import tkinter

from tkinter import ttk

from PIL import Image, ImageTk

from matplotlib import pyplot as plt

import numpy as np

import tkinter.messagebox as messagebox

import threading as Thread

import os

"""

generate the main window and set the necessary parameters

"""

# find the center position of the window

def getCenterPos(win, width, height):

x\_pos = win.winfo\_screenwidth()

y\_pos = win.winfo\_screenheight()

return width, height, (x\_pos-width)/2, (y\_pos-height)/2

root = tkinter.Tk()

root.title('Izhikevich Neural Model Simulation Tool')

center\_pos = getCenterPos(root, 500, 618)

root.geometry('%dx%d+%d+%d'%(center\_pos[0], center\_pos[1], center\_pos[2], center\_pos[3]))

# root.minsize(center\_pos[0], center\_pos[1])

# root.maxsize(center\_pos[0], center\_pos[1])

root.resizable(0, 0)

"""

generate the widgets

"""

neuron = {

'LTS':(0.02, 0.25, -65, 2),

'TC':(0.02, 0.25, -65, 0.05),

'RS':(0.02, 0.2, -65, 8),

'IB':(0.02, 0.2, -55, 4),

'CH':(0.02, 0.2, -50, 2),

'RZ':(0.1, 0.26, -65, 2),

'FS':(0.1, 0.2, -65, 2),

}

cmb = ttk.Combobox(root)

'''

LTS:(0.02, 0.25, -65, 2)

TC:(0.02, 0.25, -65, 0.05)

RS:(0.02, 0.2, -65, 8)

IB:(0.02, 0.2, -55, 4)

CH:(0.02, 0.2, -50, 2)

RZ:(0.1, 0.26, -65, 2)

FS:(0.1, 0.2, -65, 2)

'''

type\_lb = tkinter.Label(root, text='Neuron Type:')

# type\_lb.place(x=60, y=20)

type\_lb.place(x=40, y=40)

cmb['value'] = ('LTS', 'TC', 'RS', 'IB', 'CH', 'RZ', 'FS')

cmb.current(1)

cmb.place(x=140, y=40)

v\_lb = tkinter.Label(root, text='V:')

v\_lb.place(x=40, y=80)

v = tkinter.StringVar()

v\_in = tkinter.Entry(root, textvariable=v)

v\_in.place(x=60, y=80, width=30)

u\_lb = tkinter.Label(root, text='U:')

u\_lb.place(x=40, y=120)

u = tkinter.StringVar()

u\_in = tkinter.Entry(root, textvariable=u)

u\_in.place(x=60, y=120, width=30)

I\_lb = tkinter.Label(root, text='I:')

I\_lb.place(x=40, y=160)

I = tkinter.StringVar()

I\_in = tkinter.Entry(root, textvariable=I)

I\_in.place(x=60, y=160, width=30)

if not os.path.exists('test.png'):

parameters = neuron.get(cmb.get())

a = parameters[0]

b = parameters[1]

c = parameters[2]

d = parameters[3]

h = 0.1

v\_ = 63

u\_ = -20

I\_ = 0.15

data\_v = list()

data\_u = list()

scale = 1000

for i in range(scale):

u\_ += a\*(b\*v\_ - u\_)\*h

v\_ += (0.04\*v\_\*\*2 + 5\*v\_ + 140 - u\_ + I\_)\*h

if v\_ >= 30:

v\_ = c

u\_ += d

data\_v.append(v\_)

data\_u.append(u\_)

data\_x = np.arange(0, scale/10, 0.1)

plt.title("%s Neuron Simulation Demo"%cmb.get())

plt.xlabel("t/ms")

plt.plot(data\_x, data\_v)

plt.plot(data\_x, data\_u)

save\_name = 'test.png'

plt.savefig(save\_name, dpi=(plt.rcParams['figure.dpi']/2))

plt.clf()

load = Image.open('test.png')

img = ImageTk.PhotoImage(load)

image\_lb = tkinter.Label(root, image=img)

image\_lb.place(x=100, y=220)

