SerialEM HowTo: Macro Explained

Chen Xu

xuchen@brandeis.edu

$BrandeisEM: ~emdoc-xml/en_US.ISO8859-1/articles/SerialEM-howto:macro-explain/article.xml 1 2015-09-18 05:39:04 xuchen Exp$

The look of macro of **SerialEM** is very much like a shell script. It contains macro command line or lines. And running a macro is to execute macro command line by line. The macro commands themselves are high level wrap-up of functions which controls microscope and cameras. To me, the macro commands and macro are among the most attractive features in **SerialEM**. With these, we can controls the scope and camera in very flexible way.

I have tried to learn how to write macro to do my job, mainly for single particle applications. And I continuously learn new things, functions and power of this tool. I like to share what I have learn here. I want to show some examples of the macro I wrote over the past years. I won't be surprising at all if you come up with better macros with clever ideas than I show here. It is my hope that this document can help people new to **SerialEM** to understand how it works and start to write your own macros.

I list some of the macros here. And I want to explain what the macro is to do and how. I welcome any comments and feedback about these macros. This document will be frequently updated, with new or improved macros.

You can also get pdf version of this document here (article.pdf).

All the macros listed in this document be downloaded and loaded to **SerialEM** with a single text tile `macro-list.txt`.

Table of Contents

1 AlignToP

Example 1. AlignToP (AlignToP.txt)

```
MacroName AlignToP
#
# a macro to align to an image in buffer P, twice.
# it takes shots and even crop it when needed. At the end,
# the ImageShift = 0.
# SEM 3-5 beta built 10/10/2014
# update @  2014-11-18 02:09:34

SuppressReports

## Low Dose
ReportLowDose
If $reportedValue1 == 1
    ImageLowDoseSet P
    S = $reportedValue1
Elseif $reportedValue1 == 0
    ImageProperties P
    S = $reportedValue6
Endif

## get image parameters in buffer P
```

```
ImageProperties P
XP = $reportedValue1
YP = $reportedValue2
bin = $reportedValue3
E = $reportedValue4
# PixelSize = $reportedValue5
#S = $reportedValue6

# figure out from which set the image was taken
#(0=view, 1=focus, 2=trial, 3=record, 4=preview)#
If $S == 0
    shot = View
Elseif $S == 1
    shot = Focus
Elseif $S == 2 OR $S == 5
    shot = Trial
Elseif $S == 3
    shot = Record
Elseif $S == 4
    shot = Preview
Endif

## set camera parameters
SetExposure $S $E
SetCenteredSize $S $bin $XP $YP

## align
Loop 2
    $shot
    # Need crop, for Camera like Eagle or Falcon?
    ImageProperties A
    XA = $reportedValue1
    YA = $reportedValue2
    If $XA > $XP OR $YA > $YP
        echo Call CropImageAToP
        Call CropImageAToP
    Endif
    AlignTo P
    ResetImageShift
EndLoop

## reset camera
RestoreCameraSet
```

It might look complicated to you. However, the key part of this macro is actually very simple, as shown below.

```
## align
Loop 2
   $shot
   AlignTo P
   ResetImageShift
```

```
EndLoop
```

It is just to take a shot and align it to the image in buffer P and iterate twice. The image shift value resulted from aligning is then reset and compensated by stage shift. This brings stage to a position the image of which is similar to what in the buffer P. Due to stage movement error, we do it twice to make it more accurate. After this, beam is still straight down, no zigzag, which is good for high resolution imaging.

The rest of the code in the example is to mimic the conditions of image in buffer P: binning, size, exposure time, the camera set (V, F, T, R or L) and even whether in Low Dose mode or not. I even take into account the case that the image in P is cropped one and some camera can not take flexible area as Gatan ones. In this case, the image taken in buffer A is also cropped before being aligned to buffer P.

The goal of this macro is to use a template. Let assume we have picked a point item in a valid map. And this point is close to the center of a hole in Quantifoil or C-flat grid mesh. **SerialEM**'s robust function **Realign** can bring stage to that point. This procedure leaves a small mount ImageShift in the end. If we have a single centered hole image put in buffer P, running macro in above example will "refine" the stage position to the center of current hole accurately and gets rid of the ImageShift on scope.

This macro can also be called from another macro, like the following one in next section.

