CS5331: Aerial Computing
Summer II 2023

- Project number: Project #2 – Privacy-Preserving Drone Mobility
- Your name: Manvitha Aathmuri (R11847781)

This project is about the mobility of drone explained in two different models in a 2-dimensional space. The two models are,

i. Random Waypoint model
ii. Privacy Preserving model.

Let us briefly discuss on how to run the code and then get into details of RWP and PPR models.

The Matlab code for this project is written using the skeleton from project #1's source code. The methodology from previous code has been understood and enhanced a lot with new functions and loops according to the requirements on creating RWP and PPR models. The use of the code has been clearly mentioned in comments inside the source code to understand it better. On a basic note, there are 4 input parameters given. They are, mobility_model, velocity, pausing time, number_of_dummy_locations.

Ex: **mobility(1,2,5,3)** is given in the command window and the code runs to plot the graph.

The two different mobility models that are implemented:

Random Waypoint (RWP) model: (when mobility_model = 0)

The Random Waypoint model generates a random destination within the network and moves the drone towards that destination. Once the drone reaches the destination, it pauses for the specified **pausing_time** and then selects a new random destination. This process is repeated for a given number of simulations (controlled by the variable **simulation**).

Privacy-Preserving Random (PPR) model: (when mobility_model = 1)

The Privacy-Preserving Random model is designed to enhance privacy by introducing dummy locations along with the actual destination. It generates multiple random dummy locations between the drone's current location and the actual destination. The drone then moves through these dummy locations in addition to the actual destination, simulating a more privacy-aware trajectory.

Let's go through the step-by-step analysis of what this program does.

i. Code begins with initiating the network dimensions. The network is a 1000x1000 square grid representing the available space for the drone's movement.
ii. Drone's starting location and the variables for Flying distance and Flying time are initialized.
iii. Each simulation step's time duration is specified by the time_step. Figure(1) creates a new figure window for visualization after clearing any previous figures with the clf("reset").
iv. A red diamond ('rd') indicates where the drone is at its starting point. The drone will run two simulation rounds because the variable simulation is set to 2.

v. Here begins the mobility model generation. If the mobility model is set to 0 (RWP), the drone will follow the RWP mobility model. A random destination (waypoint) is generated within the network boundaries.

vi. The Euclidean distance formula (norm function) is used to determine the distance between the drone's current location and the desired location. Based on the drone's velocity, the amount of time needed to travel there is computed.

vii. The total flying distance and time are updated with the current movement.

viii. Based on the flying time and time step, the total number of simulation steps needed to get there is determined. The incremental progress made in each step toward the goal is represented by the step_vector. Here, drone is being moved incrementally to the destination.

ix. In num_steps iterations, the drone is advanced incrementally in the direction of the goal. A blue line ('b-') is used to represent the movement, and a green asterisk ('g*') is used to represent the current position at each step. Between each step, a 0.1-second delay is included for visualization.

x. After reaching the destination, the drone's position is updated to the destination coordinates, and a pause is introduced for pausing_time seconds before moving to the next destination.

xi. Now enters the PPR model, The **distance_to_dest** is not used in the PPR model. Dummy locations are generated by selecting random locations between the drone's current position and the destination to calculate the distance to destination.

xii. The drone is moved through the dummy locations one-by-one and the overall flight duration and distance are updated appropriately. Each dummy location is preceded by a 1-second delay, after which the drone moves to its target and pauses for pausing_time seconds before continuing to the following simulation round.

xiii. After each round of simulation, the drone's movement and simulation details are visualized. The total flying distance and time for the two simulation rounds are displayed in the command window.

Results and Graphs:

Below is the screenshot of the result showing us the Total flying distance and Total flying time for both the models.

```
function mobility(mobility_model, velocity, pausing_time, num_dummy_locations)
    % Network dimensions
    network_width = 1000;
    network_height = 1000;

    % Initialize drone's starting location
    drone_x = randi([0, network_width]);
    drone_y = randi([0, network_height]);

    % Initialize variables to store total flying distance and time
    total_flying_distance = 0;
    total_flying_time = 0;

    time_step = 1;
    clf('reset')
    figure(1);
    plot(drone_x, drone_y, 'rd');
    hold on;
    simulation = 2;
    % Start the simulation
    while simulation>0
        % Generate random destination within the network
```

```
>> mobility(0,200,3,0)
Total Flying Distance: 604.2124 meters
Total Flying Time: 3.0211 seconds
>> mobility(0,70,3,0)
Total Flying Distance: 1269.2269 meters
Total Flying Time: 18.1318 seconds
>> mobility(1,200,3,2)
Total Flying Distance: 1302.8548 meters
Total Flying Time: 6.5143 seconds
>> mobility(1,70,3,10)
Total Flying Distance: 3609.146 meters
Total Flying Time: 51.5592 seconds
>>
```

When mobility model is 0, it generates RWP mobility as shown below.

mobility(0,200,3,0):
Total flying distance: 604.2 mts
Total flyting time: 3.02 sec

