7. Tạo Animation từ dữ liệu 3D sử dụng MoviePy

1. Animations với Vispy (Cài đặt bằng VS code)

Vispy là một thư viện trực quan hóa dữ liệu 3D tương tác khác, dựa trên OpenGL. Đối với Mayavi, trước tiên chúng ta tạo một hình và một lưới, sau đó chúng ta tạo hoạt ảnh bằng MoviePy.

import thư viện: Cài đặt thư viện vispy và PyQt5 (hoặc PyQt6)

```
from moviepy.editor import VideoClip
import numpy as np
from vispy import app, scene
from vispy.gloo.util import _screenshot
```

Thiết lập bảng vẽ và view

```
canvas = scene.SceneCanvas(keys='interactive')
view = canvas.central_widget.add_view()
view.set_camera('turntable', mode='perspective', up='z', distance=2, azimuth=30., elevation=65.)
```

Thiết lập tọa độ và bề mặt

```
xx, yy = np.arange(-1,1,.02),np.arange(-1,1,.02)
X,Y = np.meshgrid(xx,yy)
R = np.sqrt(X**2+Y**2)
Z = lambda t : 0.1*np.sin(10*R-2*np.pi*t)
surface = scene.visuals.SurfacePlot(x= xx-0.1, y=yy+0.2, z= Z(0),shading='smooth', color=(0.5, 0.5, 1, 1))
```

Thêm bề mặt vào khung nhìn và bảng vẽ

```
view.add(surface)
canvas.show()
```

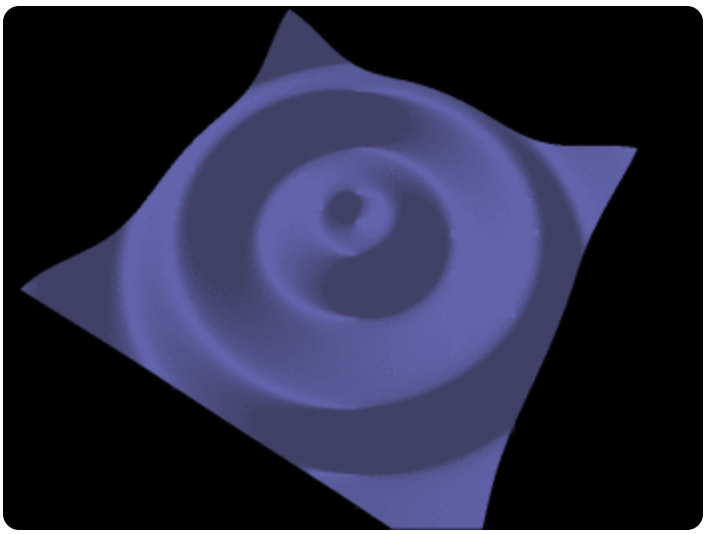
Xây dựng hàm make_frame

```
def make_frame(t):
    surface.set_data(z = Z(t)) # Update the mathematical surface
    canvas.on_draw(None) # Update the image on Vispy's canvas
```

```
    return _screenshot((0,0,canvas.size[0],canvas.size[1]))[:,:,:3]
```

Render đối tượng

```
animation = VideoClip(make_frame, duration=1).resize(width=350)
animation.write_gif('sinc_vispy.gif', fps=20, opt='OptimizePlus')
```



2. Animations sử dụng Matplotlib

import thư viện

```
import matplotlib.pyplot as plt
import numpy as np
from moviepy.video.io.bindings import mplfig_to_npimage
import moviepy.editor as mpy
```

Thiết lập duration là 2

Tạo file dữ liệu với Matplotlib

```
fig_mpl, ax = plt.subplots(1,figsize=(5,3), facecolor='white')
xx = np.linspace(-2,2,200) # the x vector
zz = lambda d: np.sinc(xx**2)+np.sin(xx+d) # the (changing) z vector
ax.set_title("Elevation in y=0")
ax.set_ylim(-1.5,2.5)
line, = ax.plot(xx, zz(0), lw=3)
```

