Numerical Methods in TEM

3-dimensional images:
producing and processing 3-dimensional data stacks

Christoph T. Koch

Max Planck Institut für Metallforschung

http://hrem.mpi-stuttgart.mpg.de/koch/Vorlesung

Topics covered in this lecture

- Applications of 3D data stacks.
- Setting and extracting Properties of 3-dimensional data stacks.
- Adding and extracting planes / data volumes.
- Acquiring 3D data stacks.
- Aligning 3D data stacks.

Applications

- Image series
  - Focal series
  - EFTEM series
  - Time laps series
    (e.g. in-situ, or beam damage experiments)
  - Multiple exposure series
    (e.g. for post-exposure drift compensation)

- Spectrum maps

Obtain the size of a 3-dimensional image

Image stack = getFrontImage()

- get3dsize(stack, xsize, ysize, zsize)

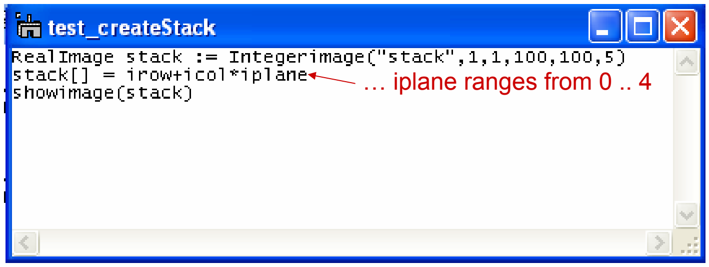
- stack.getSize(xsize, ysize, zsize)

Both ways of obtaining the image stack size may be used interchangeably.

Creation of an image stack

- Create a stack of `sz` real single precision images of size `sx x sy`:
  `RealImage stack := Realimage( “Stack”, 4, sx, sy, sz);`
- Create a stack of `sz` real double precision complex images of size `sx x sy`:
  `ComplexImage stack:=Compleximage( “Stack” , 16, sx, sy, sz);`
- Create a stack of `sz` 2-byte signed integer images of size `sx x sy`:
  `RealImage stack := Integerimage( “Stack”, 2, 1, sx, sy, sz);`
- Create a stack of `sz` 1-byte unsigned integer images of size `sx x sy`:
  `RealImage stack := Integerimage( “Stack”, 1, 0, sx, sy, sz);`

Indexing different image planes in a 3D stack



```
RealImage stack := Integerimage("stack",1,1,100,100,5)
stack[] = irow+icol*iplane
showimage(stack)
```

… iplane ranges from 0 .. 4

iplane=0
iplane=1
iplane=2
iplane=3
iplane=4

These images are of data-type 1-byte signed integer.
Their dynamic range is therefore limited to -128 .. +127

2 Ways to Acquire a CCD Image

- Use ‘SSC…’ functions
  - the old style of image acquisition
- Use the CameraManager (‘CM_...’) functions
  - a little more flexible
  - provides access to the DM Camera dialog settings

SSC - commands

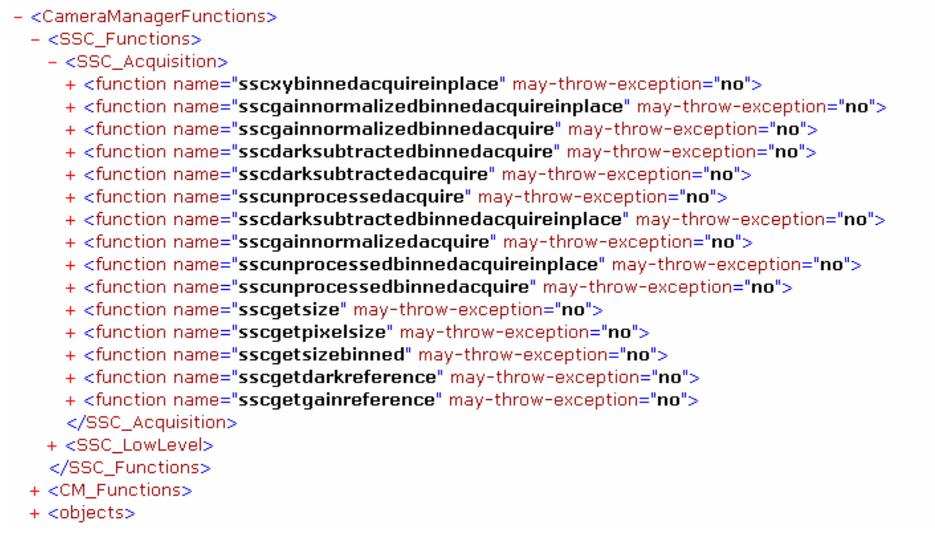
Command name:

	Unprocessed					
SSC	Darksubtracted		Acquire		(…)	(base)
	Gainnormalized					
		Binned				(optional)
				Inplace		(optional)

Parameters:	[img]	(the target image, if ‘in place’ in command name)
	exposure time	(the exposure time in seconds)
	[binning]	(the binning, if ‘binned’ in command name)
	top	(usually 0)
	left	(usually 0)
	bottom	(usually height of CCD camera)
	right	(usually width of CCD camera)

(The last 4 parameters can be used to select sub-regions of the CCD)

SSC function overview



```
- <CameraManagerFunctions>
  - <SSC_Functions>
    - <SSC_Acquisition>
      + <function name="sscxybinnedacquireinplace" may-throw-exception="no">
      + <function name="sscgainnormalizedbinnedacquireinplace" may-throw-exception="no">
      + <function name="sscgainnormalizedbinnedacquire" may-throw-exception="no">
      + <function name="sscdarksubtractedbinnedacquire" may-throw-exception="no">
      + <function name="sscdarksubtractedacquire" may-throw-exception="no">
      + <function name="sscunprocessedacquire" may-throw-exception="no">
      + <function name="sscdarksubtractedbinnedacquireinplace" may-throw-exception="no">
      + <function name="sscgainnormalizedacquire" may-throw-exception="no">
      + <function name="sscunprocessedbinnedacquireinplace" may-throw-exception="no">
      + <function name="sscunprocessedbinnedacquire" may-throw-exception="no">
      + <function name="sscgetsize" may-throw-exception="no">
      + <function name="sscgetpixelsize" may-throw-exception="no">
      + <function name="sscgetsizebinned" may-throw-exception="no">
      + <function name="sscgetdarkreference" may-throw-exception="no">
      + <function name="sscgetgainreference" may-throw-exception="no">
      </SSC_Acquisition>
    + <SSC_LowLevel>
    </SSC_Functions>
  + <CM_Functions>
  + <objects>
```

(http://hrem.mpi-stuttgart.mpg.de/koch/Vorlesung/Script/DM_PluginFunctions.html)

SSC command examples

number width,height
number expTime = 1
number binning = 1
image img1,img2

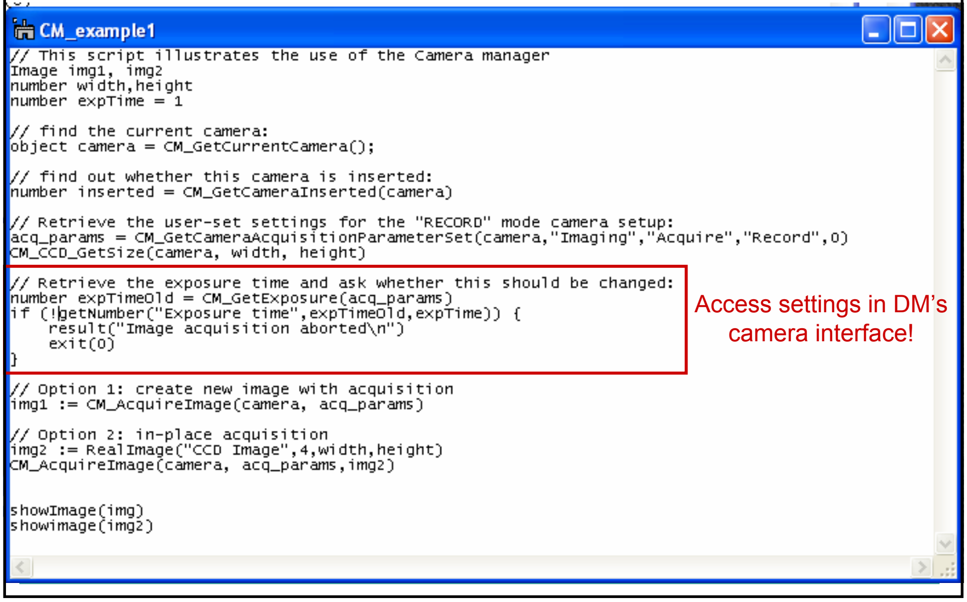
// find out the size of the CCD camera
SSCGetSize(width,height)

