Assignment-4

Sanidhya Mahale

February 2023

We solve the given problem by following the steps given below in python code:
Step 1: We import the libraries random, matplotlib.pyplot ,numpy and csv.

Step 2: We define the outer square domain as 20 and circle radius(c) as 8.

Step 3: We generate arrays of evenly spaced x and y in required domain size using .linspace method, then we create a grid of the x and y vlaues obtained using .meshgrid method.

Step 4: Construct a circle of given radius

Step 5: Define a function move_particle(position), where position is (x,y)

Step 6: To randomly choose the direction for particle we list all the possible choices of appropriate length in +x, -x, +y, -y directions and use .choice method in random library to move the particle in any of the directions mentioned.

Step 7: Next we define if statements for the time when the particle exceeds its x or y limit beyond domain, it can come to the opposite side, then take (x,y) back from the function.

step 8: We define a function generate_path(num_steps) to generate particle path. Then we set number of choices as 4 in n_points. Then we define random angles between 0 and 2π. We plot the points using random angles generated by parameterizing the circle in terms of c , random angles using .cos and .sin methods in numpy module

Step 9: We make a 'points' list containing 4 random points generated on a circle and make a list position in which .choice in random module randomly chooses 1 starting point out of 4 choices

Step 10: We use the move_particle function on initial random position to generate the next random position of the particle

Step 11: We append the new position containing 1 point to empty array particle_path initialised at the start of function generate_path, and ask it to return to us.

Step 12: By defining num_steps as 1000, we now generate three random paths path1, path2 and path3.

Step 13: While generating the paths we also save the repective paths as a list of coordinates in csv file using .write and .writerow methods in csv module.

Step 14: Create a new figure with a single set of axes which assigns axes to 'ax' and figure object to 'fig'. Now we use ax.set_aspect('equal') which is a method call on an instance ax of the matplotlib.axes.Axes class. to equalize scales of x and y axes

Step 15: Then we The ax.plot() function is used to plot a point on the ax object, where [pos[0] for pos in path1] and [pos[1] for pos in path1] are the x and y coordinates, respectively, of each point in path1, and assign it red colour be '-r'. Same for path2 and path3 also.

Step 16: Show the plot using .show method in matplotlib.pyplot module.

Step 17: Show the plot using show() function in matplotlib.pyplot.