Title / Source dir: C:\MATLAB6p5\work\imat-16-03-05\xfiles
Date: March 16, 2005

xaccumulate.m

```
----------------------------------------
 function [y]=xaccumulate(x)
----------------------------------------
 Accumulation along rows of a matrix
----------------------------------------
 Input:
   x:  matrix
 Output:
   y:  accumulated matrix
----------------------------------------
 Example:

 a =

      1     2     3
      3     2     1
      1     1     1

 xaccumulate(a)

 ans =

      1     3     6
      3     5     6
      1     2     3
```

xaddborder.m

```
----------------------------------------------------------------------
 function [X]=xaddborder(X,add,value)
----------------------------------------------------------------------
 Add border rows and columns to a matrix / image
----------------------------------------------------------------------
 Input:
   X:      Matrix / image
   add:    width in units of elements/pixels to be added
   value:  if empty or not specified the value will be
           equal to the initial border
 Ouput:
   X:      New matrix / image in the same format as the input X
 Note:
           See also padarray.m from the dipum toolbox!
----------------------------------------------------------------------
 Example:
```

xamtcompress.m

xanglesum.m

```
-------------------------------------------------------
 function A=xanglesum(B)
-------------------------------------------------------
 Calculates the sum over angles
-------------------------------------------------------
 Input:
   B:  Matrix
 Output:
   A:  summed matrix, A(angle), i.e. a vector
-------------------------------------------------------
 Example:
```

xcells2str_h.m

```
-------------------------------------------------------------------
 function [out]=xcells2str_h(x,y)
-------------------------------------------------------------------
 Catenates a string array with a single string vertically
-------------------------------------------------------------------
 Input:
   x:  string array  / (or a string)
   y:  string        / (or a string array)
 Output:
   out:    string array
-------------------------------------------------------------------
 Example:
    x=[cellstr('gfdg');       out=[cellstr('gfdga');
       cellstr('gg')];             cellstr('gga')];
    y='a';
```

xchardelimiter.m

```
-----------------------------------------------------------
 function [groups]=xchardelimiter(grup,delimiter)
-----------------------------------------------------------

-----------------------------------------------------------
 Input:
   grup:
   delimiter:
 Output:
   groups:
-----------------------------------------------------------
 Example:
```

xcheckxyn.m

```
------------------------------------------------------------
 function [guidetxt,X,Y,N]=xcheckxyn(X,Y,N)
------------------------------------------------------------
 Check that X,Y and N matrix have equal row dimensions
 If not then they are truncated and a message is returned.
------------------------------------------------------------
 Input:
   X:      X-data
   Y:      Y-data
   N:      Names
 Output:
   guidetxt:   Information on the dimensions
   X:          Evt. modified X-data
   Y:          Evt. modified Y-data
   N:          Evt. modified names
------------------------------------------------------------
 Example:
```

xcircle.m

```
------------------------------------------------------------
 function [x,y]=xcircle(cx,cy,r)
------------------------------------------------------------
 Generates a matrix, that when unfolded (classic vertical)
 is a sinus curve X=a*sin(w*[1:N*M])
------------------------------------------------------------
 Input:
 N: rows
 M: columns
 a: amplitude
 w: frequency
 Output:
 X: image
------------------------------------------------------------
 Exxample:
```

xcolldist.m

xconnected.m

```
=====================================================
 Determination of a background surrounding a body.
 Structures inside the body will NOT be recognised as background,
 since this structure can not surround the body.
 Note: The value of <bgvalue> is crucial for a good determination.
=====================================================
 INPUT:
 pic           The picture in matrix form
 bgvalue   cutoff, values below this value is defined to be background.
 sgn          If 1 then background is above the cutoff value.
 init          initial point(s)... ex. [[1,1]; [256,256]].
=====================================================
```

```
 OUTPUT:
 background        The background in matrix form.
                          Elements of value one denotes a background pixel,
                          while pixel in the body take the value 0.
==================================================
```

