**Implementation of MATLAB code and user manual**

Here are the details of our farmland and simulation parameters. The size of the farmland is 250m \* 250m. In Australia, the farmland size is much bigger than that. We are using small land because we have only one drone that used to cover all nodes within time before draining the battery. We are using three types of ground node sensors such as soil, moisture and crop health. We have 20 soil, 20 moisture and 20 crop health sensors. The colour of soil node will be green, moisture node will he blue, crop health node will be red and base station will be black. The drone will fly from the ground station and cover all the sensors. Drone ground station will be outside the farmland and the distance is 100 meters.

We are using CW-30E Hybrid Gasoline & battery long flight time UAV that is latest and advance UAV device. Its battery capacity is almost 480 minutes and flying or cruise speed is 90 km/h (<https://www.jouav.com/products/cw-30e.html>). We will apply MATLAB code with model and calculate shortest path and time that takes drone to complete task.. We do two simulations in MATLAB one with Traveling Salesman Problem: Solver-Based (TSP) algorithms and the other without TSP model compare the differences between them. We do only one simulation without TSP and do multiple simulations with TSP while changing the number of nodes on ground field. The distribution of drones will be random because same areas of land needs more than 2 or 3 soil sensor, health sensors or moisture sensors. Some are do not every type of sensor there. MATLAB implementation code will decide the distribution of nodes in Farmland. Every time drone fly from base station it will return to base station after cover all nodes in farmland no matter how much nodes are there. Drone will stay 2 sec on every node to collect data. After completing the process, MATLAB will show the total distance travel by drone in kms and total time of drone to cover all nodes.

**Simulation Parameters.**

**Table 1:**

|  |  |
| --- | --- |
| Area | 62500 square meters (CW-30E Hybrid Gasoline & battery long flight time UAV)  10000 square meters (Phantom 4 Pro V2.0)  15000 square meters (Phantom 4 Pro V2.0) |
| Soil, moisture and crop health node | 3, 30, 60, 90, 300, 450 |
| Distribution nodes | Randomly |
| Nodes movement | Static |
| Flight Time | 480 minutes (CW-30E Hybrid Gasoline & battery long flight time UAV)  30 minutes (Phantom 4 Pro V2.0) |
| Drone Speed | 90 km/h (CW-30E Hybrid Gasoline & battery long flight time UAV)  45 km/h (Phantom 4 Pro V2.0) |
| Base station | 100 meters away from farmland |