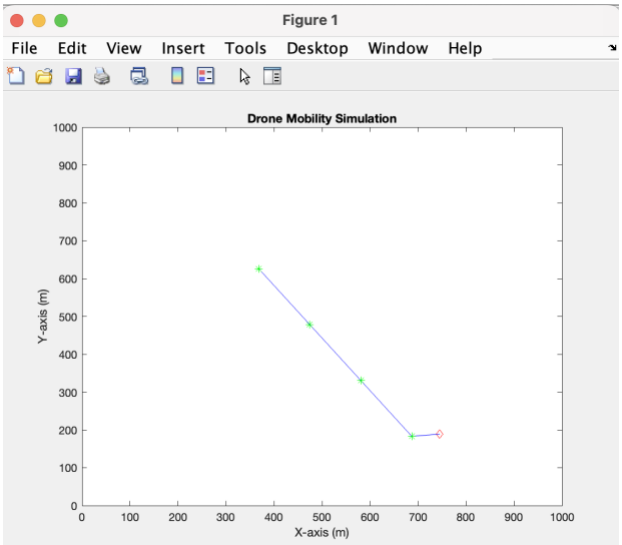

Fig (1)

mobility(0,70,3,0):
Total flying distance: 1269.2 mts
Total Flying time: 18.13 sec

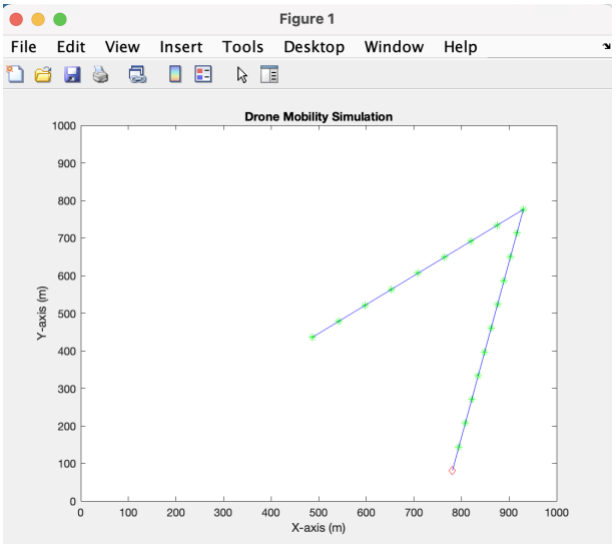

Fig (2)

When mobility model is 1, it generates PPR mobility as shown below.

mobility(1,200,3,2):
Total flying distance: 1302.8 mts
Total flyting time: 6.51 sec

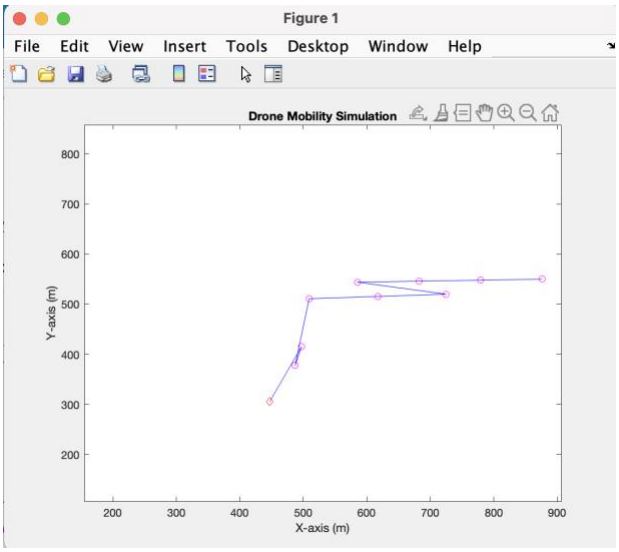

Fig (3)

mobility(1,70,3,10):
Total flying distance: 3609.1 mts
Total Flying time: 51.5 sec

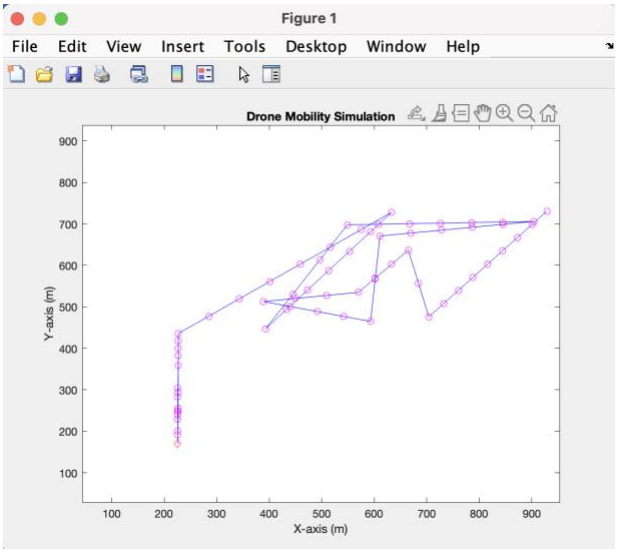

Fig (4)

Observations:

As discussed above, the graph depicts the movement of drone in RWP and PPR models in 2-dimensional space. The stars (*) represent the position of the drones in RWP model at different time steps during their movement. The rings represent the position of the drones in PPR model at different time steps during their movement along with their dummy locations. All the points are connected using a solid line.

There is an observation on the total flying distance of RWP and PPR models. From Fig (1) & Fig (3) we can observe that the distance has been increased as the path is straight line in RWP, whereas in PPR since we are using dummy locations the trajectory has become long and thus there is an increase in distance.

Fig (2) and Fig (4) shows us the drone movement if the velocity_parameter is decreased, which means the drone is moving slowly. As a result, movement of drone becomes slower & trajectories will be long, and it is clearly seen in the total distance between Fig (1) & Fig(2) in RWP and Fig (3) & Fig (4) in PPR.