Data Analysis of Recordings (C2)
(Section 4)

1 Analysis of the motion of the wave machine levers

This notebook analyses the motion of the levers of the wave machine which is stimulated at a certain frequency and amplitude. To detect the motion of the levers we use the color spots at the end of the levers and a camera. The camera settings have been adjusted to reveal red or yellow color spots in front of a dark background. The red color channel of the recorded images is then used to obtain the positions with the help of the difference of gaussians blob tracking method by skimage. The results are stored in a data frame which is written to a .CSV file. The amplitude of the motion as compared to the excitation as well as the wavelength is extracted.

[1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from numpy.random import randn
from scipy.stats import norm
#from scipy import misc
import glob
from imageio import imread
import os

from skimage.feature import blob_dog, blob_log, blob_doh
import glob
from scipy.optimize import curve_fit


# inline plotting
%matplotlib inline

# this is just for Macbooks to create higher resolution previews of the plots in␣
 ↪the notebook
%config InlineBackend.figure_format = 'retina'
import av
# the lines below set a number of parameters for plotting, such as label font␣
 ↪size,
# title font size, which you may find useful
plt.rcParams.update({'font.size': 12,
                     'axes.titlesize': 16,
```

```
                      'axes.labelsize': 16,
                      'axes.labelpad': 14,
                      'lines.linewidth': 1,
                      'lines.markersize': 10,
                      'xtick.labelsize' : 16,
                      'ytick.labelsize' : 16,
                      'xtick.top' : True,
                      'xtick.direction' : 'in',
                      'ytick.right' : True,
                      'ytick.direction' : 'in',})
```

1.1 Load images from the directory

[4]:
```
pwd
```

[2]:
```
files=glob.glob('*.mov')
file=files[0]
print('found file '+file)

container = av.open(file)

f=[]
for frame in container.decode(video=0):
    image=np.asarray(frame.to_image())
    f.append(np.array(image[150:500,0:1250,0]>100,dtype=float))

frames=np.array(f)
```

[3]:
```
plt.imshow(frames[15],cmap='gray')
```

1.2 Detect Blobs in each frame

[5]:
```
exists=os.path.isfile('Results.csv')
if exists:
    dff=pd.read_csv('Results.csv',index_col=0)
else:
    fn=0
    dff=pd.DataFrame(columns=('x', 'y', 'particle','frame'))
    for index,frame in enumerate(frames[:]):
        blobs_dog = blob_dog(frame, max_sigma=20, threshold=.5)
        x=blobs_dog[:,1]
        y=blobs_dog[:,0]
        n=len(blobs_dog)
        particle=np.array(range(n))
        fnum=np.ones(n)*fn
        df=pd.DataFrame({'x':x,'y':y,'particle': particle ,'frame': fnum})
```

```
        dff=dff.append(df,ignore_index=True)
        fn=fn+1
    dff.to_csv('Results.csv')
```

1.3 Show an example of tracked blobs and the corresponding image

[6]:
```
n=250
plt.figure(figsize=(10,3))
plt.imshow(frames[n])
plt.plot(dff.loc[(dff["frame"] ==n), "x"].values,dff.loc[(dff["frame"] ==n),␣
 ↪"y"].values,'.')
```

2 Plot all value for the tracked blobs

[6]:
```
plt.figure(figsize=(10.,3))
plt.plot(dff.x.values,dff.y.values,'.',alpha=0.1)
plt.xlabel('position [pixel]')
plt.ylabel('amplitude [pixel]')
plt.show()
```

[8]:
```
plt.figure(figsize=(10,3))
plt.plot(dff.loc[(dff["x"] >1150), "y"].values)
plt.plot(dff.loc[(dff["x"] <80), "y"].values)
plt.xlim(0,100)
plt.xlabel('time [frames]')
plt.ylabel('amplitude [pixel]')
plt.show()
```

2.1 Get the enhancement factor

[9]:
```
amp_out=dff.loc[(dff["x"] <80), "y"].values.max()-dff.loc[(dff["x"] <80), "y"].
 ↪values.min()
amp_in=dff.loc[(dff["x"] >1150), "y"].values.max()-dff.loc[(dff["x"] >1150),␣
 ↪"y"].values.min()
print("gain=",amp_out/amp_in)
```

[10]:
```
file1 = open("gain.txt","a")
file1.write(str(amp_out/amp_in))
file1.close()
```

2.2 Get the wavelength and the wavenumber

[11]:
```
def wave(x, *p):
    A, q, phi,B = p
    return( A*np.cos(q*x+phi)+B)
```

[13]:
```
n=90
plt.figure(figsize=(10,3))
xv=dff.loc[(dff["frame"] ==n), "x"].values
yv=dff.loc[(dff["frame"] ==n), "y"].values
ind=np.argsort(xv,axis=0)

xv=xv[ind]
dx=xv.max()-xv.min()
xpos=(xv-xv.min())*40/dx
yv=yv[ind]

p0 = [100., 0.1, 0.5,150.]
coeff, var_matrix = curve_fit(wave, xpos, yv, p0=p0)
fit = wave(xpos, *coeff)
print('A=',coeff[0]*40/dx,'q=',coeff[1],'lambda=',2*np.pi/coeff[1])
fff=open("wavelength.txt","w")
fff.write(str(coeff[0]*40/dx)+"\n")
fff.write(str(coeff[1])+"\n")
fff.write(str(2*np.pi/coeff[1])+"\n")
fff.close()
plt.axhline(y=np.mean(yv)*40/dx,ls='--',color='k')
plt.plot(xpos,fit*40/dx)
plt.plot(xpos,yv*40/dx,'o',alpha=0.5)
plt.xlabel('x [cm]')
plt.ylabel('y [cm]')
plt.show()
```