2 AlignToP-quick

Example 2. AlignToP-quick (AlignToP-quick.txt)

```
MacroName AlignToP-quick
#
# a macro to align to an image in buffer P, only ONCE.
# So this is rough positioning, saving some time.
# it takes single shot and even crop it when needed. At the end,
# the ImageShift != 0.
# SEM 3-5 beta built 10/10/2014
# update @  2014-11-18 02:09:22

Echo ====> Running AlignToP-quick ...

SuppressReports

## Low Dose
ReportLowDose
If $reportedValue1 == 1
    ImageLowDoseSet P
    S = $reportedValue1
ElseIf $reportedValue1 == 0
    ImageProperties P
    S = $reportedValue6
Endif

## get image parameters in buffer P
ImageProperties P
XP = $reportedValue1
YP = $reportedValue2
```

```
bin = $reportedValue3
E = $reportedValue4
# PixelSize = $reportedValue5
#S = $reportedValue6

# figure out from which set the image was taken
#(0=view, 1=focus, 2=trial, 3=record, 4=preview)#
If $S == 0
    shot = View
Elseif $S == 1
    shot = Focus
Elseif $S == 2 OR $S == 5
    shot = Trial
Elseif $S == 3
    shot = Record
Elseif $S == 4
    shot = Preview
Endif

## set camera parameters
SetExposure $S $E
SetCenteredSize $S $bin $XP $YP

## align
Loop 1
$shot
# Need crop, for Camera like Eagle or Falcon?
ImageProperties A
XA = $reportedValue1
YA = $reportedValue2
If $XA > $XP OR $YA > $YP
    echo Call CropImageAToP
    Call CropImageAToP
Endif
AlignTo P
EndLoop

## reset camera
RestoreCameraSet
```

3 ZeroIS

Example 3. ZeroIS (ZeroIS.txt)

```
MacroName ZeroIS
# a macro to to to a new position in current image
#
# update @ 2014-10-22 05:29:51
```

```
Copy A P
Call AlignToP
```

After dragging the image in buffer A to a new location by holding the right mouse, running this small macro will bring the stage to the new location, accurately. This is an easy and quick way to go to a position.

4 CropImageAToP

Example 4. CropImageAToP (CropImageAToP.txt)

```
MacroName CropImageAToP
# crop an image in A based on the dimension in P

# update @ 2014-11-20 12:38:53
#
# Note: the sharp edge might cause trouble in aligning
# need to retest more. Might need to smooth edge pixels

Echo ===> Running CropImageAToP ...

SuppressReports

## get info for A and P
ImageProperties P
XP = $reportedValue1
YP = $reportedValue2

ImageProperties A
XA = $reportedValue1
YA = $reportedValue2

Xdiff = $XA - $XP
Ydiff = $YA - $YP

half_Xdiff = $Xdiff / 2
half_Ydiff = $Ydiff / 2

X0 = $half_Xdiff
X1 = $XP + $half_Xdiff

Y0 = $half_Ydiff
Y1 = $YP + $half_Ydiff

# crop
CropImage A $X0 $X1 $Y0 $Y1
```

This macro is to crop an image in buffer A - the latest taken one to the same size of the image in buffer P. This can be useful when the camera size can not be flexible such as FEI's Eagle and Falcon cameras.

5 Z_byG

Example 5. Z_byG (Z_byG.txt)

```
MacroName Z_byG
##############################
# Z_byG.txt
# Updated @ 2014-10-21 12:22:49
##############################
#
# a macro to adjust the eccentric center using beam tilted pairs.
# It uses Autofocus to measure the focus and adjust Z instead.
#

Echo ===> Running Z_byG ...

# If in Low Dose, the area should be at Focus first
# before defocus is zeroed, because it could come from V with
# large defocus offset. It could float up in that case.

SuppressReports
ReportLowDose
# if in LowDose and area is V
If $reportedValue1 == 1 AND $reportedValue2 == 0
  GoToLowDoseArea F
Endif

#==================
# set objective lens
#==================
SetEucentricFocus
#NormalizeLenses 2
#Delay 1

#===========
# Adjust Z
#===========
Loop 2
Autofocus -1
ReportAutofocus
t = -1 * $reportedValue1
MoveStage 0 0 $t
echo --> Z moved $reportedValue1 micron
EndLoop

#=== end ====
```

This macro is to adjust specimen height to eucentricity by using tilting beam method.

Normally, we use **SerialEM**'s built-in function **Eucentric Rough** and **Eucentric Fine** for this task. These built-in functions cross-correlate image pairs from symetric stage tilts. As long as correlation works fine, the functions work reliably and give very accurate results. It also outputs the lateral displacement for this specific stage and holder. They are indeed very good.

However, this procedure takes significant mount of time because it takes quite a few images at various tilting angles. After tilting to an angle, it needs to wait some time for the stage to settle before a stable image can be taken. The stage tilting most likely generates bubbling for the already quiet LN2 in a sided-entry cryo holder like Gatan's 626. The fact is when objective lens is at Eucentric Focus, for specimen at eucentricity, the image pair from the beam tilting will overlap. Therefore, if we put objective at calibrated Eucentric Focus, and we can measure the defocus and adjust specimen Z height. This is opposite to how we do **Autofocus**.

The macro command **SetEucentricFocus** in the macro above is to recall the calibrated value and set objective lens to Eucentric Focus accordingly. The defocus is measured using **Autofocus** routine. But we ask it to only measure the defocus, not to change actual defocus. This is accomplished by the argument "-1" in the line

```
Autofocus -1
```

The measured defocus value is applied to stage move, in opposite direction. Again, due to intrinsic error of stage movement, we have to iterate to get it accurate enough. Here, we loop twice.

Normalizing objective lens is also for accuracy purpose, as one can see.

If the calibration is good, running this macro at a magnification will bring stage to Eucentricity, fairly accurate.

