Assignment 1: Agent-Based Modelling

Title: Drone Swarm Disaster Recovery Simulation

Student Name: Rinilkumar Parmar
Student ID: 110209404
Course: Applied AI

1. Introduction

Agent-Based Modelling (ABM) is a computational method used to simulate interactions of autonomous agents within a defined environment. ABM enables researchers to study emergent behaviours and system dynamics arising from individual agent interactions. In this project, we model a **drone swarm disaster recovery system**, where autonomous drones search for, rescue, and transport victims to supply hubs in a disaster-struck environment. The objective is to analyse operational efficiency, emergent behaviours, and resource utilization under varying conditions.

2. Objectives

The objectives of this ABM project are:

1. **Understand ABM principles** and their application to disaster recovery scenarios.
2. **Design, implement, and analyse** a drone-based rescue simulation.
3. **Critically evaluate emergent behaviours**, such as victim discovery patterns and drone coordination.
4. **Communicate findings** using visualization tools, performance metrics, and sensitivity analysis.

3. Problem Definition

3.1 Agents

The simulation contains four primary agent types:

Agent Type	Role	Key Attributes
DroneAgent	Autonomous drones searching for victims, carrying them to hubs, and recharging	Battery level, state (search/deliver/recharge/failed), communication range, sensor probability
VictimAgent	Immobile disaster victims whose health decays over time	Health, rescued/found status

SupplyHubAgent	Locations where drones deliver rescued victims and recharge	Static hub, no active behaviour
ObstacleAgent	Environmental obstacles blocking drone movement	Static position

3.2 Environment

The environment is a **2D grid world** with configurable width and height. It contains:

- **Supply hubs**: Starting points and recharge/delivery stations.
- **Obstacles**: Blockages that drones must navigate around.
- **Victims**: Randomly placed in the grid, representing people requiring rescue.
- **Drones**: Initially positioned at hubs, exploring the environment to locate and rescue victims.

3.3 Rules & Behaviour

DroneAgent Rules

1. **Search**: Explore unvisited adjacent cells. Prioritize cells with detected victims.
2. **Sense Victims**: Detect victims in the current cell with a probability (sensor_prob).
3. **Rescue**: Pick up detected victims and deliver them to the nearest supply hub.
4. **Battery Management**: Return to a hub when battery falls below a threshold. If battery is depleted, drone fails.
5. **Communication**: Alert neighbouring drones within a specified range about detected victims.

VictimAgent Rules

- Health decays over time at a fixed rate unless rescued.
- Status updates to rescued when delivered to a hub.

SupplyHubAgent Rules

- Provide a recharge station for drones.
- Record delivered victims.

ObstacleAgent Rules

- Prevent drone movement into blocked cells.

3.4 Research Questions

1. How efficiently can a swarm of autonomous drones locate and rescue victims in a disaster environment?

2. How does varying drone numbers, communication range, or sensor success probability affect rescue success and coverage?
3. What emergent patterns occur in victim discovery and drone coordination?
4. How robust is the drone swarm system under environmental constraints such as obstacles and limited battery capacity?

4. Model Design & Implementation

4.1 Simulation Platform

- **Mesa (Python)** ABM framework.
- **CanvasGrid** for spatial visualization.
- **ChartModule** for tracking system-level metrics over time.

4.2 Agent Attributes & Parameters

- **DroneAgent**: Battery (steps), communication range, sensor probability, state, carrying victim.
- **VictimAgent**: Health (0–100), rescued/found flags.
- **SupplyHubAgent**: Fixed position.
- **ObstacleAgent**: Fixed position.

4.3 Interaction Mechanisms

- **Local interactions**: Drones alert neighbours within `comms_range`.
- **Movement**: Drones move towards unexplored or victim-containing cells.
- **Delivery**: Drones transport victims to nearest hub.

4.4 Key Parameters

Parameter	Default Value	Description
Grid Width	20	Environment width (cells)
Grid Height	20	Environment height (cells)
Number of Drones	6	Active autonomous agents
Number of Victims	8	Targets to rescue
Number of Supply Hubs	1	Recharge/delivery stations
Number of Obstacles	20	Environmental blockages
Drone Battery	100 steps	Maximum operational steps before recharge
Sensor Probability	0.9	Likelihood of detecting victim in current cell
Communication Range	2	Cells within which neighbors are alerted

5. Simulation Experiments & Results

5.1 Experimental Scenarios

1. **Baseline**: Default parameters as above.
2. **Scenario 1**: Increase drones to 12; assess impact on rescue rate and coverage.
3. **Scenario 2**: Reduce sensor probability to 0.6; assess missed victims and delays.
4. **Scenario 3**: Increase obstacles to 50; assess drone navigation and rescue efficiency.