|  |  |
| --- | --- |
| **Code and detailed description without TSP** | **Code and detailed description TSP** |
| % Define the size of the farmland  farmland\_size\_sqm = 62500; % in square meters  % Number of sensors for each type  num\_sensors = 20;  % Generate random coordinates for all sensors  all\_sensors\_x = rand(1, num\_sensors \* 3) \* farmland\_size\_sqm;  all\_sensors\_y = rand(1, num\_sensors \* 3) \* farmland\_size\_sqm;  % Plot all sensors  figure;  hold on;  % Plot soil sensors  scatter(all\_sensors\_x(1:num\_sensors), all\_sensors\_y(1:num\_sensors), 50, 'g', 'filled');  % Plot moisture sensors  scatter(all\_sensors\_x(num\_sensors+1:2\*num\_sensors), all\_sensors\_y(num\_sensors+1:2\*num\_sensors), 50, 'b', 'filled');  % Plot crop health sensors  scatter(all\_sensors\_x(2\*num\_sensors+1:end), all\_sensors\_y(2\*num\_sensors+1:end), 50, 'r', 'filled');  xlabel('X Coordinate (m)');  ylabel('Y Coordinate (m)');  title('Sensor Locations');  grid on;  % Plot base station (outside the field)  base\_station\_x = -100; % 100 meters away from the boundary  base\_station\_y = -100;  % Plot base station  scatter(base\_station\_x, base\_station\_y, 100, 'k', 'filled');  legend('Soil Sensors', 'Moisture Sensors', 'Crop Health Sensors', 'Base Station');  % Drone starts from the base station  current\_x = base\_station\_x;  current\_y = base\_station\_y;  % Initialize animation frames  frames = struct('cdata', [], 'colormap', []);  % Define maximum flight time in minutes  max\_flight\_time\_minutes = 480 ;  % Visit all sensors without revisiting any node  total\_distance = 0;  % Visit all sensors  for i = 1:num\_sensors \* 3  % Calculate distance to current sensor  distance\_to\_sensor = sqrt((current\_x - all\_sensors\_x(i))^2 + (current\_y - all\_sensors\_y(i))^2);    % Calculate total distance remaining in the mission  remaining\_distance = sqrt((all\_sensors\_x(i) - base\_station\_x)^2 + (all\_sensors\_y(i) - base\_station\_y)^2);  total\_remaining\_distance = total\_distance + distance\_to\_sensor + remaining\_distance;    % Calculate estimated time remaining  remaining\_time = total\_remaining\_distance / drone\_speed;    % Check if remaining flight time allows to return to base after visiting the next sensor  if remaining\_time > max\_flight\_time  % Plot drone movement to the sensor  plot([current\_x, all\_sensors\_x(i)], [current\_y, all\_sensors\_y(i)], 'm', 'LineWidth', 2); % Magenta color for drone path  drawnow;  frames(i) = getframe(gcf);    % Pause for 2 seconds to collect data at the sensor  pause(2);    % Update current position  current\_x = all\_sensors\_x(i);  current\_y = all\_sensors\_y(i);  end  % Return to the base station  plot([current\_x, base\_station\_x], [current\_y, base\_station\_y], 'm', 'LineWidth', 2); % Magenta color for drone path  drawnow;  frames(num\_sensors \* 3 + 1) = getframe(gcf);  % Calculate total distance traveled  total\_distance\_km = total\_distance / 1000; % Convert meters to kilometers  fprintf('Total distance traveled by the drone: %.2f kilometers\n', total\_distance\_km);  % Calculate estimated time taken  drone\_speed = 90; % meters per second (adjust according to your drone's speed)  time\_taken\_seconds = total\_distance / drone\_speed;  time\_taken\_minutes = time\_taken\_seconds / 60; % Convert seconds to minutes  fprintf('Estimated time taken by the drone: %.2f minutes\n', time\_taken\_minutes); | % Define the size of the farmland  farmland\_size\_sqm = 62500; % in square meters  % Number of sensors for each type  num\_sensors = 20;  % Generate random coordinates for all sensors  all\_sensors\_x = rand(1, num\_sensors \* 3) \* farmland\_size\_sqm;  all\_sensors\_y = rand(1, num\_sensors \* 3) \* farmland\_size\_sqm;  % Combine x and y coordinates  all\_sensors = [all\_sensors\_x; all\_sensors\_y];  % Plot all sensors  figure;  hold on;  % Plot soil sensors  scatter(all\_sensors\_x(1:num\_sensors), all\_sensors\_y(1:num\_sensors), 50, 'g', 'filled');  % Plot moisture sensors  scatter(all\_sensors\_x(num\_sensors+1:2\*num\_sensors), all\_sensors\_y(num\_sensors+1:2\*num\_sensors), 50, 'b', 'filled');  % Plot crop health sensors  scatter(all\_sensors\_x(2\*num\_sensors+1:end), all\_sensors\_y(2\*num\_sensors+1:end), 50, 'r', 'filled');  xlabel('X Coordinate (m)');  ylabel('Y Coordinate (m)');  title('Sensor Locations');  grid on;  % Plot base station (outside the field)  base\_station\_x = -100; % 100 meters away from the boundary  base\_station\_y = -100;  % Plot base station  scatter(base\_station\_x, base\_station\_y, 100, 'k', 'filled');  legend('Soil Sensors', 'Moisture Sensors', 'Crop Health Sensors', 'Base Station');  % Initialize animation frames  frames = struct('cdata', [], 'colormap', []);  % Define maximum flight time in minutes  max\_flight\_time\_minutes = 480 ;  **Here is the implementation of TSP**  % Visit all sensors using TSP (Nearest Neighbor Algorithm)  remaining\_nodes = 1:size(all\_sensors, 2);  path = []; % Start with an empty path  current\_x = base\_station\_x;  current\_y = base\_station\_y;  total\_time\_seconds = 0;  while ~isempty(remaining\_nodes)  % Find nearest sensor  distances = sqrt((current\_x - all\_sensors(1, remaining\_nodes)).^2 + (current\_y - all\_sensors(2, remaining\_nodes)).^2);  [~, nearest\_idx] = min(distances);  nearest\_node = remaining\_nodes(nearest\_idx);    % Add nearest sensor to path  path(end+1) = nearest\_node;  remaining\_nodes(nearest\_idx) = [];    % Update current position  current\_x = all\_sensors(1, nearest\_node);  current\_y = all\_sensors(2, nearest\_node);    % Pause for 2 seconds at each node for data collection  pause(2);  total\_time\_seconds = total\_time\_seconds + 2;    % Check if the total time exceeds the maximum flight time  if total\_time\_seconds > max\_flight\_time\_minutes \* 60  fprintf('Maximum flight time reached. Stopping drone.\n');  break;  end    % Plot drone movement to the sensor  if numel(path) > 1  plot([all\_sensors(1, path(end-1)), all\_sensors(1, path(end))], [all\_sensors(2, path(end-1)), all\_sensors(2, path(end))], 'm', 'LineWidth', 2); % Magenta color for drone path  drawnow;  frames(end+1) = getframe(gcf);  else  plot([base\_station\_x, all\_sensors(1, path(1))], [base\_station\_y, all\_sensors(2, path(1))], 'm', 'LineWidth', 2); % Magenta color for drone path  drawnow;  frames(end+1) = getframe(gcf);  end  end  % Return to the base station  plot([all\_sensors(1, path(end)), base\_station\_x], [all\_sensors(2, path(end)), base\_station\_y], 'm', 'LineWidth', 2); % Magenta color for drone path  drawnow;  frames(end+1) = getframe(gcf);  % Calculate total distance traveled  total\_distance = 0;  for i = 1:length(path)-1  total\_distance = total\_distance + sqrt((all\_sensors(1, path(i+1)) - all\_sensors(1, path(i)))^2 + (all\_sensors(2, path(i+1)) - all\_sensors(2, path(i)))^2);  end  % Add the distance from the last sensor back to the base station  total\_distance = total\_distance + sqrt((all\_sensors(1, path(end)) - base\_station\_x)^2 + (all\_sensors(2, path(end)) - base\_station\_y)^2);  % Convert total distance to kilometers  total\_distance\_km = total\_distance / 1000;  % Calculate estimated time taken  drone\_speed = 90; % meters per second (adjust according to your drone's speed)  time\_taken\_seconds = total\_distance / drone\_speed;  time\_taken\_minutes = time\_taken\_seconds / 60; % Convert seconds to minutes  % Display total distance traveled and estimated time taken  fprintf('Total distance traveled by the drone: %.2f kilometers\n', total\_distance\_km);  fprintf('Estimated time taken by the drone: %.2f minutes\n', time\_taken\_minutes); |

