import numpy as np

import matplotlib.pyplot as plt

# Parameters

N = 32 # Grid size

steps = 100 # Timesteps for the braid

fraction = 1/3 # Anyon fractional charge

# --- MBT Phase Field ---

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

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

# Place two anyons: one fixed, one to braid

center1 = (N//2, N//3)

center2 = (N//2, 2\*N//3)

def inject\_anyon(phase, center, charge):

# Adds a vortex/anyon at given center with fractional winding

for i in range(N):

for j in range(N):

angle = np.arctan2(i-center[0], j-center[1])

phase[i,j] += charge \* angle

return phase

# Add the anyons

phase = inject\_anyon(phase, center1, fraction)

phase = inject\_anyon(phase, center2, fraction)

# --- Braiding Evolution ---

# Move anyon 2 in a circle around anyon 1

r = N//3

theta\_vals = np.linspace(0, 2\*np.pi, steps)

global\_phase = []

for t, theta in enumerate(theta\_vals):

# Position of moving anyon

x = int(center1[0] + r \* np.cos(theta))

y = int(center1[1] + r \* np.sin(theta))

# Reset phase, inject static anyon, move the other

phase\_braid = np.zeros\_like(phase)

phase\_braid = inject\_anyon(phase\_braid, center1, fraction)

phase\_braid = inject\_anyon(phase\_braid, (x, y), fraction)

# "Global phase": sum over the field, modulo 2pi

tot\_phase = np.mod(np.sum(phase\_braid), 2\*np.pi)

global\_phase.append(tot\_phase)

# --- Plotting ---

fig, axs = plt.subplots(1, 2, figsize=(13, 4))

axs[0].imshow(np.cos(phase), cmap="twilight", origin='lower')

axs[0].set\_title("Final MBT Phase Field (cos)")

axs[0].plot(center1[1], center1[0], 'wo', markersize=9, label="Fixed Anyon")

axs[0].plot(center2[1], center2[0], 'mo', markersize=9, label="Moving Anyon (start)")

axs[0].legend()

axs[1].plot(theta\_vals/(2\*np.pi), global\_phase, lw=2, label="Braiding Phase")

axs[1].axhline(fraction\*2\*np.pi, color="orange", ls="--", label=f"Expected (×{fraction})")

axs[1].set\_xlabel("Fraction of full braid (turns)")

axs[1].set\_ylabel("Global Phase Accumulation")

axs[1].set\_title("MBT Anyon Braiding Phase Accumulation")

axs[1].legend()

plt.tight\_layout()

plt.show()