xconnectedm.m

```
-------------------------------------------------------------------------------------
 function [background] = xconnectedm(pic,bgvalue1,bgvalue2,init)
-------------------------------------------------------------------------------------
 Determination of a background surrounding a body.
 Structures inside the body will NOT be recognised as background,
 since this structure can not surround the body.
 Note: The value of <bgvalue> is crucial for a good determination.
-------------------------------------------------------------------------------------
 Input:
   pic:        The picture in matrix form
   bgvalue:    Cutoff, values below this value is defined to be background.
   sgn:        If 1 then background is above the cutoff value.
   init:       Initial point(s)... ex. [[1,1]; [256,256]].
 Output:
   background: The background in matrix form.
               Elements of value one denotes a background pixel,
               while pixel in the body take the value 0.
-------------------------------------------------------------------------------------
 Example:
```

xconv.m

```
-----------------------------------------------------------
 function [X]=xconv(X,S)
-----------------------------------------------------------
 Arithmetric mean (blurring) of image
 Cover function of xcconv.c / xcconv.dll
-----------------------------------------------------------
 Input:
   X:  A matrix or one-layer image
   S:  integer, (2*S+1) is the size of a submatrix
       over wich to calculate a mean value
 Output:
   X:  Averaged matrix / image
-----------------------------------------------------------
 Example:
```

xconvert2numtype.m

```
-------------------------------------------------------------------
 function x=xconvert2numtype(x,bit,type)
-------------------------------------------------------------------
 Convert x specified type
-------------------------------------------------------------------
 Input:
   x:      Any numeric thing
```

```
  bit:    bits
  type:   0~signed, 1~unsigned, 2~single, 3~double
 Output:
  x:      converted x
-------------------------------------------------------------------
 Example:
```

xdecomposegroups.m

```
-------------------------------------------------------------------------------
 function [out]=xdecomposegroups(STRINGARRAY,NUMBERARRAY,sortafter,syntax)
-------------------------------------------------------------------------------

-------------------------------------------------------------------------------
 Input:
  STRINGARRAY:
  NUMBERARRAY:
  sortafter:
  syntax:
 Output:
  Out:
-------------------------------------------------------------------------------
 Example:
```

xdespace.m

```
-------------------------------------
 function [s]=xdespace(s)
-------------------------------------
 Removes any space chars in a string
-------------------------------------
 Input:
  s:  a string
 Output:
  s:  the string without spaces
-------------------------------------
 Example:
```

xdir.m

```
-----------------------------------------------------------
 function [files]=xdir(path)
-----------------------------------------------------------
 List filenames in a specified folder.
-----------------------------------------------------------
 Input:
  path:   The path where the wanted files are located
 Output:
  files:  List of filenames in the format of cells
-----------------------------------------------------------
 Examples:
```

xdivergence.m

```
------------------------------------------------
 function [H,V,DL,DR,MINI]=xdivergence(X)
------------------------------------------------
 Calculates the divergence in 4 directions
 of the image
------------------------------------------------
 Input:
   X:  Matrix
 Ouput:
   H:  Horizontal difference (-)
   V:  Vertical (|)
   DL: Diagonal (\)
   DR: Diagonal (/)
------------------------------------------------
 Example:
```

xdivergence2.m

```
------------------------------------------------
 function [H,V,DL,DR,MINI]=xdivergence(X)
------------------------------------------------
 Calculates the divergence in 4 directions
 of the image
------------------------------------------------
 Input:
   X:  Matrix
 Ouput:
   H:  Horizontal difference (-)
   V:  Vertical (|)
   DL: Diagonal (\)
   DR: Diagonal (/)
   MINI:   The smallest elements of H,V,DL and DR.
------------------------------------------------
 Example:
```

xedge.m

```
------------------------------------------------
 function [border]=xedge(X,mode)
------------------------------------------------
 Determines the edge of an object in an image.
------------------------------------------------
 Input:
   X:      Matrix / image
   mode:   'inner' / 'outer'
 Output:
   border: the edge
------------------------------------------------
 Example:
```

xexpandarray.m

```
-------------------------------------------------------
 function [X,v]=xexpandarray(X,v,expandwith)
-------------------------------------------------------
 Column expansion of arrays...
-------------------------------------------------------
 Input:
   X:           Matrix
   v:           Vector
   expandwith: element with wich to expand
 Output:
   X:      Exapaned matrix (if it was necesary)
   v:      Exapaned vector (if it was necesary)
-------------------------------------------------------
 Example:
   X=[1 2 3;5 6 7];
   v=[1 2 3 4 5];  and expandwith=NaN
   New values:
   X=[1 2 3 NaN NaN; 5 6 7 NaN NaN];
   v has more columns than X, hence no change
```

