```
import matplotlib.pyplot as plt, matplotlib.cm as cm
import proper
import math
import numpy as np
import cv2

#uses proper to simulate the expected images
def generate_images(zernikies, defocus = 0.01):
    zernikies = [z*0.55e-6 for z in zernikies]
    wavelength = 0.55 #um
    pix_size = 5.86e-6
    alpha = 0.1396
    height = defocus*math.sin(alpha)
    beam_ratio = 0.5
    npix = int(height/beam_ratio/pix_size) #number of pixels that the grid sho
uld span
    gridsize = 512 #int(2**(int(math.log(npix, 2)))) #npix to next power of 2
    setting = {'ZERN': zernikies, 'DEFOCUS': defocus,
               'diam': 0.05, 'focal_length': 0.6096, 'beam_ratio': beam_ratio}

    #run the optical simulation
    pre_im, sampling = proper.prop_run( 'prefocal_image', 0.55, gridsize, PASS
VALUE=setting)
    pos_im, sampling = proper.prop_run( 'postfocal_image', 0.55, gridsize, PAS
SVALUE=setting)
    #invert postfocal image
    pos_im = [[pos_im[gridsize - y-1][gridsize - x-1] for x in range(gridsize)
] for y in range(gridsize)]

    pre_im = cv2.resize(np.array(pre_im), (npix, npix))
    pos_im = cv2.resize(np.array(pos_im), (npix, npix))

    return pre_im, pos_im


if __name__ == "__main__":
    zern = [0,0,0.07]

    names = ['1_PISTON',
             '2_X tilt',
             '3_Y tilt',
             '4_Focus',
             '5_45DegAstigmatism',
             '6_0DegAstigmatism',
             '7_YComa',
             '8_XComa',
             '9_YTrefoil',
             '10_XTrefoil',
             '11_Spherical']

    plt.ion()
    plt.show()
    for i in range(11):
        for defocus in [0.1e-3, 0.720e-3, 1e-3, 10e-3][1:2]:
            zern = [0] * 11
            zern[i] = 0.07
            pre_im, pos_im = generate_images(zern, defocus)
            diff = pre_im - pos_im
            plt.subplot(1,3,1)
            plt.title("Prefocal")
            plt.imshow(pre_im, cmap=cm.gray)
            plt.subplot(1,3,2)
            plt.title("Postfocal")
            plt.imshow(pos_im, cmap=cm.gray)
            plt.subplot(1,3,3)
            plt.title("Difference")
            plt.imshow(pre_im - pos_im, cmap=cm.gray)
            plt.draw()
            plt.pause(0.001)
            plt.savefig("plots/" + names[i] + '_{}um'.format(int(defocus*1e6))
 + ".png")
```