run\_bt = tkinter.Button(text='Run')

run\_bt.place(x=100, y=540, width=50)

show\_bt = tkinter.Button(text='Show')

show\_bt.place(x=180, y=540, width=50)

save\_bt = tkinter.Button(text='Save')

save\_bt.place(x=260, y=540, width=50)

re\_bt = tkinter.Button(text='Reset')

re\_bt.place(x=340, y=540, width=50)

figure\_name = tkinter.StringVar()

figure\_name.set('Figure: %s Neuron Wave'%cmb.get())

figure\_lb = tkinter.Label(root, textvariable=figure\_name)

figure\_lb.place(x=200, y=480)

tip = tkinter.StringVar()

tip\_lb = tkinter.Label(root, textvariable=tip, fg='red')

tip\_lb.place(x=135, y=200)

"""

write functions for widgets

"""

save\_img = None

def run(event):

parameters = neuron.get(cmb.get())

a = parameters[0]

b = parameters[1]

c = parameters[2]

d = parameters[3]

h = 0.1

v\_ = 63

if v.get() != '':

v\_ = float(v.get())

u\_ = -20

if u.get() != '':

u\_= float(u.get())

I\_ = 0.15

if I.get() != '':

I\_ = float(I.get())

data\_v = list()

data\_u = list()

scale = 1000

for i in range(scale):

u\_ += a\*(b\*v\_ - u\_)\*h

v\_ += (0.04\*v\_\*\*2 + 5\*v\_ + 140 - u\_ + I\_)\*h

if v\_ >= 30:

v\_ = c

u\_ += d

data\_v.append(v\_)

data\_u.append(u\_)

data\_x = np.arange(0, scale/10, 0.1)

plt.title("%s Neuron Simulation Demo"%cmb.get())

plt.xlabel("t/ms")

plt.plot(data\_x, data\_v)

plt.plot(data\_x, data\_u)

save\_name = 'show.png'

plt.savefig(save\_name, dpi=(plt.rcParams['figure.dpi']/2))

plt.clf()

load = Image.open(save\_name)

global save\_img

save\_img = load

img = ImageTk.PhotoImage(load)

image\_lb.configure(image=img)

image\_lb.image = img

def show(event):

parameters = neuron.get(cmb.get())

a = parameters[0]

b = parameters[1]

c = parameters[2]

d = parameters[3]

h = 0.1

v\_ = 63

if v.get() != '':

v\_ = float(v.get())

u\_ = -20

if u.get() != '':

u\_= float(u.get())

I\_ = 0.15

if I.get() != '':

I\_ = float(I.get())

data\_v = list()

data\_u = list()

scale = 1000

for i in range(scale):

u\_ += a\*(b\*v\_ - u\_)\*h

v\_ += (0.04\*v\_\*\*2 + 5\*v\_ + 140 - u\_ + I\_)\*h

if v\_ >= 30:

v\_ = c

u\_ += d

data\_v.append(v\_)

data\_u.append(u\_)

data\_x = np.arange(0, scale/10, 0.1)

plt.title("%s Neuron Simulation Demo"%cmb.get())

plt.xlabel("t/ms")

plt.plot(data\_x, data\_v)

plt.plot(data\_x, data\_u)

plt.show()

def save(event):

global save\_img

if save\_img:

save\_img.save('%s.png'%cmb.get())

tip\_lb.configure(fg='blue')

tip.set('The figure have saved into %s.png'%cmb.get())

else:

tip\_lb.configure(fg='red')

tip.set('You must click \'Run\' and generate a figure')

def reset(event):

v.set('')

u.set('')

I.set('')

tip.set('')

def callback():

if messagebox.askokcancel('Quit', 'Do you really wish to quit?'):

plt.close(1)

root.destroy()

"""

bind the functions with the widgets

"""

re\_bt.bind('<Button-1>' , reset)

run\_bt.bind('<Button-1>', run)

save\_bt.bind('<Button-1>', save)

show\_bt.bind('<Button-1>', show)

root.protocol("WM\_DELETE\_WINDOW", callback)

root.mainloop()