xfillspheres.m

```
-------------------------------------------------------------------------
 function [pic,dist,err]=xfillspheres(ids,radii,options)
-------------------------------------------------------------------------
 Generate an image of Gaussian distributed spheres.
 The Gaussian distribution:
 The images is generated by first placing the larger (according
 to the distribution) spheres followed by the next largest etc.
-------------------------------------------------------------------------
 Input:
   ids:        Indices given to the file names, if a vector, then N images
               is generated, where N is the length of ids.
   radii:      Vector containing the radii
   options:    [image size ({256}), save image ({0}/1)]
               The file name is defined by the radii.
 Output:
   pic:    The image
   info:   Structured array with the distribution (digitalized)
           info.dist:      is the distribution
           info.centers:   contains the centers and radii [x y R]
   err:    There is no possible errors
-------------------------------------------------------------------------
 Example:
```

xfillspheresx.m

```
-------------------------------------------------------------------------
 function [pic,dist,err]=xfillspheres(ids,radii,options)
-------------------------------------------------------------------------
 Generate an image of Gaussian distributed spheres.
 The Gaussian distribution:
 The images is generated by first placing the larger (according
```

```
 to the distribution) spheres followed by the next largest etc.
-------------------------------------------------------------------------
 Input:
   ids:        Indices given to the file names, if a vector, then N images
               is generated, where N is the length of ids.
   radii:      Vector containing the radii
   options:    [image size ({256}), save image ({0}/1)]
               The file name is defined by the radii.
 Output:
   pic:    The image
   info:   Structured array with the distribution (digitalized)
           info.dist:      is the distribution
           info.centers:   contains the centers and radii [x y R]
   err:    There is no possible errors
-------------------------------------------------------------------------
 Example:
```

xfirstmin.m

```
-----------------------------------------------------
 function [first]=xfirstmin(x)
-----------------------------------------------------
 The position of the first minimum in a vector
-----------------------------------------------------
 Input:
   x:  A vector
 Output:
   first:  The position of the first minimum
   val:    The minimum value
-----------------------------------------------------
 Example:
 x=[5 4 3 2 4 5 4 1 5 6];
          |       |
          minimums
 [m,v]=xfirstmin(x)
 m=4, v=2
```

xfolding.m

```
-----------------------------------------------------------
 function [y]=xfolding(x,n,m,type,dir)
-----------------------------------------------------------
 Folding of a vector into a matrix
-----------------------------------------------------------
 Input:
   x:      the vector
   n:      rows of the wanted matrix
   m:      columns of the wanted matrix
   type:   folding type, classic or snake
   dir:    folding direction, hor(izontal) or ver(tical)
 Output:
   y:      the matrix
-----------------------------------------------------------
```

```
Example:
>> v=[1 2 3 4 5 6 7 8 9];
>> xfolding(v,3,3,'snake','hor')

ans =

     1     2     3
     6     5     4
     7     8     9
```

xformfactor.m

```
-----------------------------------------
function [ff]=xformfactor(p,a)
-----------------------------------------
Calculates the formfactor 4*pi*a/p^2
-----------------------------------------
Input:
   p:    vector with perimeters
   a:    vector with areas
Output:
   ff:   the form factors
-----------------------------------------
Example:
```

xgaussianspheres.m

```
-------------------------------------------------------------------------
[pic,dist,err]=xgaussianspheres(sigma,m,A,ids,xmin,xmax,step,options)
-------------------------------------------------------------------------
Generate an image of Gaussian distributed spheres.
The Gaussian distribution:
 g(x) = 1/(sigma*sqrt(2*pi)) * exp[ -1/2 * ((x-m)/sigma)^2 ]
The images is generated by first placing the larger (according
to the distribution) spheres followed by the next largest etc.
-------------------------------------------------------------------------
Input:
  ids:        Indices given to the file names
  sigma:
  m:
  A:
  xmin:       smallest sphere
  xmax:       largest sphere
  step:       step in size of spheres
  options:    [image size (256), save image (0/1), it (1000)]
Output:
  dist:   The distribution (digitalized)
  X:      The image
  err:    if 1, then the distribution is impossible within it iterations
-------------------------------------------------------------------------
Example:
  x=xgaussianspheres(1,15,35,200,1,0,100,5);
```

