	not	+	not
Histogram	not	+	+	not	not	not	not
Star volume	not	+	+	must	not	must	not
Geometer	not	must	+	not	not	not	not
FFT2	not	+	+	not	not	not	not
Polar FFT2	not	+	+	not	not	not	not
SP mapping	not	must	+	not	not	not	not
Frac.dim. 2D	not	must	+	not	not	not	not
Frac.dim. 3D	not	must	+	not	not	not	not

Table 1: Overview of possible parameter setting for the analysis methods.

5.1 None

When no method is selected, the images is being pre-processed and saved in new image filenames with names generated as <original file name>_<Output index>.tif.

5.2 AMT circle

The AMT circle method is the algorithm originally proposed by R. Andrle [2]. The abscissa of the circle method is a combination spatial distance and pixel value; actually the color is interpreted as an metric, such that the image forms a 3D metric space.

For a set Ω of (usually random) point the method is summarized by

$$M\alpha(s) = \frac{1}{|\Omega|}\sum^{A\in\Omega}\pi - \angle CAB \quad (1)$$

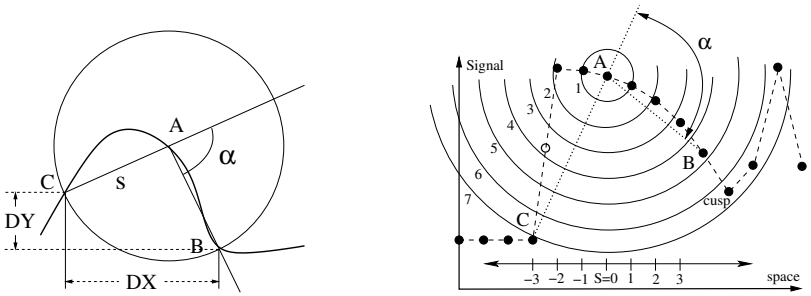
$$= \frac{1}{|\Omega|}\sum^{A\in\Omega}\pi - \arccos\left(\frac{DX^2+DY^2-2s^2}{2s^2}\right) \quad (2)$$

where s is the radius of the circle, $DX = B_x(s) - C_x(s)$ and $DY(s) = B_y(s) - C_y(s)$, where $B = \min(|B_i' - A| - s)$ and $C = \min(|C_i' - A| - s)$. The index i is introduced in order to specify that it is the first incident that fulfills the

condition. Often the averaging over DX and DY is used together with the $M\alpha$, and is given by

$$MDX(s) = \frac{1}{|\Omega|} \sum^{A \in \Omega} |DX| \tag{3}$$

$$MDY(s) = \frac{1}{|\Omega|} \sum^{A \in \Omega} |DY| \tag{4}$$



(a) Analogue (b) Discrete

Figure 6: The AMT circle method.

Although Andrle developed this method to be used on geomorphic curve the present formulation of the method is not able to handle this kind of curves. The software is restricted to analyse sequential signals and images only.

5.3 AMT circle with correction (PWL)

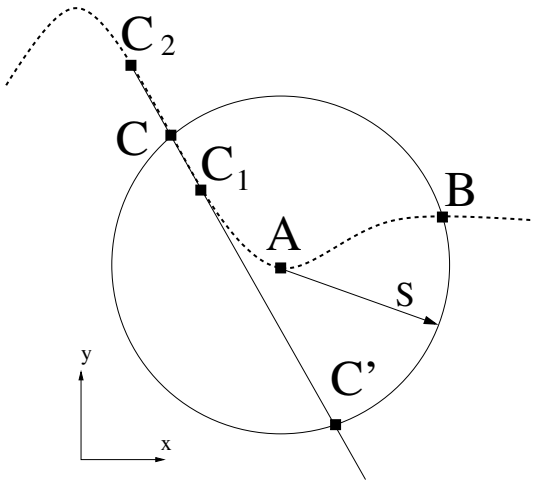
If the image or signal physically is a discrete non-continuous set of points then it is most correct to use the plain AMT circle method described previously. On the other hand images are often pictures of objects, which physically are continuous. In such cases it is the digitalization of images that causes the discrete and discontinuous pixelvalues. From this point of view it is most correct to use the intersection of the line connecting an interior point with an exterior point with the circle. With the geometry and definitions shown in figure 5.3 the relevant intersections are

$$C_x = \frac{-2(\alpha c_{2,y} - \alpha^2 c_{2,x} - \alpha a_y - a_x) \pm \sqrt{D}}{2(\alpha^2 + 1)} \tag{5}$$

and

$$C_y = \alpha C_x + c_{2,y} - \alpha c_{2,x} \tag{6}$$

and similar for the B-point.



5.4 AMT linear

The abscissa of the linear method is identical to the physical space of the image, i.e. it has dimension 1 after unfolding[4].

$$M\alpha(s) = \frac{1}{|\Omega|}\sum^{A\in\Omega} \pi - \arccos\left(\frac{2DY_{AC}DY_{AB} + 2s^2}{2\sqrt{(s^2 + DY_{AC}^2)(s^2 + DY_{AB}^2)}}\right) \tag{7}$$

$$MDX(s) = \frac{1}{|\Omega|}\sum^{A\in\Omega} |DX| \tag{8}$$

$$MDY(s) = \frac{1}{|\Omega|}\sum^{A\in\Omega} |DY| \tag{9}$$

where $DX = B_x(s) - C_x(s)$ and $DY(s) = B_y(s) - C_y(s)$. DY_{AB} and DY_{AC} are projected distances from A. Note, that contrary to the circle method DX, DY and hence α are perfectly determined since the scaling parameter s is coinciding with the index of the image. A consequence of this is that the points B and C are determined at the moment of fixing s and hence this method is much faster that the circle method[5].

Compared to the circle method, the abscissa of the present method is independent of the color axis. This allow us to relate the spectra linearly to the actual physical space of the image, which is preferable in many cases, such as studies of particles (size and distribution). However, one must be very carefull not to overinterpret the spectra.

Note on scaling: In some cases it may be efficient to scale the 8 bit image to n bit. See Johansen *et al.*[3] for further details.

[4]The discrete space is a metric but *not* an intrinsic metric nor a Hilbert space.

[5]Performance for the circle method depends on image structure. For typical images the linear method performs by approx. factor 10 better.

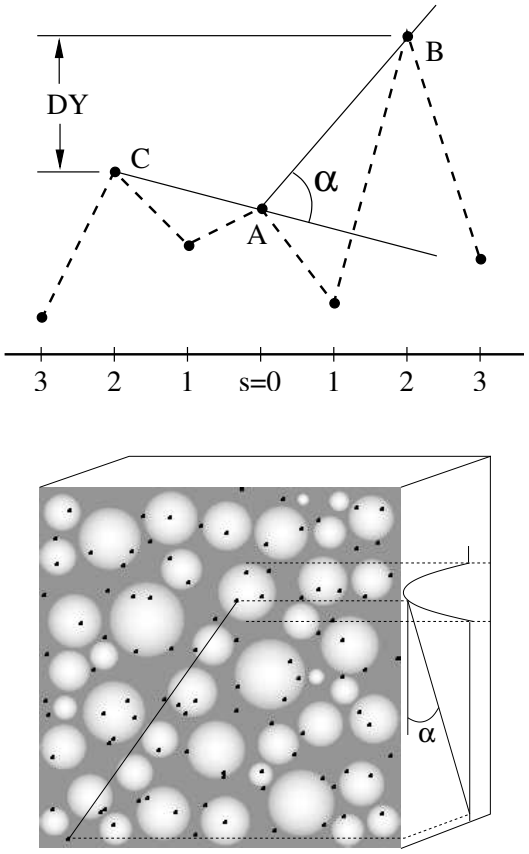


Figure 7: Principle of the two-dimensional AMT method. Note this formulation only includes two points and is therefore not a direct generalization of the one-dimensional AMT linear algorithm.

5.5 AMT linear 2d

This AMT algorithm quite similar to a 2d version of the linear AMT method and it works directly on the unfolded image, thus, unfolding method should be set to *None*. For performance reasons the angle calculation only includes two points, both members of the gridsampling set. Figure 5.5 show an image of homogeneously distributed spheres with a random grid Ω set superposed onto the image (black points).

$$M\alpha(s) = \frac{1}{|\Omega|} \sum^{A\in\Omega} \sum^{B\in\Omega} \left| \arctan\left(\frac{\Delta y}{s}\right) \right| \tag{10}$$

and

$$MDY(s) = \frac{1}{|\Omega|} \sum^{A\in\Omega} \sum^{B\in\Omega} |\Delta y| \tag{11}$$

where $s = \sqrt{(b_1 - a_1)^2 + (b_2 - a_2)^2}$ and $\Delta y = |I(b_1, b_2) - I(a_1, a_2)|$.

5.6 Histogram

This method is the most simple way of extracting global information. Despite its simplicity and can be quite effective, depending on the subject of interest of course. It is simply the intensity of pixel values written as

$$N(c) = \sum^{i \in I} (I = c) \tag{12}$$

where c is the pixel value (for 8-bit $c \in [0, 2^8 - 1]$) and I is the image matrix. In imat this method simply makes use of the matlab function *imhist*.

5.7 Geometer

Measurement of area and perimter is here called geometer. It usually requires a lot of pre-processing since the objects must preferable appear as segmented eventually binarised in the image. Although this method do not require grid-sampling, thresholds must be defined in the fields related to the *Cut grid*[6]. Useful shape factors:

Form factor (FF)	$4\pi A/P^2$	$]0;1]$
Roundness (R)	$4A/(\pi L^2)$	$]0;1]$

The form factor (FF) is a measure of the degree of form and shape complexity. The roundness (R) is only well defined measure for uniaxial conformal deformations of a circle (ex. ovals). In the limit of the circle R approaches 1 and in the limit of a finite line it approach 0. In practical analysis zero will not be approached due to the intrinsic width of a digitized line.

The output of geometer is only the area and perimeter of all the objects in the image on the form `resy=[area,border]`. The calculation of form factor and roundness must be perfomed in the post-processing file. In the xfiles folder there is a function (xformfactor) for calculating the form factors.

5.8 Discrete Fourier transform (DFT). Fourier Pw. Spectrum.

The 2-dimensional discrete Fourier transform (DFT) of a region with bounds $0 \leq m \leq M-1$ and $0 \leq n \leq N-1$ is defined by

$$F(p,q) = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} f(m,n) e^{-i2\pi pm/M} e^{-i2\pi qn/N} \tag{13}$$

where $q = 0, 1, .., M-1$ and $q = 0, 1, ..N-1$. The one-dimensional version of DFT is found simply by putting $q = 0$. The amount of energy or power in the oscillating signal is defined by

$$P(p,q) = |F(p,q)| \tag{14}$$

which is called the power spectrum.

[6]This ensures that the image is maintained

The Fourier transform is as far as global information, i.e. position independent, features the best possible analysis type. The reason for this is that the Fourier transform maps all possible information onto the frequency space, without loss of information. It is easy to convince oneself that this is true by remembering that one may perform the inverse Fourier transform, whereby the original image is fully retained. If any information could have been lost, then the inverse transform would not yield the exact same image. Unfortunately, the Fourier transform of an image is twice as large as the image itself (twice, since the DFT yields a real and an imaginary part). This may easily cause storage problems and post-analysis (f.ex. multivariate data-analysis) performance problems. Hence, there is a need to filter the Fourier transformed images, only maintaining the part of information that is relevant. This may be very difficult, since it depends on the global features, that usually is the purpose for investigation and therefore unknown.

The Fourier transform implemented in imat, only maintains the power spectrum[7] of the image and unfolds the Fourier transform by the classic vertical method, such that the output is in the format $[log(|DFT2(I)|)] : P \times NM$, where I is the image matrix, P is the number of pictures and N, M is the dimensions of the images. This is a very large vector of which normally only a small part is interesting. The vector may be subject to a filter that only maintains the part of the vector where discrimination between samples is present. A filter of this type is implemented in the plotting program `xpluto`.

Note, that the use of this method easily causes memory problems due to the large output vectors. Hence, it is necessary to run the analysis on a subset of images, and use this preliminary analysis the define the filter. The filter should then be implemented in the post-processing m-file. Also, resizing of the images should be considered.

5.9 Polar Discrete Fourier power spectrum (PDFT2)

For isotropic images there is one simple filtering method that often should work. Since, power spectrum (absolute square of the DFT(X)) DFT's of isotropic images always is spherical symmetric around the zero-frequency, the DFT(X) may be reduced by taking the "circular sum" and "angular sum", defined by

$$C(r) = \sum_{\theta=0}^{2\pi} F(ceil(N/2 + r\cos(theta)), ceil(N/2 + r\sin(theta))) \tag{15}$$

$$A(\theta) = \sum_{r=0}^{N/2} F(ceil(N/2 + r\cos(theta)), ceil(N/2 + r\sin(theta))) \tag{16}$$

where $F = log(|DFT(X)|)$, N the dimension of the square image and $ceil(x)$ is a rounding function, that rounds to the nearest upper integer.

This sort of reduction of the 2-dimensional Fourier transforms is called "Average Polar Fourier transform"[8], since it averages the polar coordinates, radius

[7] If amplitude and phase is wanted, then you must implement a new method.

[8] Named by the author. As far as I know, it does not have a name.

and phase, with the zero-frequency as the origin of the polar coordinate system.

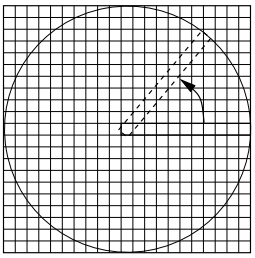


Figure 8: Circular sum

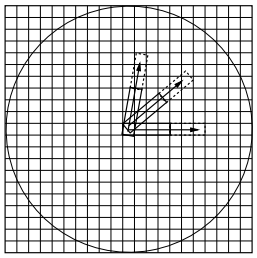


Figure 9: Angular sum

In case the method does not give successful discrimination then one needs to find an other filtering method. That is, what it is all about. The filter functionality in xpluto may be help full in determining a suitable filtering method.

