import numpy as np

import matplotlib.pyplot as plt

from scipy.integrate import trapezoid

# --- Parameters ---

grid\_size = 120

timesteps = 180

dt = 0.12

width = 6

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

dr = r[1] - r[0]

# --- Barrier Profile ---

V = np.zeros\_like(r)

V[(r > 50) & (r < 56)] = 0.08

V[(r > 62) & (r < 72)] = 0.04

# --- Mobile Seed Function ---

def moving\_seed(t, freq, base\_phase, path):

center = path(t)

phase = base\_phase + 0.5 \* np.sin(freq \* t)

return np.exp(-((r - center)\*\*2)/(2 \* width\*\*2)) \* np.exp(1j \* phase)

# --- Evolution Function with Moving Internal Field ---

def evolve\_moving\_seed(base\_phase=np.pi, freq=0.06, x0=95, k0=-1.2):

ψ = np.exp(-((r - x0)\*\*2)/(2 \* width\*\*2)) \* np.exp(1j \* k0 \* r)

ψ\_total = []

# Define ψ₁'s position tracker (approximate linear motion)

def packet\_path(t):

return x0 + k0 \* dt \* t

for t in range(timesteps):

ψ\_internal = moving\_seed(t, freq, base\_phase, packet\_path)

lap\_int = np.zeros\_like(ψ, dtype=complex)

lap\_int[1:-1] = (ψ\_internal[2:] - 2 \* ψ\_internal[1:-1] + ψ\_internal[:-2]) / dr\*\*2

lap\_ψ = np.zeros\_like(ψ, dtype=complex)

lap\_ψ[1:-1] = (ψ[2:] - 2 \* ψ[1:-1] + ψ[:-2]) / dr\*\*2

ψ += dt \* (0.65 \* lap\_ψ - 0.5 \* V \* ψ + 0.65 \* lap\_int)

norm = np.sqrt(trapezoid(np.abs(ψ)\*\*2, r))

if norm != 0:

ψ /= norm

if t % 15 == 0:

ψ\_total.append(np.abs(ψ)\*\*2)

final = np.abs(ψ)\*\*2

T = trapezoid(final[r > 85], r[r > 85])

R = trapezoid(final[r < 35], r[r < 35])

return ψ\_total, T, R

# --- Run the mobile seed experiment ---

snap\_mob, T\_mob, R\_mob = evolve\_moving\_seed()

# --- Plot Results ---

fig, axs = plt.subplots(2, 2, figsize=(15,8))

# Evolution plots

for ψs in snap\_mob:

axs[0,0].plot(r, ψs, alpha=0.5)

axs[0,0].plot(r, V / np.max(V) \* np.max([np.max(p) for p in snap\_mob]), 'k--', lw=2)

axs[0,0].set\_title("Mobile Seed Dispatch: ψ₁–Seed Trajectory Coupling")

# Transmission and Reflection

axs[0,1].bar(['→Forward'], [T\_mob], color='lightgreen')

axs[0,1].set\_ylim(0, 1)

axs[0,1].set\_ylabel("Transmission Probability")

axs[0,1].set\_title("Dispatch Access with Mobile Internal Field")

axs[1,1].bar(['→Forward'], [R\_mob], color='gold')

axs[1,1].set\_ylim(0, 1)

axs[1,1].set\_ylabel("Reflection Probability")

axs[1,1].set\_title("Dispatch Rejection with Mobile Internal Field")

axs[1,0].axis('off') # Unused subplot

plt.tight\_layout()

plt.show()