import numpy as np

import matplotlib.pyplot as plt

from scipy.ndimage import laplace

# --- Parameters ---

N = 60 # grid size (NxN)

timesteps = 180

dt = 0.09

barrier\_prob = 0.27 # fraction of sites with a barrier

barrier\_min, barrier\_max = 1.2, 1.8 # barrier heights

# --- Initial wavepacket options ---

k0 = 0.35 # set 0.35 for slow drift, or 0 for zero momentum

use\_zero\_momentum = False # <---- SET THIS TRUE/FALSE to test both!

# --- Barrier map ---

V = np.zeros((N, N))

rng = np.random.default\_rng(42) # reproducible

barrier\_mask = rng.random((N, N)) < barrier\_prob

V[barrier\_mask] = barrier\_min + (barrier\_max - barrier\_min) \* rng.random(np.sum(barrier\_mask))

# --- Initial wavepacket ---

x0, y0 = N // 2, 10

sigma = 5.0

X, Y = np.meshgrid(np.arange(N), np.arange(N), indexing='ij')

if use\_zero\_momentum:

psi = np.exp(-((X-x0)\*\*2 + (Y-y0)\*\*2)/(2\*sigma\*\*2))

else:

psi = np.exp(-((X-x0)\*\*2 + (Y-y0)\*\*2)/(2\*sigma\*\*2)) \* np.exp(1j \* k0 \* Y)

# Normalize

psi = psi / np.sqrt(np.sum(np.abs(psi)\*\*2))

# --- Time evolution ---

for t in range(timesteps):

lap = laplace(psi, mode='wrap')

psi += dt \* (0.38 \* lap - 0.21 \* V \* psi)

# Normalize each step

norm = np.sqrt(np.sum(np.abs(psi)\*\*2))

if norm != 0:

psi /= norm

# --- Analysis: Transmission ---

transmit\_prob = np.sum(np.abs(psi[:, -1])\*\*2)

print(f"Transmission probability (right edge): {transmit\_prob:.4f}")

# --- Plot ---

plt.figure(figsize=(8,7))

plt.imshow(np.abs(psi)\*\*2, origin='lower', cmap='viridis')

y\_b, x\_b = np.where(V > 0)

plt.scatter(x\_b, y\_b, s=6, c='k', label='Barrier', alpha=0.9)

plt.title(f"2D MBT Quantum Percolation — Final Probability Density\n({'Zero' if use\_zero\_momentum else 'Low'} Momentum)")

plt.xlabel('X')

plt.ylabel('Y')

plt.colorbar(label=r'$|\psi|^2$')

plt.legend()

plt.tight\_layout()

plt.show()