#!/usr/bin/env python

# coding: utf-8

# In[16]:

import numpy as np

import imageio as imo

import scipy.ndimage as img

def toBiSinSpace(f,ampx,vx,phix,ampy,vy,phiy):

rows,cols = f.shape

# make a new numpy array depends on rows and columns of first image and amplitude that will be choosen

# rows+(2\*amp) evenly spaced samples, calculated over the interval [ -amplitude, rows+amplitude]

x,y = np.meshgrid(np.linspace(-ampx,cols+ampx,cols+(2\*ampx)),

np.linspace(-ampy,rows+ampy,rows+(2\*ampy)))

# wave function

xs,ys = x+ampx\*np.sin(2\*np.pi\*(y/cols)\*vx + phix),y+ampy\*np.sin(2\*np.pi\*(x/rows)\*vy + phiy)

xs,ys = xs.reshape(-1),ys.reshape(-1)

coords = np.vstack((ys,xs))

#Map the input array to new coordinates by interpolation

g = img.map\_coordinates(f,coords, order = 3)

# change the shape of new imgage

h = g.reshape(rows+2\*ampy,cols+2\*ampx)

return h

f = imo.imread('clock.jpg')

out1\_01=toBiSinSpace(f,0,0,0,128,0.5,0) # Vertical amplitude = 128, frequency = 1/2, phase = 0

imo.imwrite('Picture 1\_01.jpg',out1\_01)

out1\_1=toBiSinSpace(f,0,0,0,128,0.5,np.pi) # Vertical amplitude = 128, frequency = 1/2, phase = pi

imo.imwrite('Picture 1\_1.jpg',out1\_1)

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

out2\_01=toBiSinSpace(f,0,0,0,128,1,0) # Vertical amplitude = 128, frequency = 1, phase = 0

imo.imwrite('Picture 2\_01.jpg',out2\_01)

out2\_1=toBiSinSpace(f,0,0,0,128,1,np.pi/2) # Vertical amplitude = -128, frequency = 1, phase = pi/2

imo.imwrite('Picture 2\_1.jpg',out2\_1)

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

out3\_01 = toBiSinSpace(f,10,2,-np.pi/4,10,2,-np.pi/2) # Both Horizontal and Vertical

imo.imwrite('Picture 3\_01.jpg',out3\_01)

out3\_1 = toBiSinSpace(f,10,2,-np.pi/2,10,2,-np.pi/2) # Both Horizontal and Vertical

imo.imwrite('Picture 3\_1.jpg',out3\_1)

out3\_2 = toBiSinSpace(f,20,2,-np.pi/4,10,2,-np.pi/2) # Both Horizontal and Vertical

imo.imwrite('Picture 3\_2.jpg',out3\_2)

out3\_3 = toBiSinSpace(f,10,4,-np.pi/4,10,2,-np.pi/2) # Both Horizontal and Vertical

imo.imwrite('Picture 3\_3.jpg',out3\_3)

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

# first, Implement vertical

# second, Implement horizontal on the result of frist step

out4\_01=toBiSinSpace(f,0,0,0,6,5,0)

imo.imwrite('picture 4\_01.jpg',out4\_01)

out4\_02=toBiSinSpace(out4\_01,15,5,-np.pi/6,0,0,0)

imo.imwrite('picture 4\_02.jpg',out4\_02)

# just for see the result of little change

out4\_1=toBiSinSpace(f,0,0,0,6,5,0)

imo.imwrite('picture 4\_1.jpg',out4\_1)

out4\_2=toBiSinSpace(out4\_1,10,5,-np.pi/6,0,0,0)

imo.imwrite('picture 4\_2.jpg',out4\_2)

out4\_3=toBiSinSpace(out4\_1,15,7,-np.pi/6,0,0,0)

imo.imwrite('picture 4\_3.jpg',out4\_3)

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

# first, Implement vertical

# second, Implement horizontal on the result of frist step

out5\_01 = toBiSinSpace(f,0,0,0,10,8,0)

imo.imwrite('Picture 5\_01.jpg',out5\_01)

out5\_02 = toBiSinSpace(out5\_01,10,1.15,np.pi\*3/4,0,0,0)

imo.imwrite('Picture 5\_02.jpg',out5\_02)

out5\_1 = toBiSinSpace(f,0,0,0,10,4,0)

imo.imwrite('Picture 5\_1.jpg',out5\_1)

out5\_2 = toBiSinSpace(out5\_1,20,1.15,np.pi\*3/4,0,0,0)

imo.imwrite('Picture 5\_2.jpg',out5\_2)

# In[ ]:

# In[ ]: