just shown the MBT engine’s linear, non-interacting regime—the superposition works, but there’s no rich “quantum interference” unless you make the packets collide or pass through each other (or add a barrier for true double-slit).

import numpy as np

import matplotlib.pyplot as plt

from scipy.integrate import trapezoid

# PARAMETERS

grid\_size = 100

timesteps = 50

dt = 0.2

width = 5

r = np.linspace(0, grid\_size, grid\_size)

dr = r[1] - r[0]

# INIT WAVE PACKETS

xL, xR = 25, 75

kL, kR = +0.6, -0.6

psi1 = np.exp(-((r - xL)\*\*2)/(2\*width\*\*2)) \* np.exp(1j \* kL \* r)

psi2 = np.exp(-((r - xR)\*\*2)/(2\*width\*\*2)) \* np.exp(1j \* kR \* r)

psi = np.array([psi1, psi2])

n = 2

# STORAGE

psi\_total = []

overlap = []

CoMs = []

# TIME EVOLUTION

for t in range(timesteps):

V = np.zeros\_like(r) # No potential, pure interference

for i in range(n):

lap = np.zeros\_like(psi[i], dtype=complex)

lap[1:-1] = (psi[i][2:] - 2\*psi[i][1:-1] + psi[i][:-2]) / dr\*\*2

psi[i] += dt \* (0.7 \* lap - 0.6 \* V \* psi[i])

norm = np.sqrt(trapezoid(np.abs(psi[i])\*\*2, r))

if norm != 0:

psi[i] /= norm

psi\_total.append(np.sum(psi, axis=0))

ov = trapezoid(np.abs(psi[0]\*np.conj(psi[1])), r)

overlap.append(ov)

CoMs.append([trapezoid(r \* np.abs(psi[i])\*\*2, r)/trapezoid(np.abs(psi[i])\*\*2, r) for i in range(n)])

# PLOT

psi\_total = np.array(psi\_total)

CoMs = np.array(CoMs)

colors = ['navy', 'crimson']

fig, axs = plt.subplots(3, 1, figsize=(9,9))

for i in range(n):

axs[0].plot(r, np.abs(psi[i])\*\*2, label=f'ψ{i+1}', color=colors[i])

axs[0].plot(r, np.abs(psi\_total[-1])\*\*2, color='black', lw=2, label='Final Total')

axs[0].set\_title("Superposition & Interference — MBT")

axs[0].legend()

axs[1].plot(np.linspace(0, dt\*timesteps, timesteps), overlap, color='purple')

axs[1].set\_ylabel("Overlap")

axs[1].set\_title("Overlap (Entanglement Monitor)")

for i in range(n):

axs[2].plot(np.linspace(0, dt\*timesteps, timesteps), CoMs[:, i], label=f'ψ{i+1} CoM', color=colors[i])

axs[2].set\_title("Centers of Mass Evolution")

axs[2].set\_xlabel("Time")

axs[2].legend()

plt.tight\_layout()

plt.show()