6 Z_byV

Example 6. Z_byV (Z_byV.txt)

```
MacroName Z_byV
##############################
# Z_byV.txt
# by Chen Xu, Oct  23, 2010,
# Last modified: @ 2014-10-21 12:26:18
##############################
#

Echo ===> Running Z_byV ...

#===================================
# for defocus offset of V in Low Dose, save it
# ===================================
GoToLowDoseArea V
SaveFocus

#==================
# set object lens
#==================
SetEucentricFocus
#NormalizeLenses 2
#Delay 1

#===========
# Adjust Z
#===========
Loop 2
```

```
Autofocus -1 1
ReportAutofocus
Z = -1 * $reportedValue1
MoveStage 0 0 $Z
echo Z has moved --> $Z micron
EndLoop

#=======================================
# restore the defocus set in V originally
# =======================================
RestoreFocus
```

This macro is similar to the macro **Z_byG**, but for in Low Dose mode and using View area. If there is a defocus offset between V and R, then we need to save the defocus value for *View* area and restore it afterwards. This is accomplished by two macro commands **SaveFocus** and **RestoreFocus**.

Interestingly, I ran into problem when I stopped macro before it reaches **RestoreFocus**. The defocus value for *View* area got forgotten. Fortunately, internal mechanism of macro control takes care of this. As long as **SaveFocus** is executed, the defocus value will be restored even the macro is interrupted.

The command line

```
Autofocus -1 1
```

means to measure the defocus using V. The last argument 1 is for View area in Low Dose mode. This becomes very convenient, as we don't have to leave Low Dose mode.

Surprisingly, the accuracy of this method is rather good. Even at relatively low mag like in M range, it can be as accurate as within 1 micron or so.

> **Note:** Depending on the stage, it might need to iterate more than twice. I have found that on our F30 which has older CompuStage it needs to loop 3 times to get it real close - less than half macron. However, on F20 which has newer CompuStage it only needs twice.

7 CalEuentricFocus

Example 7. CalEuentricFocus (CalEucentricFocus.txt)

```
MacroName CalEucentricFocus
# macro to calibrate EucentriFocus using F
# update @ 2014-10-22 07:55:56

Echo ===> Running CalEucentricFocus ...

# use SEM built-in wobbling way to get to Eucentricity
Eucentricity 3

## record original TargetDefocus
ReportTargetDefocus
```

```
orig = $reportedValue1

## make sure TargetDefocus is at 0
SetTargetDefocus 0

## bring Objective to eucentric
## not to normalize because beam will be
## drifting afterwards.

#NormalizeLenses 2
#Delay 1
Loop 3
       G
EndLoop

## Cal - record the value in memory, unless in admin mode
## and save it to calibration file.
CalEucentricFocus

## restore original target defocus
SetTargetDefocus $orig
```

As we see from macro **Z_byG** and **Z_byV**, it is key to have EucentriFocus calibrated accurately for these two macros to work accurately.

Normally, if we calibrate this EucentriFocus manually for a specific magnification, we follow the procedure as following.

1. Make sure the specimen is at Eucentricity or we call Eucentric Height. We can run **Eucentric Both** on a specimen area with good signal and clear features. It is useful to double check how it is when the routine finishes. One can take shot and tilt stage an angle like -45 degree and take another shot. Comparing the two shots gives idea if the sample is truly at Eucentricity.
2. Set Target Defocus to 0.
3. Autofocus for a few times, until the defocus is adjusted to very close to focus. It is recommended to normalize Objective Lens before Autofocus.
4. From menu, Calibration $\longrightarrow$ Focus & AutoFocus $\longrightarrow$ Standard Focus.

These steps can be done with a single macro as above.

Note: this is to be done at each magnification that will be used for this purpose. Administrator perhaps has done for most of the magnifications. Therefore, the calibration file has these values already. However, as you might use a slightly different scope alignment from what the calibration was done with, it is the best to do this calibration at beginning of your session as an user. This macro is aim to make this task easy.

8 CalEucentricFocus_View

Example 8. CalEucentricFocus_View (CalEucentricFocus_View.txt)

```
MacroName CalEucentricFocus_View
```

```
# macro to quickly calbrate EucentricFocus value using View in LD.
# update @ 2014-10-21 12:28:58

Echo ===> Running CalEucentricFocus_View ...

# use SEM built-in wobbling way to get to Eucentricity
Eucentricity 3

## record original TargetDefocus
ReportTargetDefocus
orig = $reportedValue1

## make sure TargetDefocus is at 0
SetTargetDefocus 0

## take care of LD offset
GoToLowDoseArea V
SaveFocus
SetStandardFocus 0

## Bring objective to Eucentric Focus. Not use normalize
## because it caused beam to shift (not stable)
#NormalizeLenses 2
#Delay 1
Loop 3
# use View in LD
       G 1 1
EndLoop
CalEucentricFocus

## Restore Focus
RestoreFocus

## restore original target defocus
SetTargetDefocus $orig
```

Same as before, it is even more convenient to do this calibration in Low Dose mode with *View* area, as it can stay in Low Dose mode and one calibration for *View* mag would be sufficient.

As you can see, we need to "forget" the likely huge defocus offset for *View*, and bring it back afterwards. Otherwise, it could take long time for Objective to get close to Eucentric Focus.

