Project number: 1

**Details of what this program does**:

Ans. by breaking down the code into snippets, here is the explanation

function[]=HelloDrone(number\_of\_lines\_parameter,velocity\_parameter)

* we defined a function HelloDrone(replaced Mobility) with 2 parameters as input

num\_lines = number\_of\_lines\_parameter;

* Assigned parameter value to varaiable num\_lines

start\_points = rand(1, 3);

* a random array of size 1x3 was created and assigned to start\_points varaible

x = zeros(num\_lines, 2);

y = zeros(num\_lines, 2);

z = zeros(num\_lines, 2);

* Three Matrices x,y,z are created by assigning them with (num\_lines x 2) size. Here zeros means the all the indexes in the matrix are filled by 0’s. these 3 will store the path cooridinates of the drones in 3-dimensional space

x(1,1) = start\_points(1);

y(1,1) = start\_points(2);

z(1,1) = start\_points(3);

* The random values generated and stored in array start\_points index 1,2,3 are assigned to (1,1) indexes of matrixes x,y,z respectively

for i = 2:num\_lines

x(i,1) = x(i-1,2);

y(i,1) = y(i-1,2);

z(i,1) = z(i-1,2);

x(i,2) = rand();

y(i,2) = rand();

z(i,2) = rand();

end

* A for loop was created with 2 initialization and incremented till num\_lines input. Here loop generates the path for remaining path co-ordinates. The rand function generates values between 0-1.
* Note: In matlab array will start from 1 whereas in other programming languages it starts from 0.

figure;

for i = 1:num\_lines

plot3(x(i,:), y(i,:), z(i,:));

hold on

end

* figure: creates a new figure window. A loop was created to plot values of x,y,z matrices. plot3 was used to create a 3D plot. “ : “ was used to specify all elements along the second dimension, means selecting all columns. hold on is used to retain plots in the current axes so that new plots added to the axes do not delete existing plots.

plot3([x(1,1), x(num\_lines,2)], [y(1,1), y(num\_lines,2)], [z(1,1), z(num\_lines,2)], 'k-', 'LineWidth', 2);

* an additional line is drawn between the first and last points to complete the path. It is coloured black with line width of 2.

point\_pos = [x(1,1) y(1,1) z(1,1)];

plot3(point\_pos(1), point\_pos(2), point\_pos(3), 'ro');

* point\_pos variable is assigned with the starting point of the drone path and it is circled red in plot(ro)

velocity = velocity\_parameter;

time\_step = 0.1;

total\_distance = 0;

* Velocity\_parameter is assigned to velocity variable and time\_step is set to 0.1, total\_distance variable is declared and assigned value 0.

for i = 1:num\_lines

dist = norm([x(i,2)-x(i,1) y(i,2)-y(i,1) z(i,2)-z(i,1)]);

total\_distance = total\_distance + dist;

time\_needed = dist / velocity;

num\_steps = ceil(time\_needed / time\_step);

step\_vector = [x(i,2)-x(i,1) y(i,2)-y(i,1) z(i,2)-z(i,1)] / num\_steps;

* a loop is created to calculate the distance between two points and assigned to variable dist. For each iteration Total distance was updated time\_needed was calculated to find the num\_steps, a step\_vector was defined to calucalte the displacement for each step.

for j = 1:num\_steps

point\_pos = point\_pos + step\_vector;

plot3(point\_pos(1), point\_pos(2), point\_pos(3), 'ro');

pause(0.1);

end

end

* another nested loop was created to plot red circles in the 3D plot. With a pause of 0.1 seconds to make it easy to observe the transation.

dist\_black = norm([x(1,1)-x(num\_lines,2) y(1,1)-y(num\_lines,2) z(1,1)-z(num\_lines,2)]);

total\_distance = total\_distance + dist\_black; % add distance to tota

* the black line distance was calculated by dist\_black and updated to total\_distance

num\_steps\_black = ceil(dist\_black / velocity / time\_step);

step\_vector\_black = ([x(num\_lines,2)-x(1,1) y(num\_lines,2)-y(1,1) z(num\_lines,2)-z(1,1)] / num\_steps\_black) \* -1; % multiply by -1 to move in opposite direction

* number of steps required to cover black line distance was calculated. A black step vector was calculated to move in the drone path’s opposite direction.

for j = 1:num\_steps\_black

point\_pos = point\_pos + step\_vector\_black;

plot3(point\_pos(1), point\_pos(2), point\_pos(3), 'ro');

pause(0.1);

end

* this loop was used to create a simulation of drone movement by plotting drone position in 3D plot.

fprintf('Total distance covered: %f meter\n', total\_distance);

xlabel('X-axis');

ylabel('Y-axis');

zlabel('Z-axis');

title('A mobility model of Drone');

* Total distance was displayed and the labeling for the 3D plot was done. Title was also added.

view(3);

rotate3d on;

* view(3) was used to display the plot in 3D view and ability to rotate was enabled

end

* end was used to close a function

Snapshots:

A screenshot of a computer

Description automatically generated

Fig 1. Number of lines is greater than velocity, both values are positive.

A screenshot of a computer

Description automatically generated

Fig 2. Number of lines is greater than velocity, both values are positive.

A screenshot of a computer

Description automatically generated

Fig 3. The number of lines is greater than velocity, both values are positive, and a large value was taken.

A screenshot of a computer

Description automatically generated

Fig 4. Velocity was used negatively to check the behavior of the plot and code.

A screenshot of a computer

Description automatically generated

Fig 5. The number of lines was taken as negative. It threw an error. Therefore, change the values to a velocity greater than the number of lines.

A screenshot of a computer

Description automatically generated

Fig 6. The velocity parameter was assigned to zero. This triggered an infinite loop and caused an error.

A screenshot of a computer

Description automatically generated

Fig 7

A screenshot of a computer

Description automatically generated

Fig 8. Number of lines was assigned as zero, it resulted in error and no plot was generated.