5.10 Geodesic distances

Geodesic distances detection is only possible if the images consist of a particulated items and preferable non-elongated particles, such as stones, bacteria or particles of other kinds.

The particles must be isolated by defining thresholds in the `imat_user_pre.m` file, like `userpar`= [`lower upper`]. The algorithm determines the centers of the particles and calculates all Euclidean distances between the particles and store them in the output vector for each image. Eventually, a histogram of the distances may be calculated in the `imat_user_post.m` file, like [`resy,resx`] =`hist(resy,N)`, where N is the number of bins.

5.11 Fractal dimension (boxcounting)

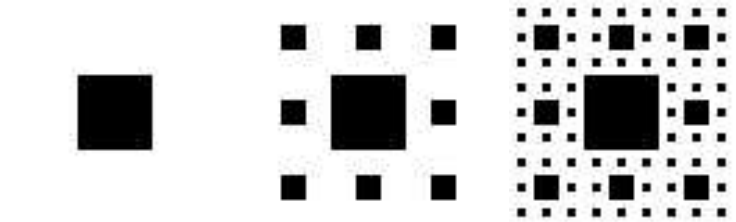
The boxcounting dimension is the fractal dimension, that is most easy to relate to geometrical objects existing in reality. It is defined as

$$D_0 = \lim_{\epsilon \to 0} \frac{log(N(\epsilon))}{log(1/\epsilon)} \tag{17}$$

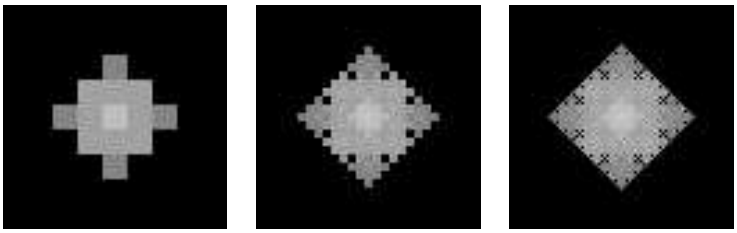
where ϵ is the length of the box sides and $N(\epsilon)$ is the number of boxes that are covered by the object.

Figure 5.11 show two kinds of computer generated fractals, known as Sierpinski's carpet and Koch's square. Matlab function for generation of variants of these fractals may be found in the `xfiles` directory. For these simple fractals, the dimensions can be evaluated analytical. For the Sierpinski's carpet one obtains

$$\begin{aligned} D_0 &= \lim_{n \to \infty} \frac{\log(8^n)}{\log(1/3^{-n})} && (18) \\ &= \frac{\log(8)}{\log(3)} && (19) \\ &= 1.893 && (20) \end{aligned}$$



a) Porous surface (Sierpinski's carpet)



b) Particle with fractal surface (Koch square in 3D)

Figure 10: Computer generated fractals.

where $\epsilon = 1/3^n$.

The Koch square is a little more tricky, since one has to decide whether to look at the object as a 3D object or as a flat object (projected). For the 3D case the surface of the particle has the dimension

$$D_0 = \lim_{n\to\infty} \frac{\log(6 \cdot 13^n)}{\log(1/3^{-n})} \tag{21}$$

$$= \frac{\log(13)}{\log(3)} \tag{22}$$

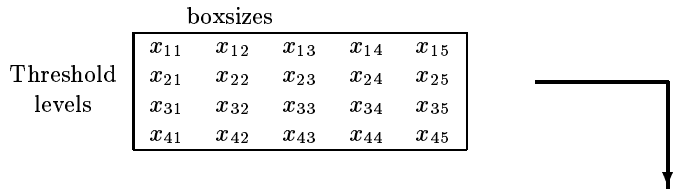
$$= 2.335 \tag{23}$$

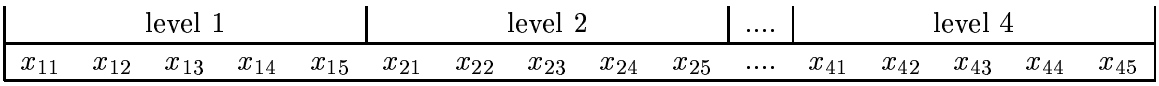
where as before $\epsilon = 1/3^n$. The image itself does not contain all the information of the 3D image. That is because the image is only 2D and the 3D image in only present in our brains, due to our knowledge of mathematics (self similarity in all dimensions) behind the object.