// Example 1: create a new image with every acquisition
img1 := **SSCGainnormalizedAcquire**(expTime,0,0,height,width)

// Example 2: copy the newly read image into an existing image
img2 := RealImage("CCD Image",4,width,height)
SSCDarksubtractedBinnedAcquireInplace(img2,expTime,binning, 0,0,height,width)

Camera Manager (a few hundred functions)

An example script that acquires 2 images: one in place, on newly created

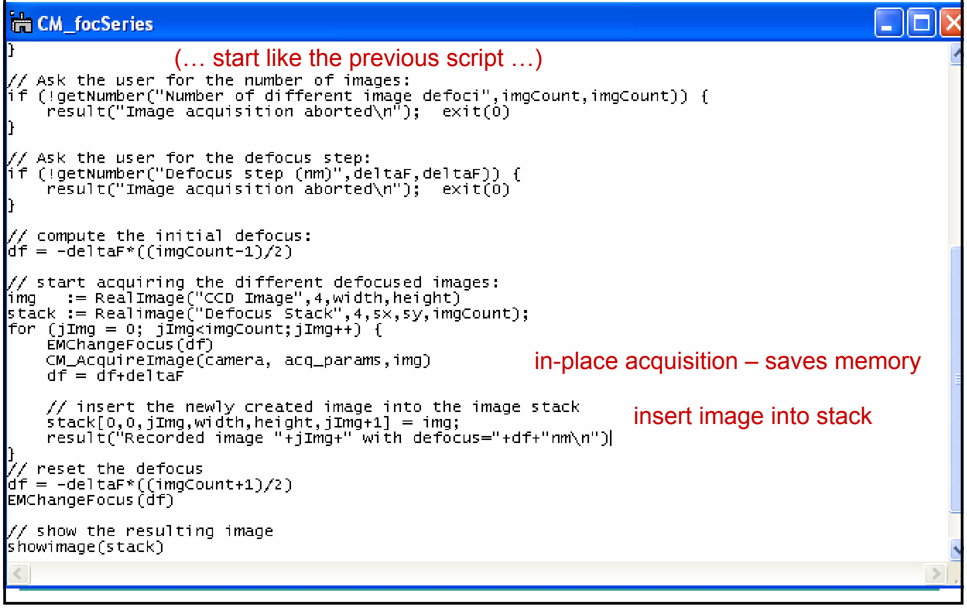


```
// This script illustrates the use of the Camera manager
Image img1, img2
number width,height
number expTime = 1

// find the current camera:
object camera = CM_GetCurrentCamera();

// find out whether this camera is inserted:
number inserted = CM_GetCameraInserted(camera)

// Retrieve the user-set settings for the "RECORD" mode camera setup:
acq_params = CM_GetCameraAcquisitionParameterSet(camera,"Imaging","Acquire","Record",0)
CM_CCD_GetSize(camera, width, height)

// Retrieve the exposure time and ask whether this should be changed:
number expTimeOld = CM_GetExposure(acq_params)
if (!getNumber("Exposure time",expTimeOld,expTime)) {
    result("Image acquisition aborted\n")
    exit(0)
}

// Option 1: create new image with acquisition
img1 := CM_AcquireImage(camera, acq_params)

// Option 2: in-place acquisition
img2 := RealImage("CCD Image",4,width,height)
CM_AcquireImage(camera, acq_params,img2)


showImage(img)
showimage(img2)
```

Access settings in DM's camera interface!