It is interesting to see the line

```
G 1 1
```

in the macro. It means to Autofocus using *View*.

9 Drift

Example 9. Drift (Drift.txt)

```
MacroName Drift
######################################################################
# Drift.txt
# by Chen Xu at 10/11/2011
# Update @ 2014-10-22 10:33:17
# Modified from the original code by David M
######################################################################
#
# A macro to measure drift rate, if good, skip to the end of loop.
# Otherwise, exit execution -- i.e. skip the point.

Echo ===> Running Drift ...

#=============
# parameters
#=============
## drift rate threshold
# crit = 2
# set these above from centralized macro - Parameters
Call Parameters
shot = F
interval = 15
times = 10

period = $interval + 1
#SuppressReports
#ResetClock

$shot
Delay $interval
Loop $times index
$shot
AlignTo B
ReportAlignShift
ClearAlignment
dx = $reportedValue3
dy = $reportedValue4
dist = sqrt $dx * $dx + $dy * $dy
rate = $dist / $period * 10
#ReportClock
echo Rate = $rate A/sec
echo ----------------

If $rate < $crit
    echo Drift is low enough after shot $index
    break
Elseif  $index < $times
    Delay $interval
```

```
Else
    echo Drift never got below $crit: Skipping ...
    exit
Endif
EndLoop
```

This macro is modified from David's original macro "drift control". I made it to output drifting rate as angstrom instead of nanometer. And I use "exit" in the loop.

This macro can be used as standalone or be called from main macro. It measures drifting rate, when the threshold is reached, it jumps to the end of this macro using "break". If the threshold can be reached for number of times, it exits this macro to end or if being called from main macro it exits to the end of the main macro - skip this point.

This macro is to be called right before final exposure. It is also useful with direct detector camera. We use higher threshold to avoid initial drifting is too big, even with movie mode.

10 CycleTargetDefocus

Example 10. CycleTargetDefocus (CycleTargetDefocus.txt)

```
MacroName CycleTargetDefocus
#
# by Chen Xu, update @ 2014-10-21 12:12:55
#
# change Target Defocus on the fly
#
##########################################
# Note: need 3.5 beta
##########################################

# define defocus up and down thresholds, and changing step
# set here or by calling Parameters
#TD_low = -1.2
#TD_high = -2.0
Call Parameters

step = 0.1

Echo ===> Running CycleTargetDefocus ...
Echo >>>> defined Range and Step (um)  => [ $TD_low, $TD_high ], [ $step ].

######### no editing below ###############
delta = -1 * $step

# if the current TargetDefocus is outside of range, make it as TD_low.
# Otherwise, increase a step of $delta

SuppressReports
ReportTargetDefocus
If $reportedValue1 > $TD_low OR $reportedValue1 < $TD_high
   SetTargetDefocus $TD_low
```

```
Else
   IncTargetDefocus $delta
Endif

ReportTargetDefocus
Echo TargetDefocus = $reportedValue1 um
```

When we take many images for single particle application, we usually take them in slightly different defocus value, but within certain range. This macro is to be used right before **Autofocus** command in a macro. It checks current TargetDefocus value, if it is still inside the range, increases limit value. It cycles TargetDefocus on the fly.

I also took care of the case that current TargetDefocus as starting is not inside the range. In this case, I let it to be the middle of the range. Therefore, we don't need to care about what TargetDefocus set in the interface.

11 LD

Example 11. LD (LD.txt)

```
MacroName LD
# main macro for single particle data collection in Low Dose Mode
# update @ 2014-10-23 08:01:15

Echo ===> Running LD ...

Call Parameters

ResetClock

## position (X,Y)
RealignToNavItem 1
Copy A P
Call AlignToP

## Center Beam, assume AutoCenterBeam policy has been set up
AutoCenterBeam

## CycleTargetFocus, range is set in macro "Parameters".
Call CycleTargetFocus

## Autofocus, twice to be more accurate
G
G

## Drift control
# drift control or not is set in Parameters
# with skip set value, skip = 1 -> no drift control.
Loop 1
If $skip == 1
    continue
Else
    Call Drift
```

```
Endif
EndLoop

# final shots, uncomment the next line for K2 camera
Loop 1
If $k2 == 1
    EarlyReturnNextShot $no_return
Else
    break
Endif
EndLoop

R

## save return image or not
Loop 1
if $save == 1
   S
Else
   echo R image in SerialEM window is not saved ...
Endif
EndLoop

ReportClock
echo ----------------------------------------------------------
```

This is the *Main* macro to run for single particle application. It usually runs from menu Navigator ⟶ Acquire at points... and as major action. Thus, all the navigator items with acquiring flag "A" will run this macro, one by one.

This macro should not be hard to understand. It does a few things as we normally do: go to a place, focus and take final image. The line

```
RealignToNavItem 1
```

is to get current nav item to the center position using **Realign** built-in routine, possibly with small mount of ImageShift left. The argument "1" means to return to the initial image state after the routine is done. In this case of being in Low Dose, this means to get out of Low Dose to do **Realign** and return to Low Dose again it finishes. We use our macro **ZeroIS** to clear out the ImageShift, as explained earlier.

