

## Meeting Outline:

- 1- Discussion about test results
- 2- Questions about LLM Magazine paper
- 3- Questions&Discussion about energy consumption calculations approach

1- Discussion about test results

## 2- Questions about LLM Magazine paper

- 1-) How is it different from connection to an internet network and use LLM tools?  
How can it be utilized?

### 3- Questions&Discussion about energy consumption calculations approach

$$E_{total} = E_{flight} + E_{hover} + E_{compute} + E_{comm}$$

- $E_{total}$  = Total energy consumed
- $E_{flight} = \alpha \times D \times v^2$
- $E_{hover} = \beta \times T_{hover}$
- $E_{compute} = \gamma \times C \times T_{compute}$
- $E_{comm} = \delta \times T_{comm}$

# Pseudocode for Actions for Different Energy Modes

```
class UAV:
    def __init__(self, energy, mode='mid'):
        self.energy = energy
        self.mode = mode # 'low', 'mid', 'high'
        self.alpha = 0.5 # Flight energy coefficient
        self.beta = 0.3 # Hover energy coefficient
        self.gamma = 0.2 # Computation energy coefficient
        self.delta = 0.1 # Communication energy coefficient
        self.mode_factors = {'low': 0.8, 'mid': 1.0, 'high': 1.2}

    def calculate_energy_consumption(self, distance, hover_time, computation_time, comm_time):
        flight_energy = self.alpha * distance * (2 ** 2) # Example velocity = 2
        hover_energy = self.beta * hover_time
        compute_energy = self.gamma * computation_time
        comm_energy = self.delta * comm_time

        total_energy = flight_energy + hover_energy + compute_energy + comm_energy
        mode_adjusted_energy = total_energy * self.mode_factors[self.mode]
```

```

class UAV:
    def _init_(self, energy):
        self.energy = energy
        self.mode = self.determine_mode()
    def determine_mode(self):
        if self.energy > 70:
            return "High Mode"
        elif 30 <= self.energy <= 70:
            return "Mid Mode"
        elif 10 <= self.energy < 30:
            return "Low Mode"
        else:
            return "Critical Mode"
    def act_according_to_mode(self):
        mode = self.determine_mode()
        if mode == "High Mode":
            self.aggressive_task_handling()
            self.explore_new_areas()
        elif mode == "Mid Mode":
            self.balanced_task_handling()
            self.move_to_mid_energy_zones()
        elif mode == "Low Mode":
            self.energy_saving_mode()
            self.head_to_nearest_station()
        elif mode == "Critical Mode":
            self.emergency_return()

```

### **1. High Energy Mode (Energy > 70%)**

- Aggressive Coverage: UAV actively searches for new users/tasks to offload.
- Long-Distance Mobility: UAVs prioritize high-demand zones even if far away

### **2. Mid Energy Mode (Energy 30% - 70%)**

- Balanced Behavior: UAVs focus on efficiently completing queued tasks.
- Controlled Movement: UAVs prioritize movement to locations with nearby charging stations or edge servers.

### **3. Low Energy Mode (Energy < 30%)**

- Energy Conservation: UAVs avoid high-speed movement and prefer hovering closer to charging points.
- Recharge Seeking: UAVs actively move toward the nearest charging station.

### **4. Critical Energy Mode (Energy < 10%)**

- Emergency Return: UAV halts all tasks and navigates directly to the nearest charging station or designated safe landing point.