The parameters that the algorithm needs are not accessible from the main screen of imat. They must be defined in the pre-processing file and stored in the `userpar` variable as a structured array. The box sizes must be given as a vector and stored in `userpar.boxsizes`. In order to distinguish the object from the pores or whatever it is, a cutoff must also be defined and stored in `userpar.levels`. Example:

```
userpar.boxsizes=[20:-1:1];
userpar.levels=[100:20:250];
```

When several threshold levels are specified the format of the output is





Binarised objects like the Sierpinski's carpet are easily handled by the above definitions. Objects like the 3D Koch square, on the other hand, is less obvious to handle due to the third dimension, gray or color scale. To be able to handle this the algorithm must be able to count according to the gray scale of the pixels in the boxes. Currently, the 3D-algorithm is under devlopment...

5.12 How to implement a new method

It is possible to implement new methods, if the method requires nothing more than the following input arguments:

`X`	The eventually pre-processed image in either matrix form or unfolded vector form.
`background`	The background, by default the background is a zero $N \times M$ matrix.
`userpar`	A variable in which you may store parameters for later use.
`imati`	Structure variable containing all information from the setup window (grid, thresholding etc.).
`Randv`	If unfolding if different from *None*, `Randv` is a vector containing the grid points (otherwise it is empty).
`Randx` and `Randy`	If unfolding is **equal to None**, then `Randx` and `Randy` contains the indices to the grid points (otherwise they are empty).

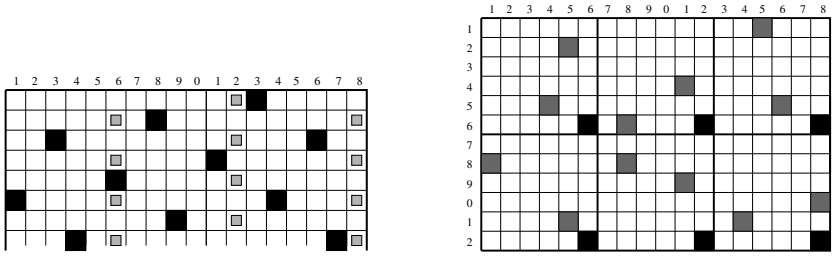
The output should be one vector. It may have variable length, but since all results is accumulated in a matrix where the rows are the result vector for each image, the columns of the matrix must be adjusted according the all result vectors. This is done by expanding the result vector or accumulation matrix by NaN[9].

[9]If you do not like the NaN then you might look into the function `xexpandarray.m`.

Note: It is recommended that the file names of the methods is names something like `imatm_<name of method>.m` in order to differentiate these fundamental algorithms from other files.

6 Grid sampling

Grid sampling may take place in either the 1D or 2D space. The choice for that is taken by setting the field *Grid before unfold*. When sampling in the 1D space the grid is defined by setting the sampling length (!) in one direction only. The sampling rate is the length of the grid cells. The unfolding in 2D space is in principle equivalent. However, the grid is know defined by sampling **rates**, rather than sampling lengths. Sampling rates being defined as the number of cells in each direction.



(a) After unfolding (b) Before unfolding

Figure 11: Gridsampling types. a) Gridsampling after unfolding with two different sampling rates, .

Important note: If you have been using the **AMTv1** program, which may be regarded as the prime version of the program, this note is relevant for you. The grid is no longer copied and dispaced by one half of the cell sizes. In case you still want to use this kind of gridsampling you may look into the file `imat_gridsampling.m` and uncomment the part taking care of the copying (do read the comments).

7 Thresholding

Thresholding is one of the most simple and most often used processing in image analysis and it is a part of the pre-processing. Thresholding may have effect on the gridsampling and the image in at least two ways. 1) pixels with values fulfilling the threshold values is abandoned from the image either by considering the pixel as part of a background or by removing the pixel from the image. The former is only relevant for images that are not unfolded and the later is relevant only for unfolded images. 2) pixels fulfilling the thresholds are unchanged in the image, but *can not* be member of the gridsampling set. In the table below the actions taken by imat for the possible combinations, is listed. The two cases presented above is in imat referred to as *Cut image* and *Cut grid*, respectively.

Thresholds are processed before generation of grid.

Cut image	Cut grid	Action
0	0	Image and grid unchanged
1	0	pixels fulfilling the thresholds are abandoned both from image and grid
0	1	pixels fulfilling thresholds are abandoned only from the grid
1	1	You do not want to do that

Determination of background is also a variant of thresholding. It differs from the above by being position dependent. In most cases a background is a basis on which the object of interest is placed (ex. a pizza on a black table). Then the corners of the image are by guarantee part of the background and probably also a large part of the border of the image. Background is in imat defined as the set of pixels that are path connected to the border and fulfill the threshold value. By this definition a black olive on the pizza will not become a part of the background. If this definition do not meet your requirement, you are obliged to find it yourself using the pre-processing file `imat_prepro.m`.