Acquisition of a focal series image stack

This script acquires a focal series of imgCount images with defocus step deltaF



(… start like the previous script …)

```
}
// Ask the user for the number of images:
if (!getNumber("Number of different image defoci",imgCount,imgCount)) {
    result("Image acquisition aborted\n");  exit(0)
}

// Ask the user for the defocus step:
if (!getNumber("Defocus step (nm)",deltaF,deltaF)) {
    result("Image acquisition aborted\n");  exit(0)
}

// compute the initial defocus:
df = -deltaF*((imgCount-1)/2)

// start acquiring the different defocused images:
img   := RealImage("CCD Image",4,width,height)
stack := RealImage("Defocus Stack",4,sx,sy,imgCount);
for (jImg = 0; jImg<imgCount;jImg++) {
    EMChangeFocus(df)
    CM_AcquireImage(camera, acq_params,img)
    df = df+deltaF

    // insert the newly created image into the image stack
    stack[0,0,jImg,width,height,jImg+1] = img;
    result("Recorded image "+jImg+" with defocus="+df+"nm\n")
}
// reset the defocus
df = -deltaF*((imgCount+1)/2)
EMChangeFocus(df)

// show the resulting image
showimage(stack)
```

in-place acquisition – saves memory

insert image into stack

Acquisition of EFTEM stack with variable exposure time

Requirements for auto-exposure EFTEM series acquisition:

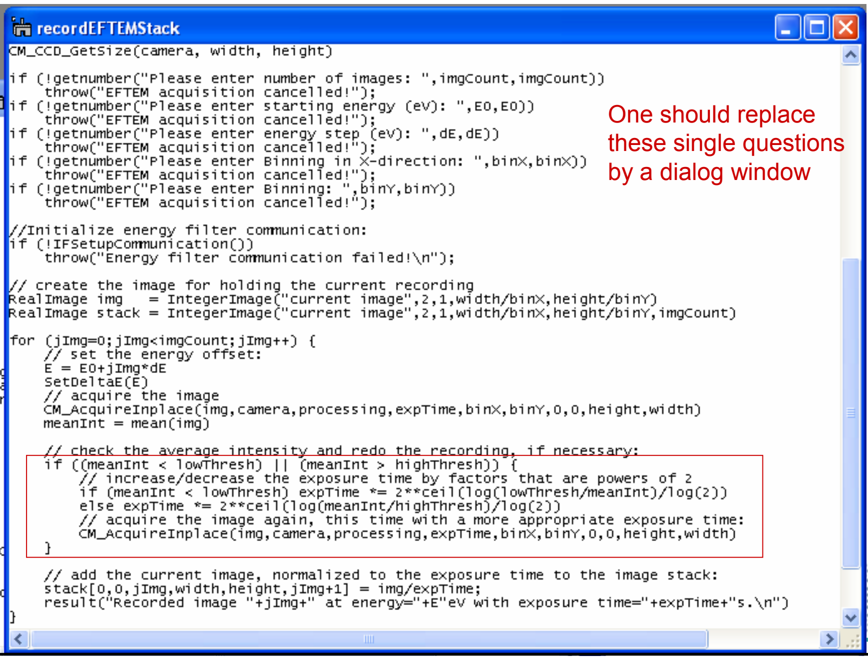
• Only acquire as many dark references as necessary (takes too long otherwise)

• Do not waist memory
  - Use integer arrays
  - Only change the exposure time (needs new dark reference after that), if a lower/upper threshold is exceeded, and then only by factors that are powers of 2.

• Should be easy to use
  -Implement full auto-exposure, for all energies

• EFTEM series should be directly interpretable
  -Scale all images by the exposure time factor relative to the first image.

