chaotic_oscillator_wRoyce_March_2022

March 4, 2022

[3]:

```
import serial.tools.list_ports
def find_port():
    ports = list(serial.tools.list_ports.comports())
    for p in ports:
        print(p)

print(find_port())
```

```
/dev/cu.Bluetooth-Incoming-Port - n/a
/dev/cu.usbmodem58794001 - USB Serial
None
```

[24]:

```
#!/usr/bin/env python

'''
        mco.py
        Eric Ayars
        June 2016

        A simplistic program to allow communication with and data collection␣
 ↪from
        the MCO.

        Special commands are as follows:
                quit - quits the program
                c - collects data. 'c' must be followed by the number of drive␣
 ↪cycles
                        and then the filename in which to save the data.
                        Addition: 'c' also plots a phase space plot of the data␣
 ↪and saves it.
                p - plots last data set. If 'p' is followed by 's' (i.e. 'p s')␣
 ↪then it
                        saves the figure also.

        All other commands are sent directly to the MCO.
        e.g.
                freq?
                ampl 400
                coil 1
                etc.
'''

#from pylab import *
import serial

def writeln(S):
    # encodes a string as bytearray and writes it on serial
    S = S + '\n'
    ser.write(S.encode())

def readln(ser):
    # reads a bytearray from serial and converts it to a string
    BA = ser.readline().decode("utf-8")
    return BA

def getData(N, file):
        # gets N periods worth of data, saves raw data lines to a file.

        N = 256*N
        print("Saving %d data points to file '%s'." % (N, file))

        # get useful data from the MCO
        writeln('freq?')
        freq = 0 #float(readln(ser))
        writeln('ampl?')
        ampl = int(0) #int(readln(ser))

        # open the file and write the header
        fh = open(file, 'w')
        fh.write("# Frequency = %0.2f\n# Amplitude = %d\n" %(freq, ampl))

        # Activate data reporting
        writeln('rept 1')

        # drop lines until the end of a drive cycle.
        line = readln(ser)
        print(type(line), type(line.split()),type(line.split()[0]), int(float(line.split()[0])))
        print("This is line", line, "and this is line.split()[0]", line.split()[0])
        while int(float(line.split()[0])) != 255:
                line = readln(ser)

        # collect N data lines
        for j in range(N):
                fh.write(readln(ser))
        writeln('rept 0')

        # clean up
        fh.close()

def showPlot(file, save=False, polar=False):

        # get the data
        phase, angle, velocity = loadtxt(file, unpack=True)

        # plot the phase space graph, cartesian
        if not polar:
                ax = subplot(111)
                ax.plot(angle*2.*pi/1016., velocity*2.*pi/1016., 'b,')
                xlabel('Angle (radians)')
                ylabel('Angular Velocity (radians/second)')
                xlim(-pi, pi)

        # plot the phase space graph, polar
        else:
                ax = subplot(111, projection='polar')
                ax.plot(angle*2.*pi/1016., velocity*2.*pi/1016., 'b,')
                ax.set_rmax(max(ceil(abs(velocity))))
                ax.grid(True)

        title(file)

        # save the plot
        if save:
                savefig(file+'.pdf')

        # show the plot
        show()

port = '/dev/cu.usbmodem58794001'

ser = serial.Serial(port, 115200, timeout=1)

# startup test
writeln('*idn?')
print(readln(ser))

command = ''


# Main loop
```

```
while command != 'quit':
        command = input('-> ')

        # parse command

        # go through some special cases here

        if command == 'quit':
                continue

        if command == '':
                print(readln(ser))
                continue

        com = command.split()
        if com[0] == 'c':
                # c for collect. Follow 'c' with N and filename.
                # Note: no error-checking! Do it right or suffer. :-(
                N = int(com[1])
                filename = com[2]
                getData(N, filename)
                continue
        if (com[0] == 'p') or (com[0] == 'l'):
                save_figure = False
                if filename:
                        try:
                                if com[1] == 's':
                                        save_figure = True
                        except:
                                pass
                        if com[0] == 'l':
                                showPlot(filename, save=save_figure, polar=True)
                        else:
                                showPlot(filename, save=save_figure, polar=False)
                else:
                        print('No filename given.')
                continue

        # No special case, just write and read.
        writeln(command)
        print(readln(ser))

ser.close()
```

```
CSU Chico Physics Advanced Lab - Chaotic Rotor V2.0 - firmware
Teensy3.1_20170713
```

```
->  c 1 blah.txt
```

```
Saving 256 data points to file 'blah.txt'.
<class 'str'> <class 'list'> <class 'str'> 0
This is line 0.00
 and this is line.split()[0] 0.00
```

```
---------------------------------------------------------------------------
IndexError                                Traceback (most recent call last)
/var/folders/qy/dj5b43xx28b5710pjylf9_hmzsp4nc/T/ipykernel_73842/3634672865.py in <module>
    133                 N = int(com[1])
    134                 filename = com[2]
--> 135                 getData(N, filename)
    136                 continue
    137         if (com[0] == 'p') or (com[0] == 'l'):

/var/folders/qy/dj5b43xx28b5710pjylf9_hmzsp4nc/T/ipykernel_73842/3634672865.py in getData(N, file)
     61         print(type(line), type(line.split()),type(line.split()[0]), int(float(line.split()[0])))
     62         print("This is line", line, "and this is line.split()[0]", line.split()[0])
---> 63         while int(float(line.split()[0])) != 255:
     64                 line = readln(ser)
     65 

IndexError: list index out of range
```

Port is /dev/cu.usbmodem69578601

running ’c 10 test.txtgives a "could not convert string to float:" error. Seems to be inreadln(ser)‘

```
def readln(ser):
    # reads a bytearray from serial and converts it to a string
    BA = ser.readline().decode("utf-8")
    return BA
```

Commenting out `freq=float(readln(ser))` and just setting `freq=0` lets code run and opens file. Nothing in file

0.1 Code for getdata:

```
def getData(N, file):
    # gets N periods worth of data, saves raw data lines to a file.

    N = 256*N
    print("Saving %d data points to file '%s'." % (N, file))
    # get useful data from the MCO
    writeln('freq?')
    freq = 0 #float(readln(ser))
    writeln('ampl?')
    ampl = int(readln(ser))

    # open the file and write the header
    fh = open(file, 'w')
    fh.write("# Frequency = %0.2f\n# Amplitude = %d\n" %(freq, ampl))

    # Activate data reporting
    writeln('rept 1')

    # drop lines until the end of a drive cycle.
    line = readln(ser)
    while int(line.split()[0]) != 255:
        line = readln(ser)

    # collect N data lines
    for j in range(N):
        fh.write(readln(ser))
    writeln('rept 0')

    # clean up
    fh.close()
```

To Do: Look at code to see why `freq?` and `ampl?` give `0 0 nan`. Look at teensy code

Looking at DS box

Doing `c 10 newtest.txt` gives the following error: `ValueError: invalid literal for int() with base 10: 0.00' which is coming from this line ingetdata:`while int(line.split()[0]) != 255:‘

See below for example of `line.split()`

The output I'm getting is `0.00` for `line` as well as for `line.split()[0]`

The issue is `int()` can't handle a decimal string so doing `int(float(...))` works.

[12]:

```
txt = "hello, my name is Peter, I am 26 years old"

x = txt.split()

print(x)
```

```
['hello,', 'my', 'name', 'is', 'Peter,', 'I', 'am', '26', 'years', 'old']
```

[23]:

```
int(float("0.00"))
```

```
[23]: 0
```

```
[ ]:
```