We use use **Autofocus** twice here. This is because if it is little further away from target focus, only once might not get close enough. Two rounds are usually sufficient. And centering beam is needed to make sure beam stays centered all the time, because beam could wander away with time.

Needless to say here, all this is on the base that the Low Dose has been setup right.

12 LD-Modulo

Example 12. LD-Modulo (LD-Modulo.txt)

```
MacroName LD-Modulo
# Macro to run from Acquire at points...
# it skips some point for preparing step just take shot
```

```
# last update @ 2014-10-23 02:30:54

ResetClock
echo ===> Running LD-Modulo ...

## define the next line, or by calling "Parameters"
# point to do things (5,10,15...), skip between
# and do Z_byV at double space
# pointXY = 5
# pointZ = 12
# pointZ = 0
Call Parameters

Echo *****
Echo ***** Check Beam, Defocus etc. at every $pointXY_th point ...
Echo ***** Check Eucentricity at every $pointZ_th point...
Echo *****
############ no edit below ################

# find index number for current acquire
ReportNavItem
echo Acquiring Item is $navAcqIndex
remainderXY = modulo $navAcqIndex $pointXY
Loop 1
If $pointZ == 0
    remainderZ = 9999
Else
    remainderZ = modulo $navAcqIndex $pointZ
Endif

## very first one, force two round realigning
If $navAcqIndex == 1
    ForceCenterRealign
    echo -> Very first "A" item, ForceCenterRealign
Endif

# find GroupStatus for current item. It is 1 for first
# acquire item in the group.
# It is recommended to have a new group for each mesh.
ReportGroupStatus
GS = $reportedValue1
EndLoop

## now a few actions after get to the point
Loop 1
# realign to that item first
RealignToNavItem 1

# what to do after realign and before Z change
# which type to do. 0 for precise, 1 for quick, 2 for template
# in buffer P already
If $type == 0 OR $type == 1
    Copy A P
```

```
Elseif $type == 2
    Echo --> type = $type -- A template is assumed in buffer P, aligned to precisely
Elseif $type == 3
    Echo ---> Type = $type -- align to template once, rough positioning
Else
    Echo ---> Type = $type is not defined, exit... You should End Acquire
    exit
Endif


# if pointZ = 0, skip Z chang i.e., not to do any Z_byV or Z_byG
If $remainderZ == 9999
    Echo --> Not doing Z Change,  as pointZ is set to 0.
Elseif $remainderZ == 0 OR $GS == 1
    Call Z_byV
    UpdateGroupZ
Endif

# take care of XY postion after possible Z change
If $type == 0 OR $type == 2
    echo --> Type = $type (0:precisely, 2:template)
    Call AlignToP
Elseif $type == 1
    echo --> Type = $type (use ZeroIS-quick - rough positioning)
    Call ZeroIS-quick
Elseif $type == 3
    echo --> Type = $type (use template to align one round)
    Call AlignToP-quick
    Call ZeroIS-quick
Else
    echo --> Type = $type is not defined, exit... You should End Acquire
    exit
Endif

# skip XY, based on remainderXY
If $remainderXY == 0 OR $GS == 1
    AutoCenterBeam
    Call CycleTargetDefocus
   ## If focus change too much, abort and use original
   ## need latest beta
    FocusChangeLimits -10 10
    G
    G
Else
    Continue
Endif
Endloop

## drift control or not is set in Parameters
## with "skip" set value, 1 will skip.
Loop 1
If $skip == 1
    continue
```

```
Else
    Call Drift
Endif
EndLoop

# final shots
Loop 1
If $k2 == 1
    echo => This is a k2 camera.
    echo => EarlyReturnNextShot $no_return
    EarlyReturnNextShot $no_return
Else
    continue
Endif
EndLoop

R

## save return image or not
Loop 1
if save == 1
   S
Else
   echo R image in SerialEM window is not saved ...
Endif
EndLoop

## in case the objective went off too much
SetStandardFocus 0

#clock
ReportClock
Echo $reportedValue1 seconds!

Echo ---------------------------------------------------------------------
```

The idea behind this macro is to skip some points for any of the preparing tasks, but just to take a shot there.

For all the numbers defined by "point" in the beginning part of macro and very first acquired point in a group, I ask the program to perform thses actions as below.

1. Realign to that point.
2. Adjust current specimen location to Eucentricity.
3. Update all the points in the same group to the same Z.
4. "Refine" X,Y position to correct the lateral shift caused by Z change.
5. Center the beam.
6. Change TargetDefocus by a step.
7. Autodefous, twice.
8. wait for drift to settle down.

9. Take the final shot, and save it.

For all the other points between, it just "realign"s to that position, and take the shot.