Acquisition of EFTEM stack with variable exposure time



```
CM_CCD_GetSize(camera, width, height)

if (!getnumber("Please enter number of images: ",imgCount,imgCount))
    throw("EFTEM acquisition cancelled!");
if (!getnumber("Please enter starting energy (eV): ",E0,E0))
    throw("EFTEM acquisition cancelled!");
if (!getnumber("Please enter energy step (eV): ",dE,dE))
    throw("EFTEM acquisition cancelled!");
if (!getnumber("Please enter Binning in X-direction: ",binX,binX))
    throw("EFTEM acquisition cancelled!");
if (!getnumber("Please enter Binning: ",binY,binY))
    throw("EFTEM acquisition cancelled!");

//Initialize energy filter communication:
if (!IFSetupCommunication())
    throw("Energy filter communication failed!\n");

// create the image for holding the current recording
RealImage img   = IntegerImage("current image",2,1,width/binX,height/binY)
RealImage stack = IntegerImage("current image",2,1,width/binX,height/binY,imgCount)

for (jImg=0;jImg<imgCount;jImg++) {
    // set the energy offset:
    E = E0+jImg*dE
    SetDeltaE(E)
    // acquire the image
    CM_AcquireInplace(img,camera,processing,expTime,binX,binY,0,0,height,width)
    meanInt = mean(img)

    // check the average intensity and redo the recording, if necessary:
    if ((meanInt < lowThresh) || (meanInt > highThresh)) {
        // increase/decrease the exposure time by factors that are powers of 2
        if (meanInt < lowThresh) expTime *= 2**ceil(log(lowThresh/meanInt)/log(2))
        else expTime *= 2**ceil(log(meanInt/highThresh)/log(2))
        // acquire the image again, this time with a more appropriate exposure time:
        CM_AcquireInplace(img,camera,processing,expTime,binX,binY,0,0,height,width)
    }

    // add the current image, normalized to the exposure time to the image stack:
    stack[0,0,jImg,width,height,jImg+1] = img/expTime;
    result("Recorded image "+jImg+" at energy="+E"eV with exposure time="+expTime+"s.\n")
}
```

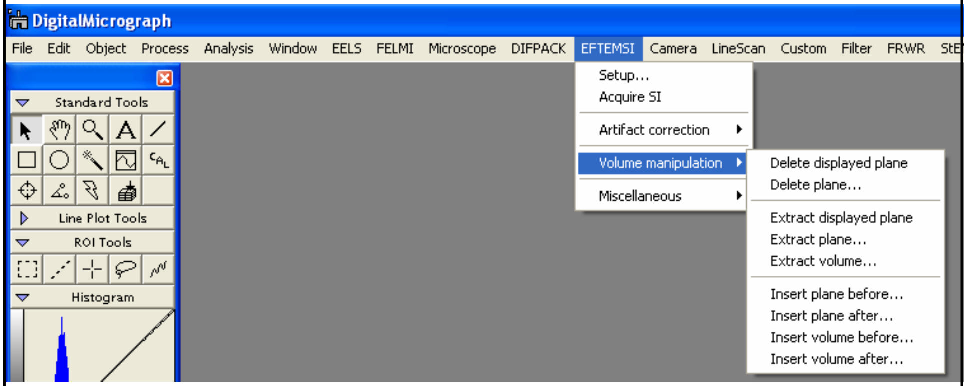
One should replace these single questions by a dialog window

Spectrum Images

- "Spectrum Image" must be of data type "real", single precision
- Spectrum Images have additional tags to be identified by the EELS/EDX plugins. The following commands should make a data stack be recognized by Gatan plugins:
  - `stack.SetStringNote( "Meta Data:Format" , " Spectrum image" )`
  - `stack.SetStringNote( "Meta Data:Signal" , " EELS" )`
- Use script **recordEFTEMStackReal.s** instead. This incorporates these features (and produces files that are twice as large!)