xgetix.m

```
-----------------------------------------------
 function [i,j]=xgetix(x,s)
-----------------------------------------------
 Get the indices of occurencies of a value
 in a matrix.
-----------------------------------------------
 Input:
   x   matrix or vector
   s   scalar value
 Output:
   [i,j]   row,column
-----------------------------------------------
 Example:
```

xgetos.m

```
-----------------------------------------------------
 function [DELI,os]=xgetos
-----------------------------------------------------
 Return os specific file delimiter and a string
 telling the operating system.
-----------------------------------------------------
 Input:
   No input
 Output:
   DELI:   ux-like '/' and windows '\'
   os:     ux/win
-----------------------------------------------------
 Example:
```

xgetsyntax.m

```
-------------------------------------------------------------------
 function [syntax,errormessage]=xgetsyntax(names)
-------------------------------------------------------------------
 Generates the syntax of a set of names
-------------------------------------------------------------------
 Input:
   names:  a matrix of strings containing the names
 Output:
   syntax: the syntax, a string of 's' and 'n'
-------------------------------------------------------------------
 Example:
  names = '2abcd8nm'
          '2q3y'
   => syntax = 'nsns'

 Symbols '-'; '+'; '='; '_'; '.' are treated as strings and
 Both upper and lower case is allowed but not distinguised.
 Other chars are NOT allowed !!!
```

xgroup_isos.m

```
-------------------------------------------------------------------
 function xgroup_isos(X,s)
-------------------------------------------------------------------
 Grouping of X. Each group must be of the same size
 and adjecent in the matrix X
-------------------------------------------------------------------
 Input:
   X:  matrix where the rows are samples
   s:  size og the groups
 Output:
   Grouped:
-------------------------------------------------------------------
 Example:
```

xgroups.m

xhardcolor.m

```
-------------------------------------------------------------
 function [hard]=xhardcolor(a)
-------------------------------------------------------------
 List of colors that all can be distinghuised clearly
-------------------------------------------------------------
 Input:
   a:  dummy
 Output:
   hard:   colors
-------------------------------------------------------------
 Example:
```

ximat2unscr.m

```
-----------------------------------------------------------------------
 function ximat2unscr(resy,pictures,RES,filename,AYX,strip)
-----------------------------------------------------------------------
 Convert resy matrix and pictures to a txt-format compatible
 with a format required by unscrambler
-----------------------------------------------------------------------
 Input:
   resy:       Y-data (e.x. AMT matrix)
   pictures:   filenames of picture names
   RES:        columns in resy
   filename:   save conversion in filename
   AYX:        text string, specifies the y-data, that are
               to be exported. Ex.: '110' only exports MA and MDY.
   strip:      if 1, then extension is removed.
 Output:
   No output, result saved in filename
-----------------------------------------------------------------------
 Example:
   load('IMAT_<name provided by you>.mat');
```

```
  ximat2unscr(imato.resy,imato.pictures,251,'test.txt','110',1)
  where imato.resy and imato.pictures are the result of AMT analysis
  with SMAX=250. It will export MA and MDY without extension.
```

xinvertimage8.m

```
---------------------------------------------------------------
 function [X]=xinvertimage8(X);
---------------------------------------------------------------
 Inversion / negation of an 8 bit image
---------------------------------------------------------------
 Input:
   X:  8 bit image (other formats are allowed)
 Output:
   X:  Inverted 8 bit image (output format is maintained
   8bit=>8bit, double=>double).
---------------------------------------------------------------
 Example:
```

xisafunction.m

```
---------------------------------------------------------------
 function ok=isafunction(fn)
---------------------------------------------------------------
 Determines wheter a file is a function or a script
---------------------------------------------------------------
 Input:
   fn: file name
 Output:
   ok: 1~function, 0~non-function (script)
---------------------------------------------------------------
 Example:
```

xlastcol.m

```
-----------------------------------------------------
 function [y]=xlastcol(x,n)
-----------------------------------------------------
 The n last elements/columns in a vector/matrix
-----------------------------------------------------
 Input:
   x:  vector or matrix
   n:
 Ouput:
   y:  the last ...
-----------------------------------------------------
 Example:
```

xlastmin.m

```
-----------------------------------------------------
 function [last]=xlastmin(x)
-----------------------------------------------------
 The position of the last minimum in a vector
-----------------------------------------------------
 Input:
   x:  A vector
 Output:
   last:  The position of the last minimum
-----------------------------------------------------
 Example:
```