Here are the images of Farmland after drone covering the nodes with TSP and without TSP.

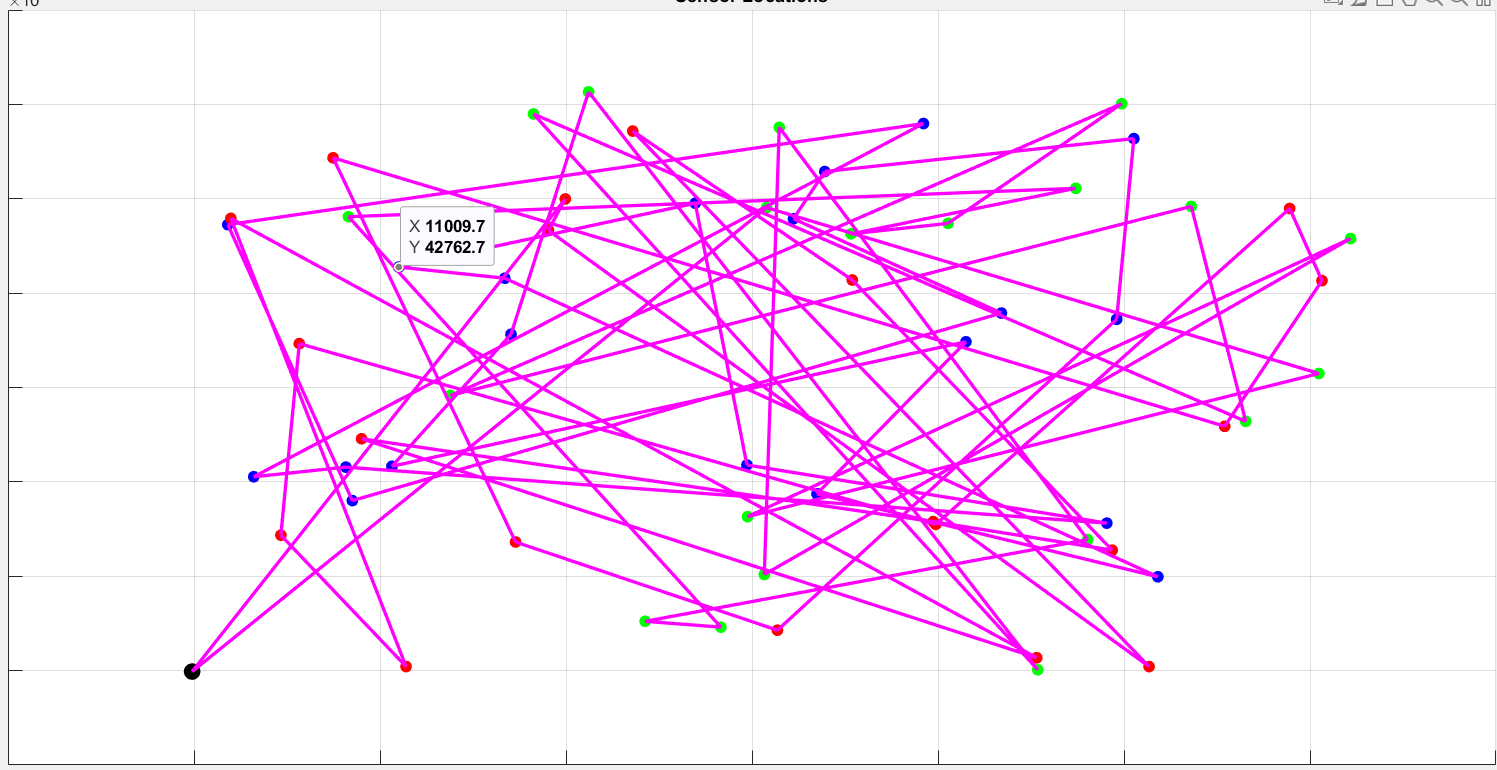


Figure 1: drone visiting nodes Without TSP algorithm code.