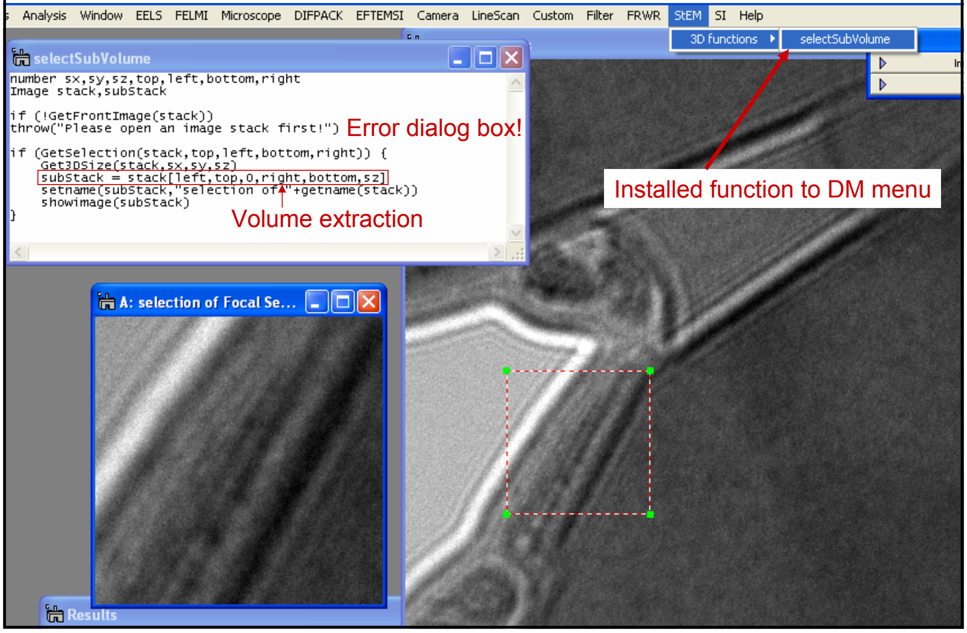
Built-in 3D volume manipulation menu

Menu functions provided by EFTEMSI plugin for manipulating 3D volumes:



Extracting a volume from an image stack

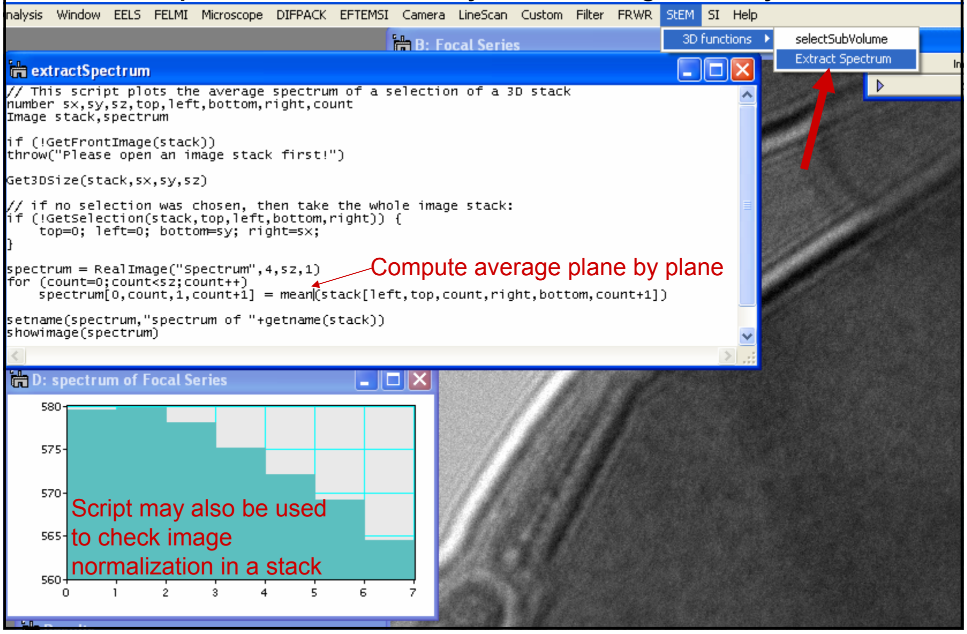
subStack = stack[left,top,firstPlane,right,bottom,lastPlane+1]