13 LD-Group

Example 13. LD-Group (LD-Group.txt)

```
MacroName LD-Group

# macro to skip points except the very first in the group.
# assume LD is setup.

# X,Y position
RealignToNavItem 1
Copy A P
# ReadFile hole.st P

# preparation for first item in group
ReportGroupStatus
If $repVal1 == 1 OR $repVal1 == 0
    Call Z_byV
    UpdateGroup Z
    CallFunction MyFuncs::AlignToP 2
    #CallFunction MyFuncs::ZeroIS-quick 0.3 3.0
    AutoCenterBeam
    CallFunction MyFuncs::CycleTargetDefocus -1.2 -2.0 0.2
    G
    G
Else
    CallFunction MyFuncs::AlignToP 2
    #CallFunction MyFuncs::ZeroIS-quick 0.3 3.0
Endif

# for other in group, clear out shift and take shot
# EarlyReturnNextShot 0
R
S
```

If we draw group of points in patch fashion, then it makes sense to only do some tasks for very first point item in the group. It appears that this is smarter than LD-Modulo.

14 TiltPair

Example 14. TiltPair (TiltPair.txt)

```
MacroName TiltPair
################################
# TiltPair.txt
# update @ 2014-10-21 12:21:06
################################
#
# A main macro for Tilting image pairs.
# work for any two angles

Echo ===> Running TiltPair ...
ResetClock

# set angles
ang_1 = 0
ang_2 = 45

# Make sure it starts with 0, in case it was stopped at
# some angle from last round
TiltTo 0

# collect tilting pairs
Loop 2 i
if $i == 1
   ang = $ang_1
else
   ang = $ang_2
endif

RealignToNavItem 1
WalkUpTo $ang
Call ZeroIS
AutoCenterBeam
G
G
Call Drift
R
S
EndLoop

# back to 0
TiltTo 0

ReportClock
echo ------------------------------------------------------------
```

Collecting tilting pair at any angles is only to set the two angles in the macro. I think this demonstrates how flexible **SerialEM** is.

15 ZeroIS-quick

Example 15. ZeroIS-quick (ZeroIS-quick.txt)

```
MacroName ZeroIS-quick
# macro to clears out IS by either using
# Clearalignment or Resetimageshift, depending on
# the limit set. And if the left over IS from Realign
# is too big, skip this point, likely someting is wrong.

# 2014-10-21 12:05:23

Echo ===> Running ZeroIS-quick ...

# position off limit in micron, now set in macro "Parameters"
limit = 0.3
#Call Parameters

dead = 3.0

############### no editing below ##################

# IS from Realign routine
ReportImageShift
X = $reportedValue1
Y = $reportedValue2
IS = sqrt $X * $X + $Y * $Y

Loop 1
If $IS <= $limit
    ClearAlignment
    echo IS ($IS) <= limit ($limit) um
    echo Clearalignment
Elseif $IS > $limit AND $IS <= $dead
    echo IS ($IS) > limit ($limit) um
    echo Resetimageshift
    ResetImageShift
Else
    Echo IS ($IS) > $dead um, skip this point!
    exit
Endif
EndLoop
```

The major difference of this one from **ZeroIS** is that this one doesn't take any shot. The price paid for the speed is that the position won't be quite at original where it's picked. For small size camera on relatively larger hole, this might be fine. But if you want to be precisely on the target position as you picked, you better off using **ZeroIS** instead.

16 PrepTS

Example 16. PrepTS (PrepTS.txt)

```
MacroName PrepTS
# macro to prepare for Tilting Series
# It brings to each of the picked item and
# refine eucentricity using tilting beam method.
# This is used from Acquire at points... as pre-action macro.
#
# It is similar to "refine and realign", but without wobbling
# the stage
#
# update @ 2014-10-22 08:21:20

Echo ===> Running PrepTS ...

# realign to
RealignToNavItem 1

# copy last image to buffer P
Copy A P

# not adjust Z height, in LD with View area
Call Z_byV

# refine position, incase the lateral displacement
# is siginificant after Z height change
Call AlignToP
```

If **Z_byV** is tested working, this macro can save some time. This works the best with bi-directional method for tilting series.

17 Parameters

Example 17. Parameters (Parameters.txt)

```
MacroName Parameters
# macro to include all the parameters used in all the macros
# It should be called in the beginning of any main macro

# 2014-10-21 12:08:06

Echo ---> calling Parameters ...

## for LD-Modulo
# 1) type to do
# type = 0 -> Precisely to the picked point
# type = 1 -> Use ZeroIS-quick, i.e. rough positioning
# type = 2 -> Align to template in P twice, precisely
```

```
# type = 3 -> Align to template once, rough positioning
type = 3

# 2) points to do things. Skip between those points.
# E.g. pointXY = 7, pointZ = 14, do XY at 7th, and Z at 14th
# E.g. pointZ = 0 -> not do Z change at all
# E.g. pointXY = pointZ = 1, do everything at everything point
#      i.e. not to skip any point.
pointXY = 1
pointZ = 1

## for CycleTargetDefocus (um), default step is 0.1(um)
TD_low = -1.2
TD_high = -2.2

## for Drift
# skip = 1 -> no drift control
# skip = 0 -> drift control
# drift rate threshold (A/sec.), only get used for skip = 0
skip = 1
crit = 5

## K2 special, is this a K2 camera. 1 for yes, 0 for no.
# this is for no return for Record Frame exposure for a K2 camera.
# how many first frames to return sum, only get used for k2 = 1
k2 = 0
no_return = 0

## for movie collection on K2 or Falcon Hack, return sum to SEM might
## not need to be saved. Set "save" to 1 to save or 0 not to save.
save = 1
```

This is the macro only contains some variables for various macros. The purpose of this macro is to set all the needed parameters in this single macro. There is bascially no need to edit all the other macros.

