Convert Oxford EBSD patterns in HDF5 for EMSOFT

The following python script converts a list of pattern in the _picRep_ folder with naming conventions “NAME_ID” where ID = Nx*y+x, into an HDF5 format compatible with EMSOFT.

To use it, change the values of the variables in the PARAMETERS section to the one that fits to you.

Note that you will need an HDF5 template file, I usually use one of the HDF5 provided for the EDAX Nickel example on the EMSOFT website:

```
##########################################################################
# Converts a set of Kikuchi pattern to a readable EDAX HDF5 format
# Kikuchi pattern must all be in the same folder (defined by 'picRep'
variable)
# Pattern names must be of the form 'Something_ID', where ID = Nx*y+x
# As all the patterns are stored in RAM for writing, it is possible to make
an HDF5
# of a subdomain of the map
#
# Note: for now, the scrip just fills the Patterns and their coordinates on
the map
# All other fields contain dummy values.
#
# 31/10/2019            Simon Breumier            simon.breumier@mines-
saint-etienne.org
##########################################################################
import h5py
import matplotlib.pyplot as plt
import matplotlib.image as mpimg
from PIL import Image
import numpy as np
import os
from scipy import ndimage
import math
from get_tiff_param import get_pat_pc
from progress.bar import Bar

print('******************** Pattern picture ==> EDAX HDF5 conversion
*********')
####################    P    A    R    A    M    E    T    E    R    S
###########

out = 'MTL_100_bin2' # output file name
picRep = 'GND_100_CP/Pattern' #pattern pic repository path
template_file='EDAX-Ni.h5' #HDF5 template file name
# Carto parameters
Xsize=201 #Map width in px
Ysize=164 #Map height in px
X0 = 1 #X position to start the writing (usefull to make an HDF5 file with a
partition of your map)
```

```
Y0 = 1 #Y position to start the writing
Xmax = 201 #X position to stop the writting
Ymax = 164 #Y position to stop the writting
step=0.003 #Step size in mm

#Pattern dimensions (after binning)
pattHeight = 512
pattWidth = 672

#####################################
tmp = h5py.File(template_file,'r')

f = h5py.File(out+'.h5','w')
for elem in tmp.keys():
    tmp.copy(elem,f)
tmp.close()

#############   H    E    A    D    E    R
del(f[' Manufacturer'])
f.create_dataset(' Manufacturer', data=np.array([b'Oxford']))
head = f['Scan 1']['EBSD']['Header']
print('** Fill Header data')
tmp = f['Scan 1']['EBSD']['Header']['Pattern Height']
tmp[...] = pattHeight
tmp = f['Scan 1']['EBSD']['Header']['Pattern Width']
tmp[...] = pattWidth
tmp = f['Scan 1']['EBSD']['Header']['Sample Tilt']
tmp[...] = 70.
tmp = f['Scan 1']['EBSD']['Header']['Step X']
tmp[...] = step*1000
tmp = f['Scan 1']['EBSD']['Header']['Step Y']
tmp[...] = step*1000
tmp = f['Scan 1']['EBSD']['Header']['Working Distance']
tmp[...] = 16022.
tmp = f['Scan 1']['EBSD']['Header']['nColumns']
tmp[...] = Xmax - X0+1
tmp = f['Scan 1']['EBSD']['Header']['nRows']
tmp[...] = Ymax - Y0+1