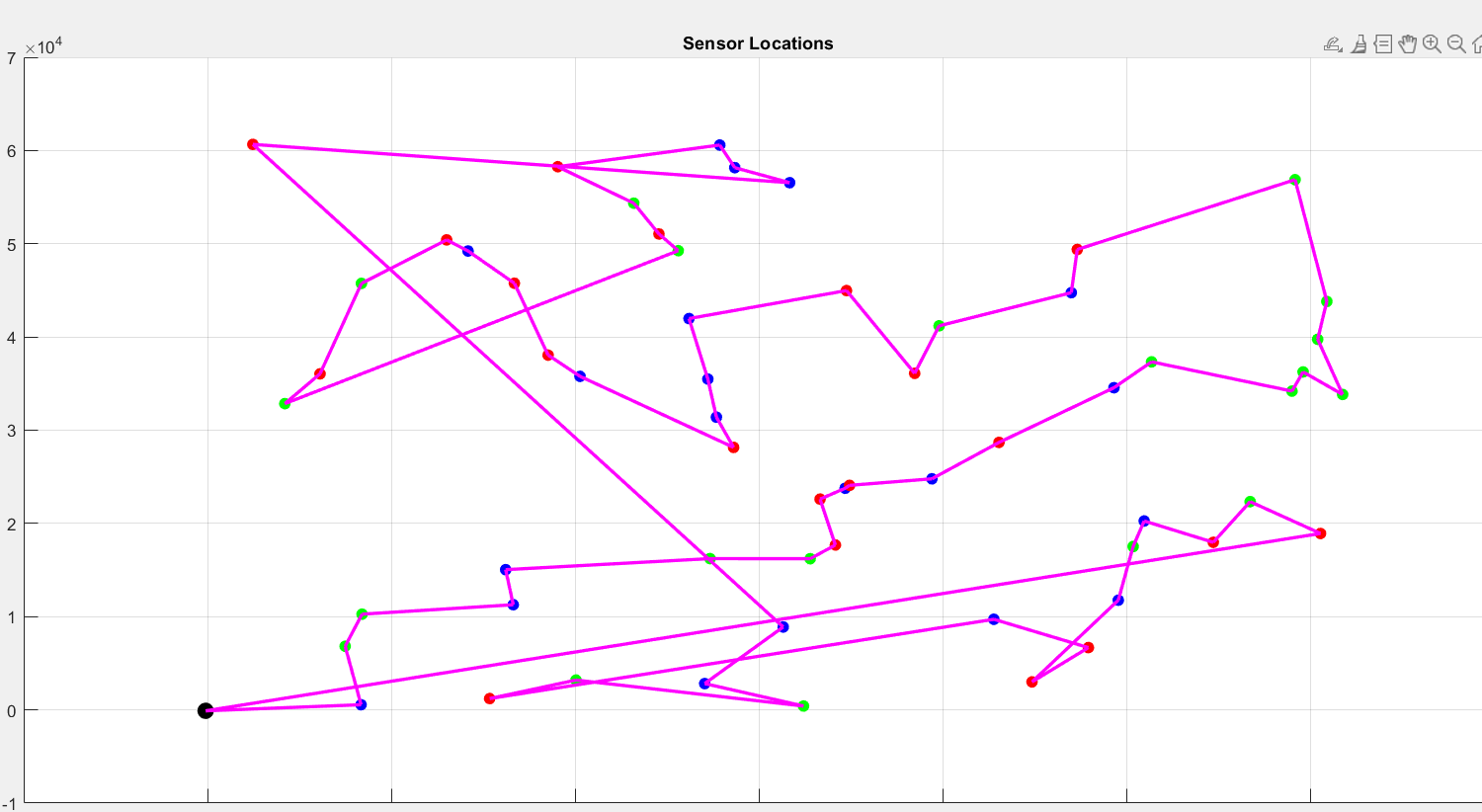
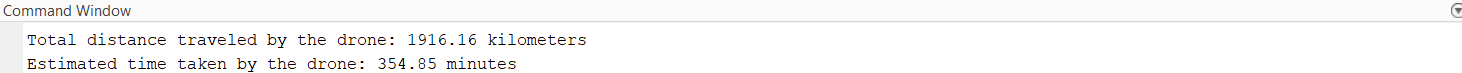


Figure 2: drone visiting nodes TSP algorithm code.

There is too much time and distance difference between them.

Drone time and distance without TSP algorithm.



Drone time and distance TSP algorithm.



Now we will run TSP algorithm multiple time with same parameters and setting but we change the number of nodes and check the difference between them.