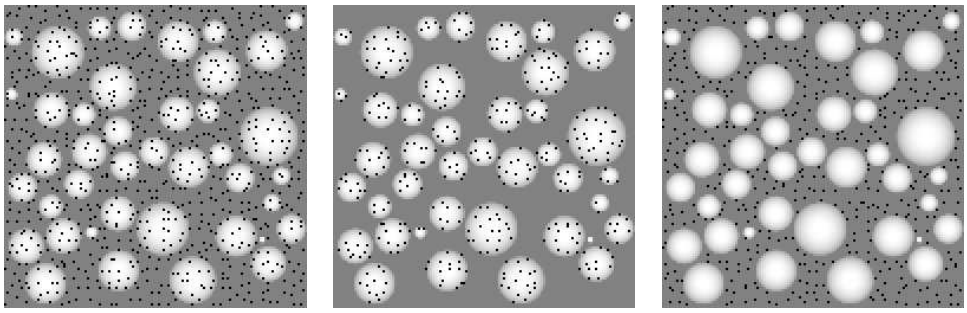


Figure 12: Effect of thresholding the grid (Clip grid) on an image with pixel value in the range 130-255. The dark background has the value 130. Left image: no thresholds. Middle image: grid points used due to threshold lower=130 and upper=255. Right image: lower=0 and upper=131.

8 Structure of the output variable `imato`

All information, as well as input parameters and output of an analysis, is stored in the structures variable `imato`. This variable is saved in a file with file name `imat_<index>.mat` in matlab binary format, where index is the output index specified. In order to access the contents it is necessary to load the file or alternatively declare `imato` as a global variable (before performing the analysis!), by typing `global imato` in the matlab command prompt.

Data of `imato` is accessed by using either `xstructbrowse` or typing `imato.<entry>`, where entry is the names of the fields present when just typing `imato`. For instance, `imato.resy` return the data of an analysis.

Note: You might find the algorithm `ximat2unscr.m` usefull for exporting numeric data to a format that can be imported directly into unscrambler. Check the help to the function in chapter 13 or type `help ximat2unscr` in the command prompt.

imato	resy resx pictures pc_lower pc_upper pc_lower_on pc_upper_on			
	setup	source	path single channel resize imtype	
		output analysis	 method	
			methodtxt prepro preprofile postpro postprofile unfold unfoldtxt smax points	
		grid	type	gbu systematic stratified keepgrid
			gridx gridy	
		cutting	type	plus onhill
			uppercutoff lowercutoff upperonhill loweronhill removebg bgcutoff	
		sys	softdelay mustdelay	
	time	start end		

Explanation of the entries:

`resy` and `resx` are matrices with the analysis result. In case the analysis method do not define an abscissa (`resx`) then `resx` is just 0 or index values. The matrices are arranged such that rows correspond to samples/pictures and the columns are variables.

`pictures` is a matrix containing the filenames in the sequence the images are being treated and in a sequence corresponding to the rows in `resx` and `resy`.

`pc_lower` and `pc_upper` contain the percentage of pixels that are lower and above the lower and upper cutoff values.

`pc_lower_on` and `pc_upper_on`

`setup` contain the setup for the calculation. Parts of the entries within the setup exist as both number fields and text fields. For instance the entry `preprofilecontent` contain the complete content of the pre-processing file used (default `imat_prepro.m`).

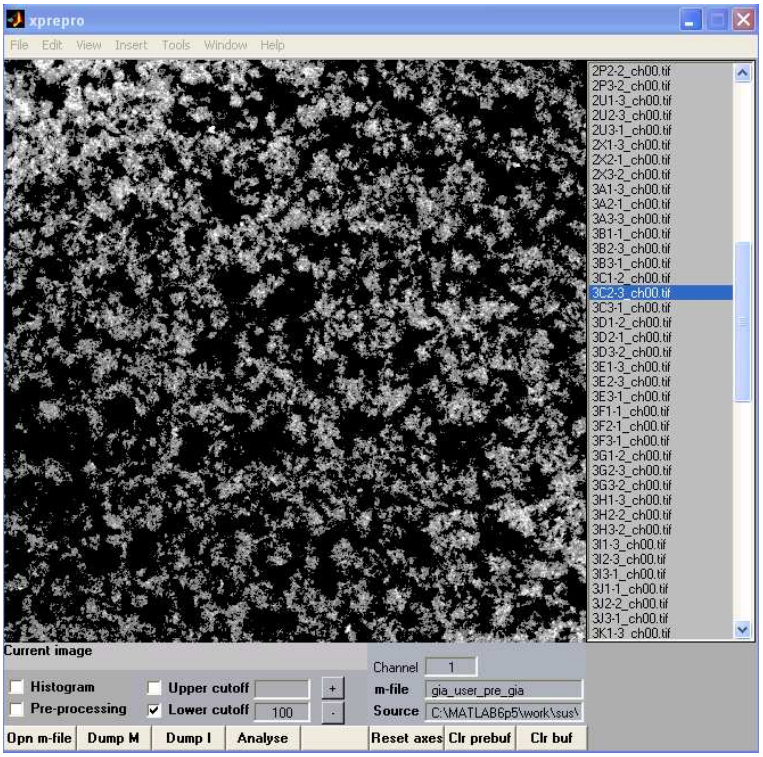
9 User guide for `xprepro`

The `xprepro` utility may used directly from the imat window by pressing the *View pre-processing* button, to investigate for instance the effect of a pre-processing of images. The folder or image chosen in the imat window is parsed directly to the xprepro window. You may, however, change the source path in the field named *Source*.