xlocmins.m

```
-----------------------------------------------------
 function [val,locm]=xlocmins(x,cir)
-----------------------------------------------------
 Get all minimums in a vector.
-----------------------------------------------------
 Input:
 Output:
-----------------------------------------------------
 Example:
```

xmakehelp.m

```
-----------------------------------------------------------------------
 function xmakehelp(title)
-----------------------------------------------------------------------
 Make help doc on the current folder.
-----------------------------------------------------------------------
 Input:
   title:  Title of the document. If not supplied then the title
           by default is the current path.
 Output:
   A latex file:  xhelp.tex
-----------------------------------------------------------------------
 Example:
```

xmaxij.m

```
-----------------------------------------------------
 function [i,j]=xmaxij(x)
-----------------------------------------------------
 The index (ij) of the maximum element in a matrix
-----------------------------------------------------
 Input:
   x:  matrix
 Output:
   i:  row
   j:  column
 Note: If multiple solutions then all are listed
```

```
-----------------------------------------------------
 Example:
```

xmean.m

```
-----------------------------------------------------
 function m=xmean(x2,d)
-----------------------------------------------------
 Averaging. Interpreting NaN as missing.
-----------------------------------------------------
 Input:
   x2: matrix
   d:  1~over rows, 2~over cols
 Output:
   m:          The average
   nanix2:     NaN ix
-----------------------------------------------------
 Example:
```

xmergeimatos.m

```
-----------------------------------------------------
 function [imato1]=xmergeimatos(imato1,imato2)
-----------------------------------------------------
 Merges to imat output files into one file
-----------------------------------------------------
 Input:
   imato1:
   imato2:
 Output:
   imato1:
-----------------------------------------------------
 Example:
```

xmin2ij.m

```
-------------------------------------------------------------------
 function [ij]=xmin2ij(x)
-------------------------------------------------------------------
 Like xminij, but output is a Nx2 matrix with the
 rows being the indicies.
-------------------------------------------------------------------
 Input:
   x:  a matrix
 Output:
   ij: a matrix
-------------------------------------------------------------------
 Example:
```

xminij.m

```
-------------------------------------------------------------------
 function [i,j]=xminij(x)
-------------------------------------------------------------------
 The index (ij) of the minimum element in a matrix/vector
-------------------------------------------------------------------
 Input:
   x:  matrix
 Output:
   i:  row
   j:  column
 Note: If multiple solutions then all are listed
-------------------------------------------------------------------
 Example:
```

xminn.m

```
-------------------------------------------------------------------
 function [i,j]=xminn(X,n)
-------------------------------------------------------------------
 n Smallest elements in a matrix
-------------------------------------------------------------------
 Input:
   X:  matrix
   n:  an integer smaller than the numer of elements in X.
       It is the number of minimum elements wanted.
 Output:
   i:  row
   j:  column
-------------------------------------------------------------------
 Example:
```

xmirrormatrix.m

```
-------------------------------------------------------
 function [x]=xmirrormatrix(n)
-------------------------------------------------------
 Indentity matrix with the ones in the other diagonal.
-------------------------------------------------------
 Input:
   n:  rows
 Output:
   x:  matrix
-------------------------------------------------------
 Example:
```

xmovaverage.m

xnamesofgroup.m

```
ixc=[ixc swhat+1];
ncomma=sum(what==',');
```

xnan2val.m

```
-------------------------------------------------------------------
 function [x2,nanix2]=xnan2val(x2,val)
-------------------------------------------------------------------
 Replace NaN by specified value in a matrix / vector
-------------------------------------------------------------------
 Input:
   x2:     matrix / vector
   val:    scalar / vector
 Output:
   x2:     New matrix / vector
   nanix2: Index
-------------------------------------------------------------------
 Example:
   If val is a vector, then the lenght must be equal to the number
   of columns in the matrix x2.
   q=[1 2 NaN 4; 6 NaN 8 9];
   xnan2val(q,[1 2 3 4]);
 ans =

     1     2     3     4
     6     2     3     9
```

xnumtype.m

```
-----------------------------------------------------------
 function [bit,type]=xnumtype(x)
-----------------------------------------------------------
 Returns the bit of x and its type.
-----------------------------------------------------------
 Input:
   x:  Any numeric thing
 Output:
   bit:    bits
   type:   0~signed, 1~unsigned, 2~single, 3~double
-----------------------------------------------------------
 Example:
```

