Project-2

Results

Case -1

```
[bidhanpoudel]@olympus ~> (22:10:38 03/19/24)
:: python Project_2.py
Total number of Walks: 200
Step size: 1
Total charge on the ground: -17.86412576
Run time: 0.777535915375 seconds
```

Case -2

```
[bidhanpoudel]@olympus ~> (22:10:59 03/19/24)
:: python Project_2.py
Total number of Walks: 200
Step size: 0.5
Total charge on the ground: -17.4213964971
Run time: 11.9134149551 seconds
```

Case -3

```
[bidhanpoudel]@olympus ~> (22:11:26 03/19/24)
:: python Project_2.py
Total number of Walks: 200
Step size: 0.25
Total charge on the ground: -16.3145733397
Run time: 199.678235054 seconds
```

Source Code

```
import numpy as np
import time

# Start the timer
start_time = time.time()

# Domain parameters
X, Z = 10, 10  # Domain size
Xmin, Xmax, Zmin, Zmax = 4, 6, 4, 6  # Conductor boundaries
h = 0.25  # Discretization step

# Grid initialization
nx, nz = int(X/h) + 1, int(Z/h) + 1

V = np.zeros((nz, nx))  # Potential matrix

N_walks = 200  # Number of random walks per point

np.random.seed(2)

# Function to perform a random walk from a starting point
def random_walk(x, y):
    while True:
        # Random step: 0=left, 1=right, 2=up, 3=down
        step = np.random.randint(0, 4)
        if step == 0:
            x -= 1
        elif step == 1:
            x += 1
        elif step == 2:
            y -= 1
        elif step == 3:
            y += 1

        # Check for boundary conditions
        # Reflective boundaries
```

```
        if x <= 0: x = 1
        if x >= nx-1: x = nx-2
        if y >= nz-1: y = nz-2

        # Ground (0V) boundary
        if y == 0:
            return 0  # Potential is 0V

        # Conductor (1V) boundary
        if Xmin <= x*h <= Xmax and Zmin <= y*h <= Zmax:
            return 1  # Potential is 1V

# Main simulation loop
for j in range(1, nz):
    for i in range(0, nx):
        if not (Xmin <= i*h <= Xmax and Zmin <= j*h <= Zmax):  # Skip the
conductor area
            V[j, i] = np.mean([random_walk(i, j) for _ in range(N_walks)])
        else:
            V[j, i] = 1  # Set the potential to 1V in the conductor area

eps = 1.0  # Relative permittivity of the medium
eps0 = 8.854585259  # Permittivity of vacuum

# Calculate the electric field normal to the ground
E_ground = 0
for i in range(1, nx-1): # Exclude the extreme left and right boundaries
    E_ground +=  V[0, i]-V[1, i]    # Potential difference between ground
and row above

# For the extreme boundaries, use only half the potential difference

E_ground += 0.5 * ( V[0, 0]-V[1, 0] ) + 0.5 * ( V[0, nx-1]-V[1, nx-1] )

# Compute the charge on the ground
Q_ground = E_ground * eps0 * eps

# End the timer and calculate the run time
end_time = time.time()
run_time = end_time - start_time

# Result output
print("Total number of Walks: {}".format(N_walks))
```

```
print("Step size: {}".format(h))
print("Total charge on the ground: {}".format(Q_ground))
print("Run time: {} seconds".format(run_time))
```