18 Functions

Example 18. MyFuncs (MyFuncs.txt)

```
MacroName MyFuncs

## functions which can be called from a macro

## or a function.



########################################

Function CycleTargetDefocus 3 0
```

```
## three variables, not string


# define defocus up and down thresholds, and changing step

# set here or by calling Parameters

TD_low = $argVal1

TD_high = $argVal2

step = $argVal3


#Call Parameters


Echo ===> Running CycleTargetDefocus ...

Echo >>>> defined Range and Step (um)  => [ $TD_low, $TD_high ], [ $step ].


######### no editing below ###############

delta = -1 * $step


# if the current TargetDefocus is outside of range, make it as TD_low.

# Otherwise, increase a step of $delta


SuppressReports

ReportTargetDefocus

If $reportedValue1 > $TD_low OR $reportedValue1 < $TD_high

   SetTargetDefocus $TD_low

Else

   IncTargetDefocus $delta

Endif
```

```
ReportTargetDefocus

TargetDefocus = $repVal1

Echo TargetDefocus = $repVal1 um

EndFunction


################################

Function CropImageAToP 0 0

Echo ===> Running CropImageAToP ...


#SupressReports


## get info for A and P

ImageProperties P

XP = $reportedValue1

YP = $reportedValue2


ImageProperties A

XA = $reportedValue1

YA = $reportedValue2


Xdiff = $XA - $XP

Ydiff = $YA - $YP


half_Xdiff = $Xdiff / 2

half_Ydiff = $Ydiff / 2
```

```
X0 = $half_Xdiff

X1 = $XP + $half_Xdiff


Y0 = $half_Ydiff

Y1 = $YP + $half_Ydiff


# crop

CropImage A $X0 $X1 $Y0 $Y1

EndFunction


######################################################

Function CropImage 1 1

## crop image in A buffer to quarter size

## CallFunction MyFuncs::CropImage A 0.25

buffer = $argVal1

frac = $argVal2


ImageProperties $buffer

X = $repVal1

Y = $repVal2


halfX = $X / 2

halfY = $Y / 2


QX = $X * $frac

QY = $Y * $frac
```

```
echo $QX $QY


X0 = $halfX - ( $QX / 2 )

X1 = $halfX + ( $QX / 2 )

Y0 = $halfY - ( $QY / 2 )

Y1 = $halfY + ( $QY / 2 ) - 1


# crop

CropImage $buffer $X0 $X1 $Y0 $Y1

EndFunction


#####################################################

Function AlignToP 1 0

# align round

# a function to align to an image against buffer P, $iter times.

# it takes shots and even crop it when needed. At the end,

# the ImageShift = 0.

# SEM 3-5 beta built 10/10/2014

# update @  2014-11-18 02:09:34


iter = $argVal1


SuppressReports


## Low Dose

ReportLowDose
```

```
If $reportedValue1 == 1
    ImageLowDoseSet P
    S = $reportedValue1
Elseif $reportedValue1 == 0
    ImageProperties P
    S = $reportedValue6
Endif

## get image parameters in buffer P
ImageProperties P
XP = $reportedValue1
YP = $reportedValue2
bin = $reportedValue3
E = $reportedValue4
# PixelSize = $reportedValue5
#S = $reportedValue6

# figure out from which set the image was taken
#(0=view, 1=focus, 2=trial, 3=record, 4=preview)#
If $S == 0
    shot = View
Elseif $S == 1
    shot = Focus
Elseif $S == 2 OR $S == 5
    shot = Trial
Elseif $S == 3
```

```
    shot = Record
Elseif $S == 4
    shot = Preview
Endif

## set camera parameters
SetExposure $S $E
SetCenteredSize $S $bin $XP $YP

## align
Loop $iter
    $shot
    # Need crop, for Camera like Eagle or Falcon?
    ImageProperties A
    XA = $reportedValue1
    YA = $reportedValue2
    If $XA > $XP OR $YA > $YP
        echo CallFunction  MyFuncc::CropImageAToP
        CallFunction  MyFuncs::CropImageAToP
    Endif
    AlignTo P
    ResetImageShift
EndLoop

## reset camera
RestoreCameraSet
EndFunction
```

```
#########################################################

Function ZeroIS-quick 2 0

# two arguments are -  shift threshold and dead shift (too large)


# 2014-10-21 12:05:23


Echo ===> Running ZeroIS-quick ...


limit = $argVal1

dead = $argVal2


#- no editing below -#


# IS from Realign routine

ReportImageShift

X = $reportedValue1

Y = $reportedValue2

IS = sqrt $X * $X + $Y * $Y


If $IS <= $limit

    ClearAlignment

    echo IS ($IS) <= limit ($limit) um

    echo Clearalignment

Elseif $IS > $limit AND $IS <= $dead

    echo IS ($IS) > limit ($limit) um
```

```
    echo Resetimageshift

    ResetImageShift

Else

    Echo IS ($IS) > $dead um, skip this point!

Endif

EndFunction


################################################

Function FixBeamCenter 0 0

ReportLowDose

If $repVal1 == 0

     ChangeMag -5

     T

    CenterBeamFromImage

    ChangeMag +5

ElseIf $repVal1 == 1

     SetLowDoseMode 0

     ChangeMag -5

     T

    CenterBeamFromImage

    SetLowDoseMode 1

Endif