**Table 2:**

|  |  |  |
| --- | --- | --- |
| **Number of nodes** | **Total distance (Km)** | **Total time (minutes)** |
| 3 | 125.36 | 23.21 |
| 30 | 381.17 | 70.59 |
| 60 | 528.94 | 97.95 |
| 90 | 551.86 | 102.20 |
| 300 | 1068.17 | 197.81 |
| 450 | 1230.23 | 227.82 |

The code of simulation with TSP and without TSP will be same but will change drone speed and maximum time according to the new drone capabilities. Now, we are using other type of drone that battery is limited to explain our idea. The parameters will be same, but the number of nodes and different farm size should be different because **Phantom 4 pro V2.0** has 30 minutes flight time. Because the farm size we used earlier was big and too much for whole who has 30 minutes flight time. No algorithm can cover this land within 30 minutes while using this drone.

We will use 2 different farm size such as 10000 square meters and 15000 square meters.

Here is the detailed description of Phantom 4 Pro V2.0.

**Table 3:**

|  |  |
| --- | --- |
| Weight | 1375 g |
| Diagonal size | 350 mm |
| Max service ceiling above sea level | 19685 ft (6000m) |
| Max wind speed resistance | 10 m/s |
| Max flight time | 30 minutes |
| Max Speed | S-mode: 45 mph (72 kph) |

**Source**: <https://www.dji.com/au/phantom-4-pro-v2/spe>

Here is the simulation without TSP and result.

Area 10000 square meters with 30 nodes.

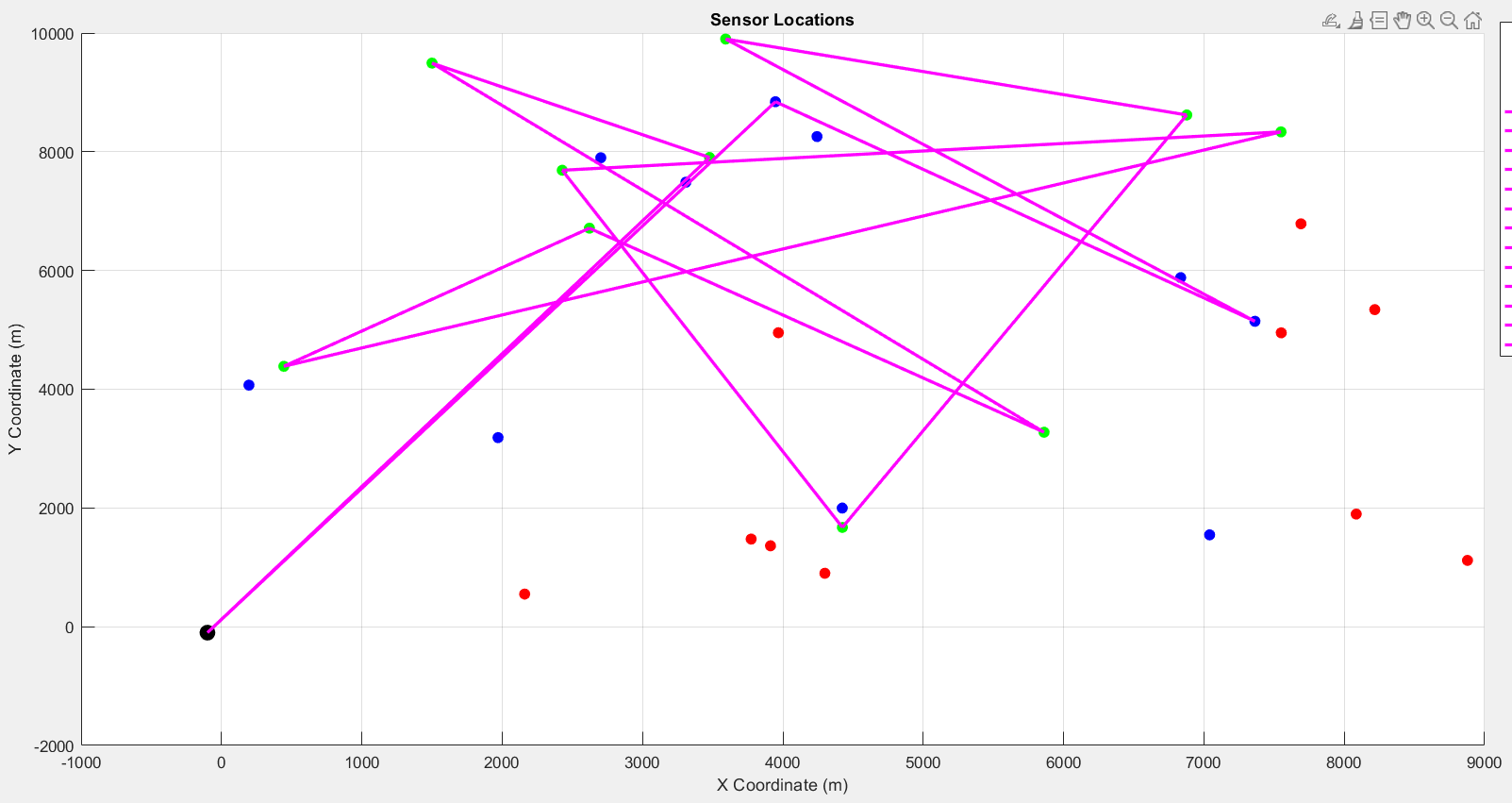


Figure3: Drone return to base station without completing all nodes.