#############   E    B    S    D         D    A    T    A
if 'Scan 4' in f.keys():
    del(f['Scan 4'])
if 'Scan 6' in f.keys():
    del(f['Scan 6'])
ebsdDat =f['Scan 1']['EBSD']
print('** Fill EBSD data')
print('listing EBSD patterns...')
listPat = os.listdir(picRep)
if 'Thumbs.db' in listPat: listPat.remove('Thumbs.db')
```

```
lenPat = len(listPat)
print('Filling dummy values...')
dum = np.ones((Xsize-X0)*(Ysize-Y0)+1)
i =0
if 'X Position' in f['Scan 1']['EBSD']['Data'].keys():
    del(f['Scan 1']['EBSD']['Data']['X Position'])
if 'Y Position' in f['Scan 1']['EBSD']['Data'].keys():
    del(f['Scan 1']['EBSD']['Data']['Y Position'])
if 'Pattern' in f['Scan 1']['EBSD']['Data'].keys():
    del(f['Scan 1']['EBSD']['Data']['Pattern'])

dumList = ['CI','Fit','IQ','PRIAS Bottom Strip','PRIAS Center Square','PRIAS
Top Strip','Phase','Phi','Phi1','Phi2','SEM Signal','Valid']

for elem in dumList:
    if elem in f['Scan 1']['EBSD']['Data'].keys():
        del(f['Scan 1']['EBSD']['Data'][elem])
    #f['Scan 1']['EBSD']['Data'].create_dataset(elem,data=dum)
#############
lenTot = (Xmax-X0+1)*(Ymax-Y0+1)
f['Scan 1']['EBSD']['Data'].create_dataset("X
Position",data=np.zeros(lenTot))
f['Scan 1']['EBSD']['Data'].create_dataset("Y
Position",data=np.zeros(lenTot))
f['Scan 1']['EBSD']['Data'].create_dataset("X BEAM",data=np.zeros(lenTot))
f['Scan 1']['EBSD']['Data'].create_dataset("Y BEAM",data=np.zeros(lenTot))
f['Scan
1']['EBSD']['Data'].create_dataset("Pattern",(lenTot,pattHeight,pattWidth),
dtype="u1")

Patval = []
DDs = []
PCXs = []
PCYs = []
i = 0
last = 0
j = 0
limAct = [Ysize/4,Ysize/2,3*Ysize/4,Ysize]
tmp,tmp2,picId = listPat[2].split('_')
tmp = tmp+'_'+tmp2
numZeros = len(picId.split('.')[0])-1
print('Find initial point')

bar = Bar('Processing', max=(Xmax-X0+1)*(Ymax-Y0+1))

for Yact in range(Y0,Ymax+1):
    for Xact in range(X0,Xmax+1):
        picId = (Yact-1)*Xsize+Xact
        picId_str = (numZeros-math.floor(math.log10(picId)))*'0'+str(picId)
        PCX, PCY, lfov, Xbeam, Ybeam, DD =
get_pat_pc(picRep+'/'+tmp+'_'+picId_str+'.tiff')
```

```
        tabID = (Yact-Y0)*(Xmax-X0+1)+Xact-X0
        f['Scan 1']['EBSD']['Data']['X BEAM'][tabID] = Xact-X0
        f['Scan 1']['EBSD']['Data']['Y BEAM'][tabID] = Yact-Y0
        DDs.append(DD*pattWidth/pattHeight)
        PCXs.append(PCX)
        PCYs.append(PCY)
        imActu = mpimg.imread(picRep+'/'+tmp+'_'+picId_str+'.tiff')
        f['Scan 1']['EBSD']['Data']['X Position'][tabID] = (Xact-X0)*step
        f['Scan 1']['EBSD']['Data']['Y Position'][tabID] = (Yact-Y0)*step
        f['Scan 1']['EBSD']['Data']['Pattern'][tabID,:,:] = imActu
        i+=1
        bar.next()

bar.finish()
print('\n saving ...')
f['Scan 1']['EBSD']['Data'].create_dataset("Phase",data=np.ones(((Xmax-
X0+1)*(Ymax-Y0+1))))
f['Scan 1']['EBSD']['Data'].create_dataset("DD",data=np.asarray(DDs))
f['Scan 1']['EBSD']['Data'].create_dataset("PCX",data=np.asarray(PCXs))
f['Scan 1']['EBSD']['Data'].create_dataset("PCY",data=np.asarray(PCYs))
f.close()
print('done!')
print('************************************************')
```