xoperator3.m

```
 MATRIX
 pictures
 syntax = of the filename  'nsn'
 sortsyntax = part of the filename syntax after which sorting must be done
               in order to do the operator task
 sortsequence = 'ABCDEFGHIJKLMNOPQR'
q=load('GIA_lins1.mat');
AMT=q.giao.resy;
pictures=q.giao.pictures;
 [a b c]=xoperator3(AMT,pictures,'nsnsns', '12', 'analf' ,0); MAKROyoghurt

 [a b c]=xoperator3(AMT,pictures,'nsnsns', '12', 'mikro' ,0); MIKROyoghurt

 [a b c]=xoperator3(AMT,pictures,'nsnsns', '12', 'mikra' ,0); MIKROyoghurt all
```

xpathfile.m

xplotamt_bysyntax.m

```
AMT      = matrix
pictures = picture names
syntax     = string
sortafter  = string
what       = string

EXAMPLE:
plot all A and Ref grouped such that
one group is all A with -2
and all Ref is with -3

    pictures is=
                 '12Ref1-3_ch00.tif'
                 '12Ref2-2_ch00.tif'
                 '12Ref3-3_ch00.tif'
                 '2A1-2_ch00.tif'
                 '2A2-2_ch00.tif'
                 '2A3-1_ch00.tif'
                 '2C1-3_ch00.tif'
                 '2C2-3_ch00.tif'
whith syntax     'nsnsnsns'
                   |  |
sortafter         12345678    -->  '25'
```

xpoccur.m

```
-------------------------------------------------------------
function [ix,pos]=xpoccur(x,element,n)
-------------------------------------------------------------
The indexed matrix (ix) and of the n'th occurence
and position (pos) of an element in a matrix
-------------------------------------------------------------
Input:
  x:          a matrix or vector
  element:    element to look for
  n:          pay atention to the n'th ocurrence
Output:
  ix:         indexed matrix (0's and 1's)
  pos:        the position (not ij)
-------------------------------------------------------------
Example:
```

xpolyxy.m

xpolyxy2.m

```
F = a(1)*(x-a(2)).^2+a(3)*(y-a(4)).^2   +  a(5)*(x-a(6)).*(y-a(7))   +  a(8);
```

xpolyxyfit.m

```
X1=Xtemp1+X;
```

xpolyxyfit2.m

```
X2=xreyscale(X2,255);
X2=double(histeq(uint8(X2)));
```

xpwfilter.m

```
-----------------------------------------------------------------
 function [X]=xpwfilter(X,N,M,S,cut,pixn)
-----------------------------------------------------------------
 Cover m-file for xcpwfilter.c
 Calculates piecewise linear conditional blurring
 or contrast enhancement.
-----------------------------------------------------------------
 Input:
   X:      image
   N:      rows
   M:      columns
   S:      size of submatrix
   cut:    Cutoff
   pixn:   the condition. If the submatrix contain more than
           pixn pixels of values lower than cut, then all
           pixels in the submatrix are attributed the smaller
           value of the submatrix and the opposite. If the
           condition is not met, then the submatrix is left
           unchanged.
 Output:
   X:
-----------------------------------------------------------------
 Example:
```

xradiussum.m

```
-------------------------------------------------------------
 function vv=xradiussum(x)
-------------------------------------------------------------
 Calculates the radius sum of a matrix, with respect to the
 center of the matrix.
-----------------------------------------------------------------
 Input:
   x:  a square matrix
   vv: a vector with the radius sum.
-----------------------------------------------------------------
 Example:
```

xrevert.m

```
------------------------------------
 function xrevert(X)
------------------------------------
 Revert matrix or vector
------------------------------------
 Input:
   x: matrix or vector
 Ouput:
   x: the reverted matrix/vector
------------------------------------
 Example:
```

xreyscale.m

```
-----------------------------------------------------
function [X,par]=xreyscale(X,low,high,par)
-----------------------------------------------------
 Formerly called xnormalize, by mistake!
 Re-scales the values of a matrix to within the
 interval [low;high]
-----------------------------------------------------
 Input:
   X:      A matrix or image (must be double)
   low:    New lower limit
   high:   New upper limit
 Output:
   X:      Scaled matrix / image
   par:    parameters used to scale
-----------------------------------------------------
 Example:
```