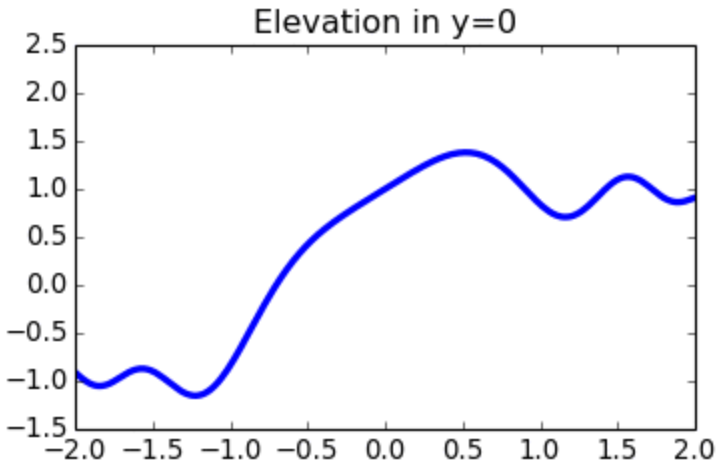
Xây dựng hàm make_frame

```
def make_frame_mpl(t):
    line.set_ydata( zz(2*np.pi*t/duration))
```

```
    return mplfig_to_npimage(fig_mpl)
```

Render đối tượng

```
animation =mpy.VideoClip(make_frame_mpl, duration=duration)
animation.write_gif("sinc_mpl.gif", fps=20)
```



> **Bài tập 1: Tạo file Animations cho đối tượng với yêu cầu bên dưới:**

```
from sklearn import svm
from sklearn.datasets import make_moons
```

1. **Thiết lập đối tượng random ngẫu nhiênvới X, Y = make_moons(50, noise=0.1, random_state=2)**
2. **Thiết lập giá trị subplots với figsize là 4 x 4 và facecolor là (1,1,1)**
3. **Sử dụng thuộc tính subplots_adjust của fig để thiết lập left=0, right=1, bottom=0**
4. **Thiết lập xx, yy lần lượt là np.meshgrid(np.linspace(-2,3,500), np.linspace(-1,2,500))**
5. **Xây dựng hàm make_frame:**

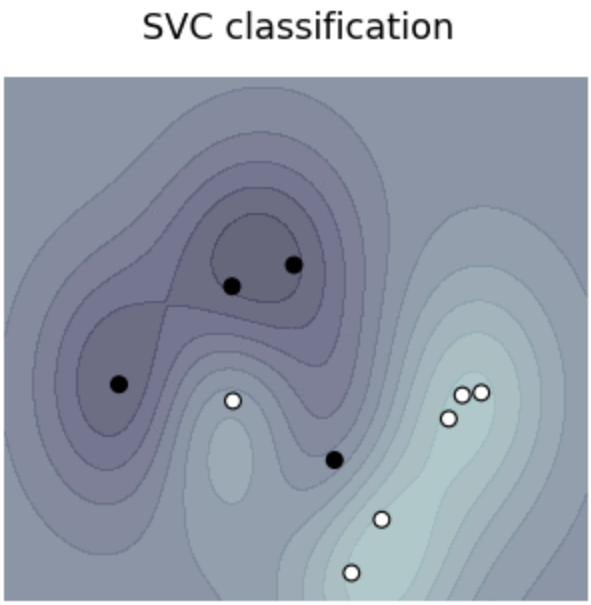
```
def make_frame(t):
    ax.clear()
    ax.axis('off')
    ax.set_title("SVC classification", fontsize=16)

    classifier = svm.SVC(gamma=2, C=1)
    # the varying weights make the points appear one after the other
    weights = np.minimum(1, np.maximum(0, t**2+10-np.arange(50)))
    classifier.fit(X, Y, sample_weight=weights)
    Z = classifier.decision_function(np.c_[xx.ravel(), yy.ravel()])
    Z = Z.reshape(xx.shape)
    ax.contourf(xx, yy, Z, cmap=plt.cm.bone, alpha=0.8,
                vmin=-2.5, vmax=2.5, levels=np.linspace(-2,2,20))
    ax.scatter(X[:,0], X[:,1], c=Y, s=50*weights, cmap=plt.cm.bone)