To start `xprepro` directly from the matlab command window, type

```
>> xprepro(1,path,'*.tif','C:\MATLABxpx\work\imat-dd-mm-yy
\imat\userfiles\imat_user_pre','1')
```

where `pwd` is the path to the image folder.



10 User guide for `xpluto`

Visualization of analysis results is effectively done by **`xpluto`**. The force of this utility is the possibility to group the results by the file name. If **`xpluto`** is used from the **`imat`** program, analysis data is parsed directly to **`xpluto`**, i.e. there is no need to load data, unless some old results is to be inspected.

When it used independently from **`imat`** the program may be started in two ways. If the data is already present in the matlab workspace, you may prefer to give the data as input arguments as

```
>> xpluto(Xdata,Ydata,Names)
```

where the rows in the variables correspond to one image, hence the number of rows in all three data variables must be equal. If they are not **`xpluto`** will try to transpose Xdata and/or Ydata to make it fit[10].

If data is not in workspace you may just type **`xpluto`** in the command prompt and load the data. If the data is embedded in a structured array, as the results from **`imat`** are a structure browser will help you to find the entry.

10.1 Names and groups

Naming of images is an important task to do before beginning analysis[11]. Example of a proper naming of images: Assume that you have 5 images of 3 different samples, the one possibility is to name the individual images after a

[10] That is of course not recommended to work like that.

[11] It will save you very much time in the end.

syntax like for instance: <sample name><image number>.tif. It is crucial for correct functionality of `xpluto`, that each term in the syntax is alternating between being syntaxes containing only letters and numbers. The 3 samples you may name for instance big, medium and small and the image number should then simply be numbers, 1,2,3,4,5, i.e.

5big1.tif 5medium1.tif 5large1.tif
4big2.tif 4medium2.tif 4large2.tif
3big3.tif 3medium3.tif 3large3.tif
2big4.tif 2medium4.tif 2large4.tif
1big5.tif 1medium5.tif 1large5.tif

The syntax for these names are *nsns*.

10.2 Grouped Plotting

In order to plot all the *big* images with same color etc., an index to the syntax must be defined. In the field *Plot by* the index is defined by typing the position(s) of 's' and/or 'n' in the syntax string that defines the groups. For the present case this index is just 1. Doing this the listbox *Possible groups* show the groups that the index generate. Groups of interest are moved to the listbox *Groups to plot* by selecting the group and right clicking with the mouse. Colors are chosen in similar way in the appropriate listboxes. The plotting is know done by pressing the *Plot* button.

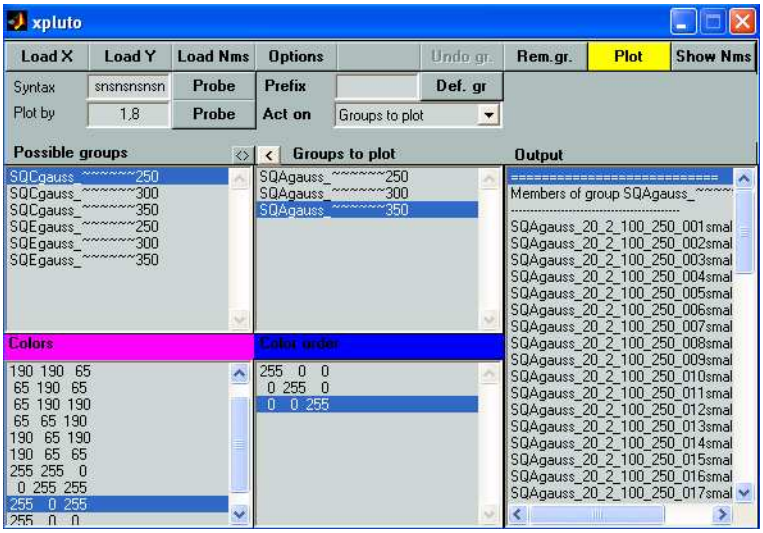


Figure 13: After clicking the Plot button on the imat screen, the syntax is probed. With a *Plot by* syntax set to 1,8 and selecting the groups of interest the spectra are ready to be plotted. NOTE: a pause is by default present between plotting of each group. Hit a key to plot the next group!