xroundness.m

```
-----------------------------------------------------------
 function [r]=xroundness(l,a)
-----------------------------------------------------------
 Calculates the roundness 4*a/(pi*l^2)
-----------------------------------------------------------
 Input:
    l:    vector with maximum projected lengths
    a:    vector with areas
 Output:
    r:    the roundness
-----------------------------------------------------------
 Example:
```

xsearch.m

```
-------------------------------------------------------------------
 function xsearch(files,searchfor,path)
-------------------------------------------------------------------
 Serach after content in the files in a folder
```

```
------------------------------------------------------------
 Input:
   files:      filenames or part of filename group, ex. *.m
   searchfor:  string to look for.
   path:       path to search in.
 Output:
   No outout
 Example:  xsearch('*.m','percent')
```

xsetmatrixij.m

```
------------------------------------------------------
 function [X]=xsetmatrixij(X,iv,jv,val)
------------------------------------------------------
 Insert the value val into indices iv and jv in a matrix
------------------------------------------------------
 Input:
   X:
   iv:
   jv:
   val:
 Output:
   X:
------------------------------------------------------
 Example:
```

xsinusimg.m

```
-------------------------------------------
function [y,x]=xsinusimg(N,M,a,w)
-------------------------------------------
 Generates a matrix, that when
 unfolded (classic vertical) is a
 sinus curve X=a*sin(w*[1:N*M])
-------------------------------------------
 Input:
 N: rows
 M: columns
 a: amplitude
 w: frequency
 Output:
 y:  The image
   x:  x
-------------------------------------------
 Example:
```

xsplithalf_random.m

xstd.m

```
-------------------------------------------------------
 function s=xstd(x2,d)
-------------------------------------------------------
 Standard deviation. Interpreting NaN as missing.
-------------------------------------------------------
 Input:
   x2: matrix
   d:  1~over rows, 2~over cols
 Output:
   s:  The standard deviation
-------------------------------------------------------
 Example:
```

xstrfindix.m

```
-------------------------------------------------------
 function [ix]=xstrfindix(s,sa)
-------------------------------------------------------

-------------------------------------------------------
 Input:
   s:
   sa:
 Output:
   ix:
-------------------------------------------------------
 Example:
```

xstripfilename.m

```
-------------------------------------------------------------
 function [new,ext]=xstripfilename(filenames)
-------------------------------------------------------------
 Divides filenames into name and extension
-------------------------------------------------------------
 Input:
   filenames:  string array with filenames in the rows
 Output:
   new:        the filename (without extension)
   ext:        the extension
-------------------------------------------------------------
 Example:
   filenames=
           [ '3P2-2_ch00.tif' ;
             '3P3-3_ch...tif' ;
             '3Q1-3_ch00.tif']
   new=                          ext=
        [ '3P2-2_ch00' ;             [ 'tif' ;
          '3P3-3_ch..' ;               'tif' ;
          '3Q1-3_ch00']                'tif' ]
```

xtextwrite.m

```
-------------------------------------------------
 function xtextwrite(T,f,d,r,u)
-------------------------------------------------
 Write a textmatrix to a file.
-------------------------------------------------
 Input:
   T:  matrix
   f:  filename
   d:  deblank
   r:  replace tab by space
   u:  replace sequential space by one space only
 Output:
   No output
-------------------------------------------------
 Example:
```

xtime.m

```
-------------------------------------------------------------------
 function [tdiff,h,m,s]=xtime(t1,t2,p)
-------------------------------------------------------------------
 Convert matlab start and end time to a period i hours,
 minutes and seconds.
 Input:
   t1:     Time (start)
   t2:     Time (end)
   p:      A time period
 Output:
   h: hours
   m: minutes
   s: seconds
 Usage:
 Either t1 and t2 is specified
 or t1=t2=[] and instead the period is specified.
-------------------------------------------------------------------
 Example:
```

xzero2empty.m

```
-----------------------------------------
 function [y]=xzero2nan(x)
-----------------------------------------
 Convert zeros to [], i.e. remove zeros
-----------------------------------------
 Input:
   x:  vector
 Ouput:
   x:  vector
 Example:
```

xzero2nan.m

```
-----------------------------------------
 function [y]=xzero2nan(x)
-----------------------------------------
 Convert zeros to NaN
-----------------------------------------
 Input:
   x:  vector
 Ouput:
   x:  vector
 Example:
```