Area 15000 square meters with 30 nodes.

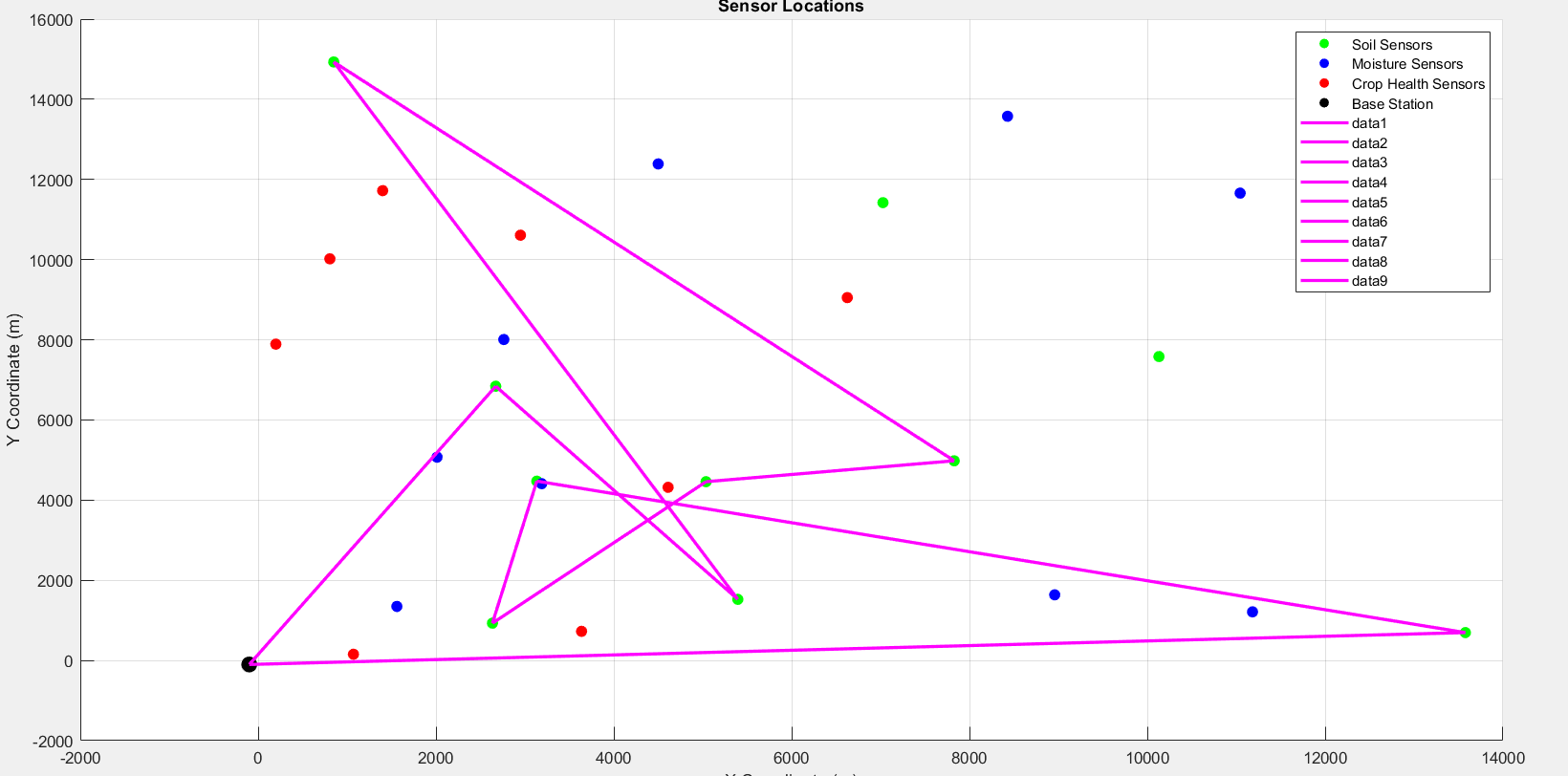


Figure 4: Drone return to base station without completing all nodes.

Result of 10000 square meters and 15000 square meters



Here is the simulation with TSP algorithm.

Area 10000 square meters with 30 nodes.