EndFunction


################################################

Function Drift 1 0

# A function to measure drift rate, if good, skip to the end of loop.
```

```
# Otherwise, exit execution -- i.e. skip the point.


Echo ===> Running Drift $argVal1(A)...


#=============

# parameters

#=============

## drift rate threshold

crit = $argVal1

# set these above from centralized macro - Parameters

shot = F

interval = 15

times = 10


period = $interval + 1

#SuppressReports

#ResetClock


$shot

Delay $interval

Loop $times index

$shot

AlignTo B

ReportAlignShift

ClearAlignment

dx = $reportedValue3
```

```
dy = $reportedValue4

dist = sqrt $dx * $dx + $dy * $dy

rate = $dist / $period * 10

#ReportClock

echo Rate = $rate A/sec

echo ----------------


If $rate < $crit

    echo Drift is low enough after shot $index

    break

Elseif  $index < $times

    Delay $interval

Else

    echo Drift never got below $crit: Skipping ...

    exit

Endif

EndLoop

EndFunction


#######################################################

Function CalEucFocus 0 0

# macro to calibrate EucentriFocus using F

# update @ 2014-10-22 07:55:56


Echo ===> Running CalEucFocus ...

Echo *** Assuming specimen is already at Eucentricitt ***
```

```
# use SEM built-in wobbling way to get to Eucentricity

#Eucentricity 3


## record original TargetDefocus

ReportTargetDefocus

orig = $reportedValue1


## make sure TargetDefocus is at 0

SetTargetDefocus 0


## bring Objective to eucentric

## not to normalize because beam will be

## drifting afterwards.


#NormalizeLenses 2

#Delay 1

Loop 3

       G

EndLoop


## Cal - record the value in memory, unless in admin mode

## and save it to calibration file.

CalEucentricFocus


## restore original target defocus

SetTargetDefocus $orig
```

```
EndFunction


########################################################################
Function CalEucFocus_View 0 0
# macro to quickly calbrate EucentricFocus value using View in LD.
# update @ 2014-10-21 12:28:58

Echo ===> Running CalEucFocus_View ...
Echo *** Assuming specimen is already at Eucentricity ***

# use SEM built-in wobbling way to get to Eucentricity
#Eucentricity 3

## record original TargetDefocus
ReportTargetDefocus
orig = $reportedValue1

## make sure TargetDefocus is at 0
SetTargetDefocus 0

## take care of LD offset
GoToLowDoseArea V
SaveFocus
SetStandardFocus 0

## Bring objective to Eucentric Focus. Not use normalize
## because it caused beam to shift (not stable)
```

```
#NormalizeLenses 2

#Delay 1

Loop 3

# use View in LD

       G 1 1

EndLoop

CalEucentricFocus


## Restore Focus

RestoreFocus


## restore original target defocus

SetTargetDefocus $orig

EndFunction


#################################################################

Function Z_byG 0 0

#

# a function to adjust the eccentric center using beam tilted pairs.

# It uses Autofocus to measure the focus and adjust Z instead.

#


Echo ===> Running Z_byG ...


# If in Low Dose, the area should be at Focus first

# before defocus is zeroed, because it could come from V with
```

```
# large defocus offset. It could float up in that case.


SuppressReports

ReportLowDose

# if in LowDose and area is V

If $reportedValue1 == 1 AND $reportedValue2 == 0

  GoToLowDoseArea F

Endif


#==================

# set objective lens

#==================

SetEucentricFocus

#NormalizeLenses 2

#Delay 1


#===========

# Adjust Z

#===========

Loop 2

Autofocus -1

ReportAutofocus

t = -1 * $reportedValue1

MoveStage 0 0 $t

echo --> Z moved $reportedValue1 micron

EndLoop

EndFunction
```

```
############################################################

Function Z_byV 0 0


Echo ===> Running Z_byV ...

#====================================

# for defocus offset of V in Low Dose, save it

# ====================================

GoToLowDoseArea V

SaveFocus


#==================

# set object lens

#==================

SetEucentricFocus

#NormalizeLenses 2

#Delay 1


#===========

# Adjust Z

#===========

Loop 2

Autofocus -1 1

ReportAutofocus

Z = -1 * $reportedValue1

MoveStage 0 0 $Z
```

```
echo Z has moved --> $Z micron

EndLoop


#=======================================

# restore the defocus set in V originally

# =======================================

RestoreFocus

EndFunction
```

Some of the library like functions can be put into a single macro. The individual functions can be called from any macro or even within function itself. This is new added feature.