Analysis Window EELS FELMI Microscope DIFPACK EFTEMSI Camera LineScan Custom Filter FRWR StEM SI Help
3D functions
selectSubVolume
selectSubVolume
A: selection of Focal Se...
Results
Error dialog box!
Volume extraction
Installed function to DM menu

```
number sx,sy,sz,top,left,bottom,right
Image stack,substack

if (!GetFrontImage(stack))
throw("Please open an image stack first!")

if (GetSelection(stack,top,left,bottom,right)) {
    Get3DSize(stack,sx,sy,sz)
    subStack = stack[left,top,0,right,bottom,sz]
    setname(subStack,"selection of "+getname(stack))
    showimage(subStack)
}
```

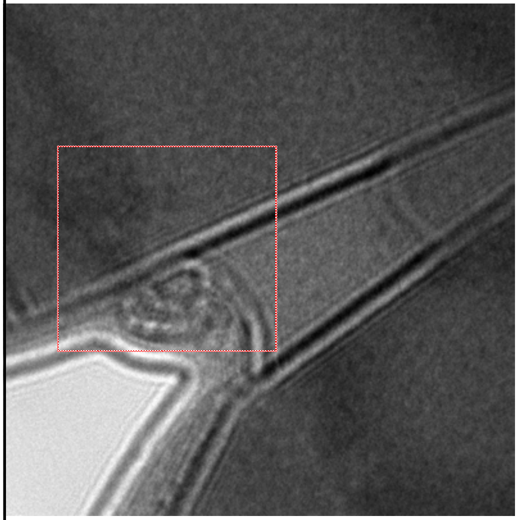
Extracting a spectrum from an image stack

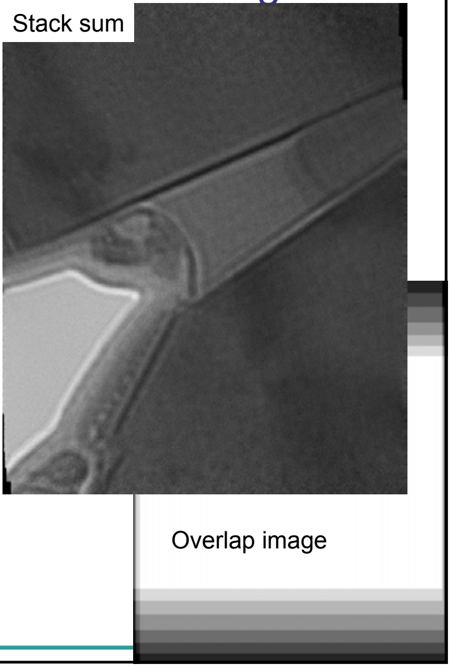
Spectra are horizontal arrays, i.e. their height is always 1!


nalysis Window EELS FELMI Microscope DIFPACK EFTEMSI Camera LineScan Custom Filter FRWR StEM SI Help
B: Focal Series
3D functions
selectSubVolume
Extract Spectrum
extractSpectrum
D: spectrum of Focal Series
Compute average plane by plane
Script may also be used to check image normalization in a stack

```
// This script plots the average spectrum of a selection of a 3D stack
number sx,sy,sz,top,left,bottom,right,count
Image stack,spectrum

if (!GetFrontImage(stack))
throw("Please open an image stack first!")

Get3DSize(stack,sx,sy,sz)

// if no selection was chosen, then take the whole image stack:
if (!GetSelection(stack,top,left,bottom,right)) {
    top=0; left=0; bottom=sy; right=sx;
}

spectrum = RealImage("Spectrum",4,sz,1)
for (count=0;count<sz;count++)
    spectrum[0,count,1,count+1] = mean(stack[left,top,count,right,bottom,count+1])

setname(spectrum,"spectrum of "+getname(stack))
showimage(spectrum)
```