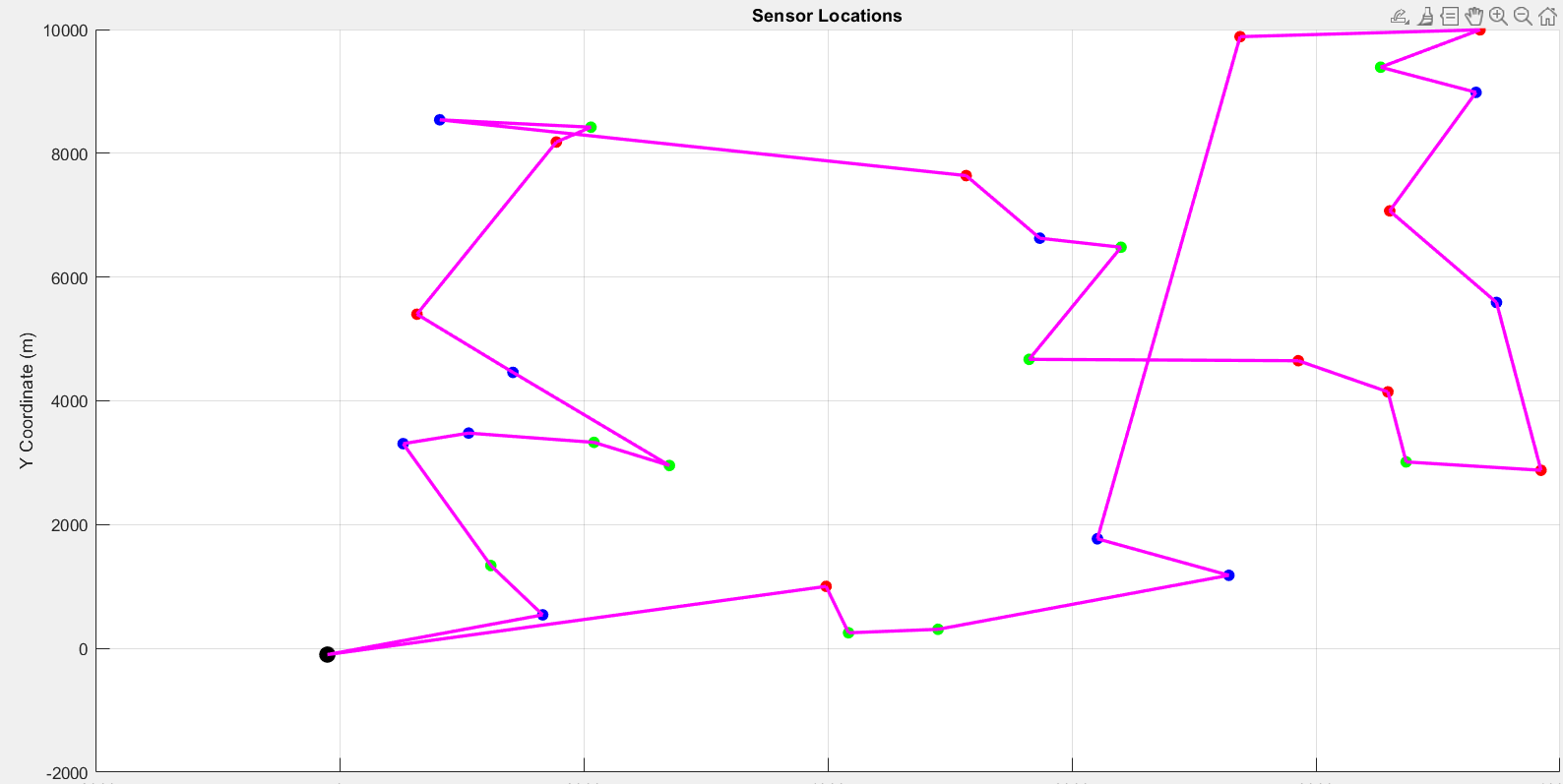


Figure 5: Drones cover all the nodes within time.

Result:



Drone travel 54.79 km and took 20.29 minutes to complete task.

Area 15000 square meters with 30 nodes.

