import numpy as np  
import matplotlib.pyplot as plt  
import matplotlib.animation as animation  
from matplotlib.animation import FuncAnimation, FFMpegWriter  
plt.rcParams['font.sans-serif'] = ['SimHei'] *# 设置中文字体为黑体*plt.rcParams['axes.unicode\_minus'] = False *# 解决负号显示问题  
# 默认基础参数*is\_video = "n"  
w = 1 *# 角频率 (rad/s)*v = 1 *# 速度 (m/s)*A = 1 *# 振幅 (m)*init1, init2 = 0, 0 *# 初始相位 (rad)*target = [2.5, 6] *# 被分析点*x\_min, x\_max = -10, 10 *# x轴的范围*y\_min, y\_max = -10, 10 *# y轴的范围*x = np.linspace(x\_min, x\_max, 750) *# x轴的采样点*y = np.linspace(y\_min, y\_max, 750) *# y轴的采样点*X, Y = np.meshgrid(x, y) *# 生成网格点*distance = 0  
*# 根据输入调整参数*print("------------------------配置参数-----------------------")  
print("1.y 表示yes n 表示 no 其他选择视为no")  
print("2.在选择后摁下回车确定")  
print("3.不输入数据直接回车表示使用默认参数\n\n")  
*# 波源位置*source1 = list(map(int, input("波源位置（坐标用空格隔开，如\“3 5\"):(默认[0, 0])").split()))  
source2 = list(map(int, input("第二波源位置：（坐标用空格隔开，如\“3 5\")(默认[5, 0])").split()))  
if not source1 or len(source1)!=2:  
 source1 = (0, 0)  
if not source2 or len(source2)!=2:  
 source2 = (5, 0)  
x3, y3 = (source1[0]+source2[0])/2, (source1[1] + source2[1])/2  
*# 传播速度*try:  
 v = int(input("波的传播速度：(默认1)"))  
except:  
 v = 1  
*# 初始相位*sub\_angle = 0  
try:  
 sub\_angle = int(input("两波源相位差(单位：π rad)(\"默认0\"):"))  
except:  
 sub\_angle = 0  
*# 是否平面化某一点振幅合成情况*is\_flatten = input("是否平面化某一点振幅合成情况(y/n)(默认no)")  
if is\_flatten=="y":  
 target = list(map(int, input("该点坐标：（默认[2.5, 8])").split()))  
 if not target or len(target)!=2:  
 target = [2.5, 8]  
*# 是否振幅随距离衰减*is\_sub = input("是否振幅随传播距离衰减(y/n)(默认n)")  
if not is\_sub:  
 is\_sub = "n"  
*# 是否导出视频*is\_video = input("是否导出视频（视频比动画更流畅)：y/n(默认no)")  
high\_bite = "n"  
if is\_video=="y":  
 video\_name = input("请输入导出视频名称(默认output\_video)")  
 if not video\_name:  
 video\_name = "output\_video"  
 high\_bite = input("是否导出高清视频(占用内存大）(y/n)(默认no)")  
*# 动画说明*print("\n---------------------动画说明--------------------------")  
print("在生成的动画中，蓝色代表正高度，红色代表负深度，颜色越深，偏离平衡位置距离越远")  
if is\_flatten:  
 print("在对点的分析的余弦图像中，我们规定从波源到被分析点为x轴正方向，x轴正方向顺时针偏移90度为y轴正方向")  
 print("不同颜色的余弦图像表示由不同波源发出，同一波源发出的余弦图像颜色相同")  
if is\_video=="y":  
 print("\n正在导出视频,这大概需要一分钟(视频会被存放在本目录下)")  
*# 计算波的振幅函数*def wave\_amplitude(x, y, t, source):  
 distance = np.sqrt((x - source[0])\*\*2 + (y - source[1])\*\*2)  
 add\_angle = sub\_angle if source == source2 else 0  
 return np.where(t > distance / v, 1.3\*np.cos(w \* np.pi \* (distance / v - t)+np.pi/2+add\_angle\*np.pi), 0)  
*# 一点到一波源的分析*def my\_amplitude(target, source1, color, t):  
 *# 绘制方向轴  
 # print("tar")* k = (target[1]-source1[1])/(target[0]-source1[0]) *# 斜率* angle = np.pi / 2 if target[0] - source1[0] == 0 else (np.arctan(k) if k > 0 else np.arctan(k) + np.pi)  
 base\_X1 = np.linspace(source1[0], target[0], 200)  
 base\_Y1 = np.linspace(source1[1], target[1], 200)  
 plt.plot(base\_X1, base\_Y1, color=color, linestyle="dashed")  
 *# 绘制余弦图像* ampl = wave\_amplitude(base\_X1, base\_Y1, t, source1)  
 angle1 = (angle - np.pi / 2) *# % np.pi* \_X1 = base\_X1 + ampl \* np.cos(angle1)  
 \_Y1 = base\_Y1 + ampl \* np.sin(angle1)  
 plt.plot(\_X1, \_Y1, color=color)  
 *# 添加端点的小黑点* plt.scatter(target[0], target[1], color='black', s=30)  
*# 同一点到两波源的分析*def total\_amplitude(target, t):  
 my\_amplitude(target, source1, "blue", t)  
 my\_amplitude(target, source2, "red", t)  
def update(t):  
 t = t/2  
 plt.clf() *# 清除之前的图像，避免叠加显示* distance = np.sqrt((X - x3)\*\*2 + (Y - y3)\*\*2)  
 k = 1 - 1.2\*distance/15 *# 振幅衰减系数* k = np.where(k>0, k, 0)  
 if is\_sub=="n":  
 k = 1  
 Z = k\*(wave\_amplitude(X, Y, t, source1)+wave\_amplitude(X, Y, t, source2)) *#if t>distance/v else 0* plt.title('两波的干涉 (时间: {:.2f})'.format(2\*t))  
 plt.imshow(Z, cmap='RdYlBu', extent=[x\_min, x\_max, y\_min, y\_max], origin='lower', vmin=-1.5, vmax=1.5)  
 *# 添加波源的小黑点* plt.scatter([source1[0], source2[0]], [source1[1], source2[1]], color='black', s=80)  
 if is\_flatten=="y":  
 total\_amplitude(target, t)  
 return plt  
*# 创建动画*fig = plt.figure(figsize=(8, 8))  
ani = animation.FuncAnimation(fig, update, frames=np.arange(0, 60, 0.1), interval=100)  
*# 导出视频*if is\_video=="y":  
 writer = FFMpegWriter(fps=10, metadata=dict(artist='lisanndesu'), bitrate=2000000 if high\_bite=="y" else 1000)  
 ani.save(video\_name+".mp4", writer=writer)  
 print("成功导出视频，已存放至本程序所在目录下")  
*# 显示动画*plt.show()