EE2703 week 5

G.Sai Krishna Reddy <ee21b049>

March 9, 2023

[1]:
```
# Magic command below to enable interactivity in the JupyterLab interface
%matplotlib ipympl
# Some basic imports that are useful
import numpy as np
import matplotlib.pyplot as plt
from matplotlib.animation import FuncAnimation
```

1 POLYGON Function:

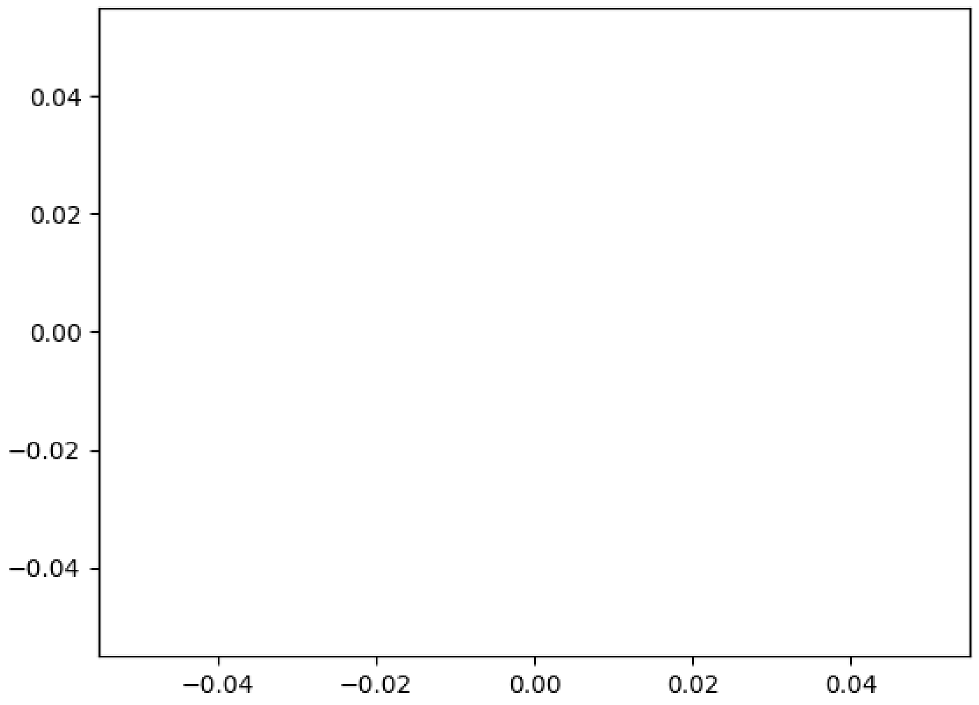
[2]:
```
def polygon(t,n):
    l=int(t/n)
    th=2*(np.pi)*1/n
    x=[]
    y=[]
    for i in range(n):
        x.append(1*np.cos(i*th))
        y.append(1*np.sin(i*th))
    #print(x,y)
    tsx=[]
    tsy=[]
    for i in range(n):
        if i==n-1:
            tsx.append(np.linspace(x[i],x[0],l))
            tsy.append(np.linspace(y[i],y[0],l))
        else:
            tsx.append(np.linspace(x[i],x[i+1],l))
            tsy.append(np.linspace(y[i],y[i+1],l))

    # print(tsx)
    # print(tsy)
    xs=np.concatenate(tsx)
    ys=np.concatenate(tsy)
    return xs,ys
```

Here in the above polygon function, I found the vertices as (rcos(th),rsin(th)). Then I created points between vertices and I concatenated them and returned as xs,ys.