5.2 Metrics Collected

- **Coverage**: Fraction of grid visited by drones.
- **Found**: Number of victims detected.
- **Rescued**: Number of victims delivered to hubs.
- **ActiveDrones**: Number of drones operational (battery > 0).

5.3 Observations

1. Increasing drone numbers improved coverage and rescue rates but introduced congestion around hubs.
2. Lower sensor probability delayed victim detection and reduced overall rescue success.
3. High obstacle density forced drones to navigate longer paths, reducing efficiency and battery life.
4. Emergent patterns included coordinated discovery through alerts, optimized paths to hubs, and clustering behaviour around victims.

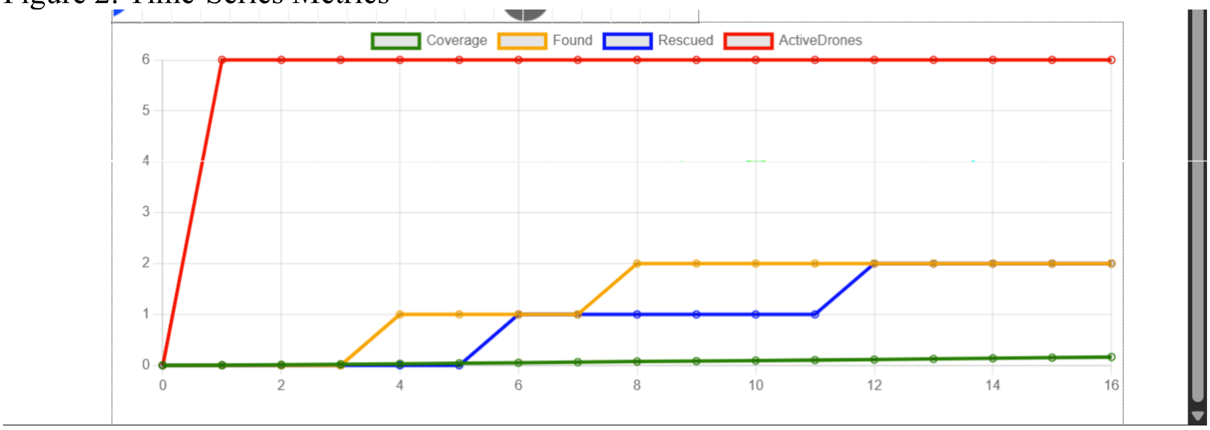
Visualization:

- Time-series plots tracked **coverage**, **found victims**, **rescued victims**, and **active drones**.
- Grid visualizations showed drone paths, victim locations, obstacles, and hubs, highlighting areas of high activity.

Figure 1: Simulation Grid Snapshot

- **Red circles:** Active drones
- **Green circles:** Drones on mission with sufficient battery
- **Cyan circles:** Rescued victims
- **Blue circles:** Supply hubs
- **Gray circles:** Obstacles
- **Red circles overlapping grey or victims:** Drones navigating around obstacles and moving towards victims

Figure 2: Time-Series Metrics



- **Green line:** Environment coverage over time
- **Orange line:** Victims found by drones
- **Blue line:** Victims rescued and delivered to supply hubs
- **Red line:** Number of active drones (with battery > 0)

6. Critical Reflection & Limitations

6.1 Model Insights

- The ABM accurately captures **autonomous agent coordination** and **emergent rescue behaviours**.
- Alerts between drones create **efficient swarm discovery patterns**, even in complex environments.

6.2 Limitations

1. Simplified movement and sensing rules may not capture real-world drone dynamics.
2. Obstacles are static and do not represent dynamic hazards.
3. Communication is limited to local interactions; long-range coordination is not modelled.
4. Health decay and battery consumption are linear, whereas real-world values may vary stochastically.

6.3 Future Extensions

- Introduce **dynamic obstacles** or disaster scenarios (e.g., spreading fire).
- Implement **advanced path planning algorithms** (A*, Dijkstra) for drones.
- Include **heterogeneous drone capabilities** and prioritization strategies.
- Incorporate **realistic communication networks** and failures.

7. Conclusion

This ABM demonstrates the potential of **drone swarms for disaster recovery**, highlighting how simple local rules can lead to **emergent cooperative behaviour** and effective victim rescue. The simulation provides a platform for evaluating system performance under varying parameters, offering valuable insights for designing robust autonomous rescue systems.

8. References

1. Mesa: Agent-Based Modelling in Python. (https://mesa.readthedocs.io/)