```

```
    return mplfig_to_npimage(fig)
```

6. **Thực hiện render đối tượng sử dụng make_frame với duration là 7**
7. **Lưu file render với tên svm.gif; biết rằng fps là 15**



3. Animations sử dụng Numpy

import thư viện

```
import numpy as np
from scipy.ndimage.filters import convolve
import moviepy.editor as mpy
```

Thiết lập tham số và hằng số

```
infection_rate = 0.3
incubation_rate = 0.1
dispersion_rates  = [0, 0.07, 0.03] # for S, I, R
dispersion_kernel = np.array([[0.5, 1 , 0.5],
                              [1  , -6, 1],
                              [0.5, 1, 0.5]])
```

Đọc ảnh: square.png

```
france = mpy.ImageClip("/content/square.png").resize(width=400)
```

Tạo hiệu ứng lan toản

```
SIR = np.zeros( (3,france.h, france.w),  dtype=float)
SIR[0] = france.get_frame(0).mean(axis=2)/255
start = int(0.6*france.h), int(0.737*france.w)
SIR[1,start[0], start[1]] = 0.8 # infection in Grenoble at t=0
```

```
dt = 1.0 # one update = one hour of real time
hours_per_second= 7*24 # one second in the video = one week in the model
world = {'SIR':SIR, 't':0}
```

Xây dựng hàm infection, dispersion và update

```
def infection(SIR, infection_rate, incubation_rate):
    """ Computes the evolution of #Sane, #Infected, #Rampaging"""
    S,I,R = SIR
    newly_infected = infection_rate*R*S
    newly_rampaging = incubation_rate*I
    dS = - newly_infected
    dI = newly_infected - newly_rampaging
    dR = newly_rampaging
    return np.array([dS, dI, dR])

def dispersion(SIR, dispersion_kernel, dispersion_rates):
    """ Computes the dispersion (spread) of people """
    return np.array( [convolve(e, dispersion_kernel, cval=0)*r
                      for (e,r) in zip(SIR, dispersion_rates)])

def update(world):
    """ spread the epidemic for one time step """
    infect = infection(world['SIR'], infection_rate, incubation_rate)
    disperse = dispersion(world['SIR'], dispersion_kernel, dispersion_rates)
    world['SIR'] += dt*( infect + disperse)
    world['t'] += dt
```

Xây dựng hàm tạo Animations

```
def world_to_npimage(world):
    coefs = np.array([2,25,25]).reshape((3,1,1))
    accentuated_world = 255*coefs*world['SIR']
    image = accentuated_world[::-1].swapaxes(0,2).swapaxes(0,1)
    return np.minimum(255, image)

def make_frame(t):
    while world['t'] < hours_per_second*t:
        update(world)
    return world_to_npimage(world)
```

Render video

```
animation = mpy.VideoClip(make_frame, duration=25)
animation.write_videofile('test.mp4', fps=20)
```