[3]:
```python
t=840
n=3
x=0
y=0
def func(t,n,x,y):
    if x==0 or y==0:
        xc, yc = polygon(t,n-1)
        xs, ys = polygon(t,n)
        return xs,ys,xc,yc
    if x==1 and y!=0:
        xc, yc = polygon(t,n+1)
        xs, ys = polygon(t,n)
        return xs,ys,xc,yc
```

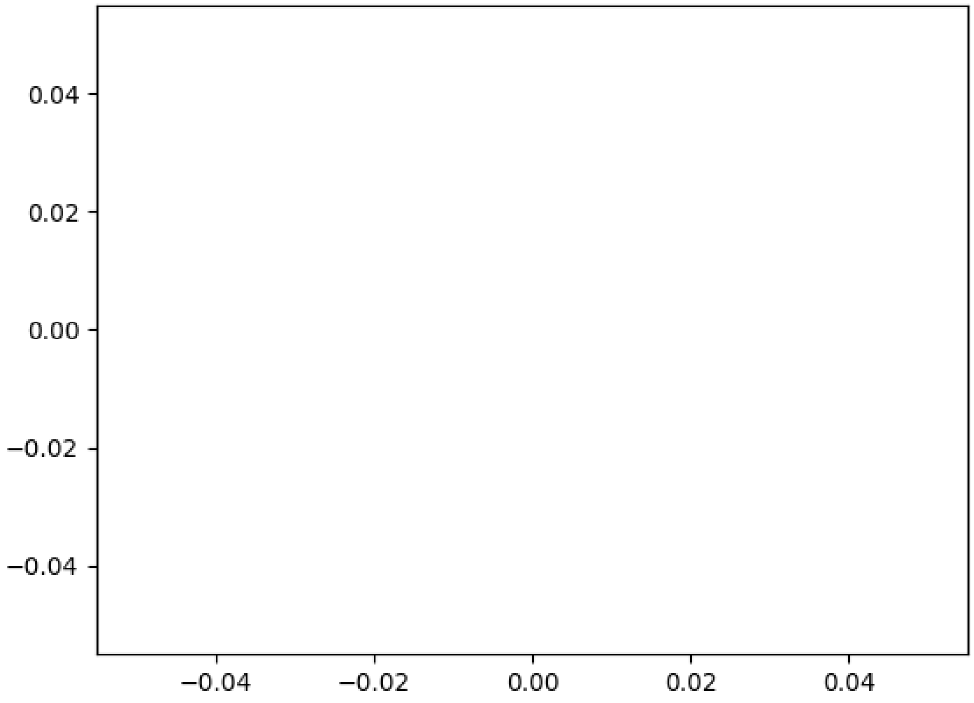
[4]:
```python
fig, ax = plt.subplots()
xdata, ydata = [], []
ln, = ax.plot([], [], 'r')
def init():
    ax.set_xlim(-1.2, 1.2)
    ax.set_ylim(-1.2, 1.2)
    return ln,
def update(frame):
    global n,x,y
    if frame==0 and x==1 and n>3:
        if n==8:
            y=1
        n=n-1
    if frame==0 and x==0:
        n=n+1
        if n==8:
            x=1
    xs,ys,xc,yc=func(t,n,x,y)
    xdata, ydata = morph(xs, ys, xc, yc, frame)
    ln.set_data(xdata, ydata)
    return ln,
```



In the above update function, every time frames becomes zero(In the below cell I gave frames as an array using for loop) I changed n value so that it continuously changes from one polygon to other.After octagon I just reversed oint xc,yc and xs and ys and decreased n so that it goes from octagon to triangle.

[5]:
```
def morph(x1, y1, x2, y2, alpha):
    xm = alpha * x1 + (1-alpha) * x2
    ym = alpha * y1 + (1-alpha) * y2
    return xm, ym
#t = np.linspace(3*np.pi/4, -5*np.pi/4, 200)
#t=np.linspace(-1,1,240)
#print(f"Square: {np.shape(xs)}")
#ani = FuncAnimation(fig, update, frames=np.linspace(0, 1, 128),init_func=init,␣
 ↪blit=True, interval=10, repeat=True)
frames = []
for i in range(11):
    if i == 5:
        frames.extend(np.linspace(1, 1, 30))
    else:
        frames.extend(np.linspace(0, 1, 60))
```

```
# Create the animation using the concatenated frames
ani = FuncAnimation(fig, update, frames=frames,init_func=init, blit=True,␣
 ↪interval=10, repeat=False)
plt.show(ani)
```



```
[ ]:
```