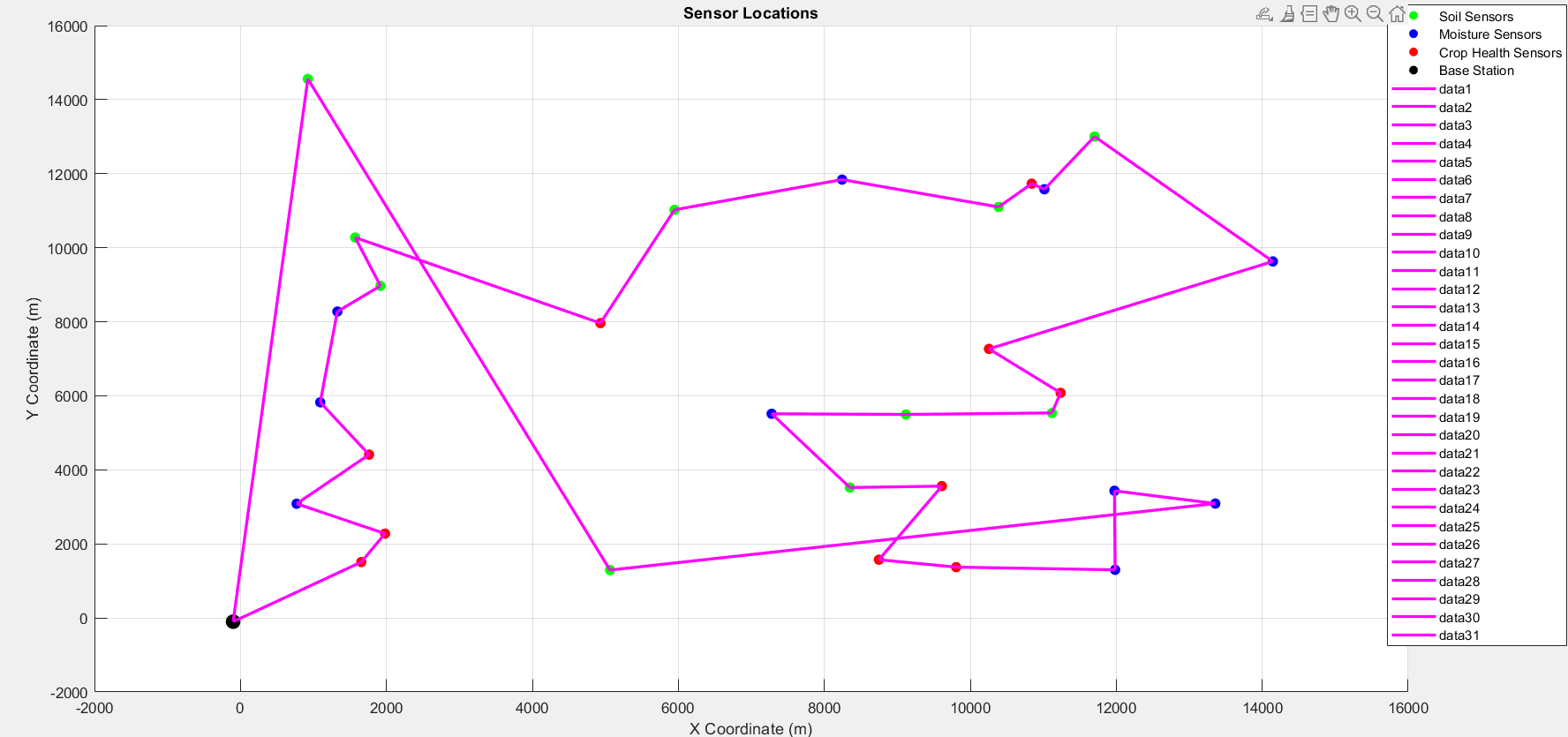


Figure 6: Drones cover all the nodes within time.

Result:



Drone travel 77.38 km and took 25.79 minutes to complete task.

Security and communication technologies are already explained in technology report.

**Difficulties and solutions**

From the beginning of this project, we faced many problems, but we address problems with time by discussing with tutor, coordinator, and supervisors. We also watched YouTube videos and read articles about MATLAB and simulation.

**Table 4:**

|  |  |
| --- | --- |
| **Problems** | **Solutions** |
| Confusion about the project requirements | Before we understood that, the project is about to collect data from node. But we were wrong. We have a meeting with tutor and supervisor, so they explain us the project is about to find a shortest path for drones to visit all nodes in farmland within minimum timeframe. |
| Creating team meeting with tutor and supervisor | That was our first time to create a zoom or Ms meeting. We do not know how to create a meeting and send invite to someone. We watched YouTube videos and address our issue. |
| Lack of knowledge about MATLAB and simulation | MATLAB and simulation were new tool for us. We check so many videos and article to understand the syntax and interface of MATLAB. MATLAB official website help us a lot to understand the hierarchy of this software. |
| MATLAB login and signup problem | When we confirm our project, it’s a big deal for us to download a sign-up MATLAB in our computers. University said that they will provide the software but when we try to sign up the MATLAB, it is not giving us conformation email code. Then we purchase software in $170 and install it. |
| Finding drones for simulation | There is multiple type of drones available in market. It was hard to find suitable drone for simulation. We used CW-30E Hybrid Gasoline & battery long flight time UAV for bigger land that’s area is 62500 sqm. The battery time of drone is 480 minutes. Everything was perfect.  One of our supervisors suggest us be more realist and use drones who has less battery power. So, we use Phantom 4 Pro V2.0 drone and its battery life are 30 minutes. We used 10000 sqm and 15000 sqm land for this drone because 62500 sqm was impossible for this drone to cover in single flight while using any shortest path algorithm. |
| Distribution of nodes in farmland | We were confused how to distribute nodes in farmland. We have 3 different type of ground nodes such as soil, moisture, and health care. We gave node different colours so we can differentiate the nodes in farmland. |
| Finding algorithm for simulation | One of biggest challenge that we face during this project is to find correct and appropriate algorithm for our simulation. We tried different algorithm such as practical swarm optimization (PSO) and extent practical optimization EPSO. They are very complicated and difficult. After getting some ideas form supervisors and YouTube, we use TSP algorithm to find shortest path. |