0:00 / 0:25

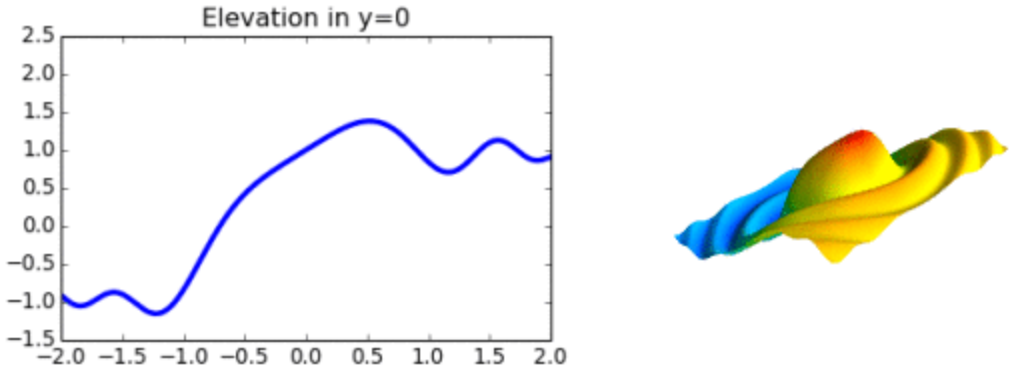
4. Ghép các ảnh animations

import thư viện

```
import moviepy.editor as mpy
```

Ghép file sử dụng clips_array

```
clip_mayavi = mpy.VideoFileClip("sinc.gif")
clip_mpl = mpy.VideoFileClip("sinc_mpl.gif").resize(height=clip_mayavi.h)
animation = mpy.clips_array([[clip_mpl, clip_mayavi]])
animation.write_gif("sinc_plot.gif", fps=20)
```

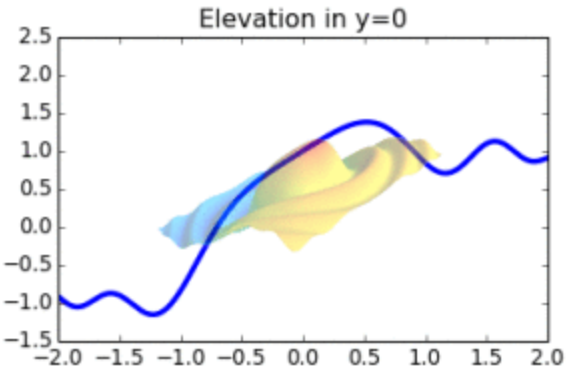


Ghép file sử dụng CompositeVideoClip

```
clip_mayavi2 = (clip_mayavi.fx( mpy.vfx.mask_color, [255,255,255])
                .set_opacity(.4)
                .resize(height=0.85*clip_mpl.h)
                .set_pos('center'))

animation = mpy.CompositeVideoClip([clip_mpl, clip_mayavi2])
```

```
7 animation.write_gif("sinc_plot2.gif", fps=20)
```



> **Bài tập 2: Ghép file animation và thực hiện các yêu cầu bên dưới:**

1. import thư viện

```
1 import moviepy.editor as mpy
2 import skimage.exposure as ske # rescaling, histogram eq.
3 import skimage.filter as skf
```

2. Khởi tạo đối tượng

```
1 clip = mpy.VideoFileClip("sinc.gif")
2 gray = clip.fx(mpy.vfx.blackwhite).to_mask()
```

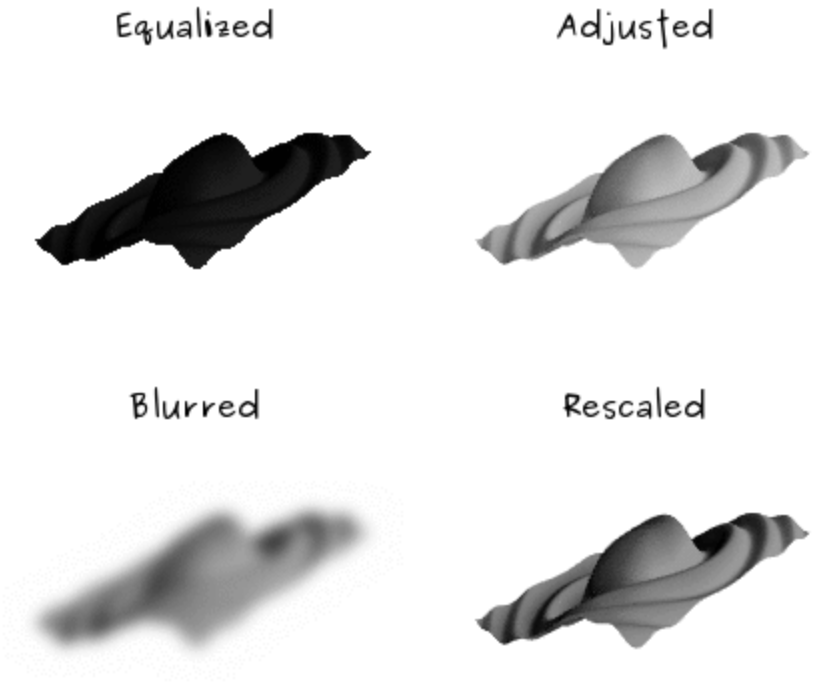
3. Xây dựng hiệu ứng ảnh xám

```
1 ef apply_effect(effect, title, **kw):
2     filtr = lambda im: effect(im, **kw)
3     new_clip = gray.fl_image(filtr).to_RGB()
4     return new_clip
```