Aligning and adding a stack of images

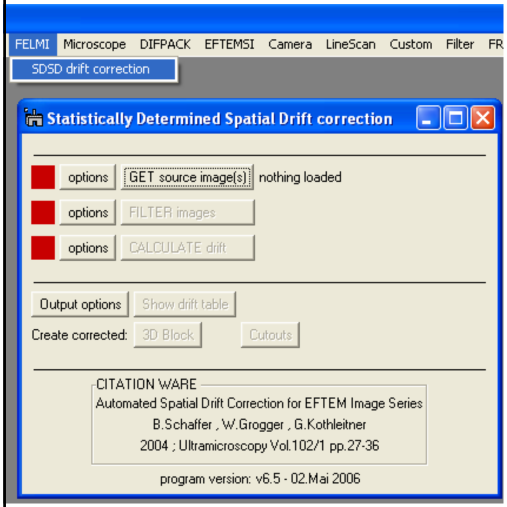




Script: AddAndAlignStack.s

Max-Planck Institut für Metallforschung

Statistically Determined Spatial Drift (SDSD) correction



nothing loaded

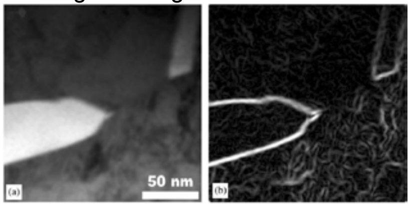
Create corrected:

CITATION WARE
Automated Spatial Drift Correction for EFTEM Image Series
B.Schaffer , W.Grogger , G.Kothleitner
2004 ; Ultramicroscopy Vol.102/1 pp.27-36

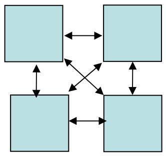
program version: v6.5 - 02.Mai 2006

GUI written by B. Schaffer
(available within StEM)

1. Image filtering



2. Computation drift between all images



N(N-1)/2 cross correlations for N images

3. Drift = weighted average of all drift vectors (weighting factor=cross correlation maximum [0 .. 1])

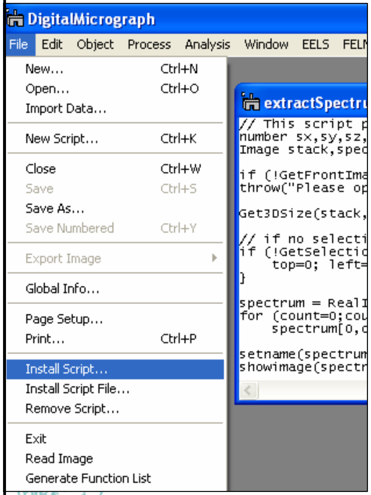
Max-Planck Institut für Metallforschung

Universität Stuttgart

Adding new functions to DM menu

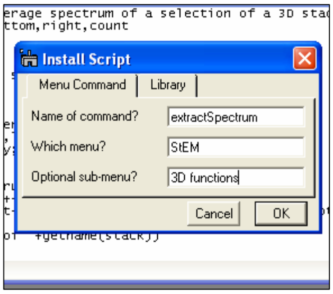
Adding a user script to the DM menu (this will be there permanently, or until deleted):

Step 1



```
// This script p
number sx,sy,sz,
Image stack,spec

if (!GetFrontIma
throw("Please op

Get3DSize(stack,

// if no selecti
if (!GetSelectio
     top=0; left=
}

spectrum = RealI
for (count=0;cou
     spectrum[0,c

setname(spectrum
showimage(spectr
```

Step 2



```
erage spectrum of a selection of a 3D stac
ttom,right,count
```

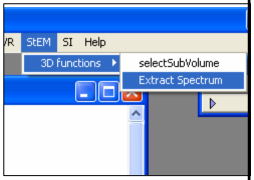
Name of command? extractSpectrum

Which menu? StEM

Optional sub-menu? 3D functions

Just leave
“Optional sub-menu”
empty, if not needed

Done



StEM
SI
Help
3D functions
selectSubVolume
Extract Spectrum