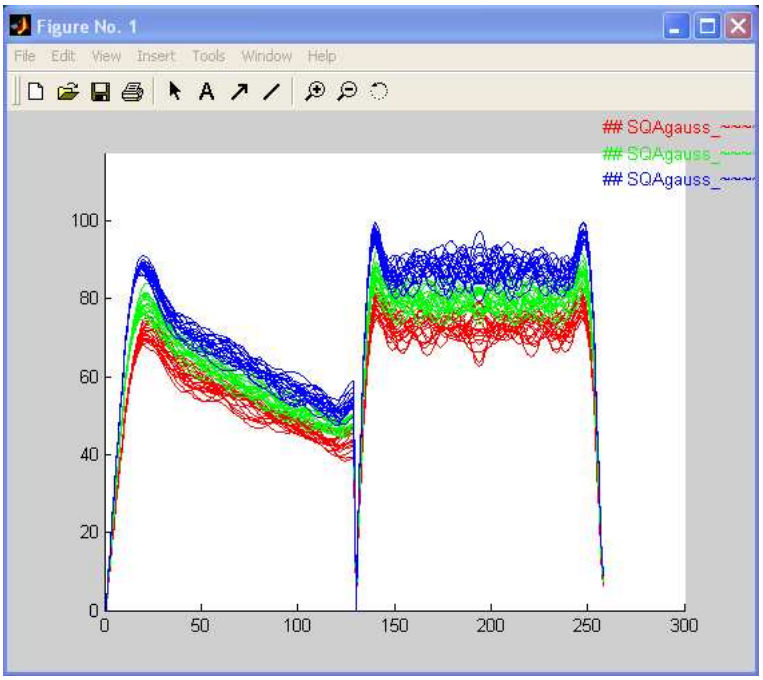


Figure 14: The 3 groups of spectra, which was chosen on the xpluto screen. Use the toolbar at the top to add labels and change ranges etc.

10.3 Selecting colors

If no colors are defined then the data is simply plotted using arbitrary colors. To add a color, select the color and right click with the mouse to move it over to the

10.4 Options

In the *Options menu* different kinds of plotting methods can be made. Especially useful is the *average groups* field. Using this type of plotting the standard deviation (or the maximum difference) of the spectra belonging to one groups is plotted around the mean of the spectra of the group. Such plots gives a first glance of how well spectra of sample groups separates. If they do not separate at all, then there is no need to proceed with more sophisticated data analysis methods, but rather a better image analysis algorithm should be seeked.

Also the option named filter is interesting. When it is turned on the differences between all combinations of the groups are plotted and in the end the average difference is plotted. This makes it easy to locate the regions of the spectra, which are important for discrimination.

11 User guide for `xstructbrowse`

The structure browser **xstructbrowse** is used by **xpluto** to load data embedded in a structured array. The browser may be used as a separate utility to browse through structures. To do this, in the command prompt type

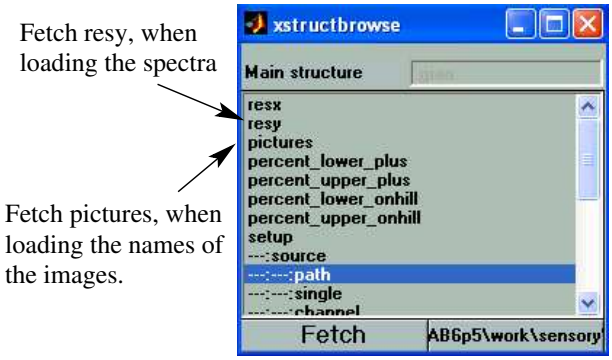
```
>> xstructbrowse(qqq,'qqq')
```

where **qqq** is the structure.

The *Fetch* button only has effect when it is used from **xpluto**. The button to the right show the content of the chosen entry. If pressing the button it is dumped to the matlab workspace. If you type

```
>> global xans
```

you can access the prompted data in the variable **xans** as well. Try it! It is very god when at a later point checking what really was done in the calculations.



12 Examples of application

See the web-site www.sensory.dk.

13 Algorithms in `xfiles`

See the pdf documentation located in the xfiles directory.

References

[1] Gonzales, R.C., Woods, R. E. and Eddins, S. L., Digital Image Processing using Matlab, Pearson Prentice Hall, 2004.

[2] Andrle, R., Mathematical Geology 1994, 26(1), 83.

[3] Johansen, S. M. B.,Laugesen, J. L., Dahl, C. K., Esbensen, K. H. and M. B. Frøst, Angle Measure Technique – Proposal of New Algorithm and Systematic Examination of Method Performances, Journal of Chemometrics, In preparation, 2004.