4. Tạo ra 4 hiệu ứng ảnh xám khác nhau gồm Equalized, Rescaled, Adjusted và Blurred (sigma là 4) như ví dụ bên dưới

```
1 equalized = apply_effect(ske.equalize_hist, "Equalized")
```

5. Thực hiện ghép animation sử dụng clips_array và lưu lại với tên test2x2.gif với fps là 20



5. Animations với Mayavi (Cài đặt bằng VS code)

Import thư viện

```
import numpy as np
import mayavi.mlab as mlab
import  moviepy.editor as mpy
```

Thiết lập giá trị duration = 2

Tạo file dữ liệu với Mayavi

```
fig_myv = mlab.figure(size=(220,220), bgcolor=(1,1,1))
X, Y = np.linspace(-2,2,200), np.linspace(-2,2,200)
XX, YY = np.meshgrid(X,Y)
ZZ = lambda d: np.sinc(XX**2+YY**2)+np.sin(XX+d)
```

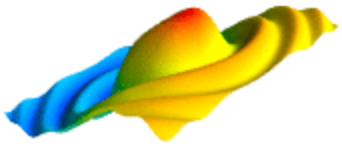
Xây dựng hàm make_frame

```
def make_frame(t):
    mlab.clf() # clear the figure (to reset the colors)
    mlab.mesh(YY,XX,ZZ(2*np.pi*t/duration), figure=fig_myv)
    return mlab.screenshot(antialiased=True)
```

Render đối tượng

```
animation = mpy.VideoClip(make_frame, duration=duration)
```

```
animation.write_gif("sinc.gif", fps=20)
```



Bài tập 3: Tạo Animation từ dữ liệu với các yêu cầu bên dưới

1. Thiết lập giá trị duration là 2
2. Giá trị mlab.figure có kích thước 500 x 500 và bg color là (1,1,1)
3. Thiết lập giá trị u là np.linspace(0,2*np.pi,100)
4. Tọa độ xx, yy, zz lần lượt là np.cos(u), np.sin(3*u), np.sin(u)
5. Thiết lập giá trị l là

```
l = mlab.plot3d(xx,yy,zz, representation="wireframe", tube_sides=5, line_width=.5, tube_radius=0.
```

6. Xây dựng hàm make_frame

```
def make_frame(t):
    """ Generates and returns the frame for time t. """
    y = np.sin(3*u)*(0.2+0.5*np.cos(2*np.pi*t/duration))
    l.mlab_source.set(y = y) # change y-coordinates of the mesh
    mlab.view(azimuth= 360*t/duration, distance=9) # camera angle
    return mlab.screenshot(antialiased=True) # return a RGB image
```

7. Thực hiện render cho đối tượng biết rằng kích thước đối tượng được resize còn 0.5 sau khi render
8. Thực hiện lưu file render với tên wireframe.mp4 và wireframe.gif với fps là 20