Drone Energy & Routing Simulation — README

Overview

This project simulates the behaviour, energy usage, and routing decisions of a small electric delivery drone operating in a simplified 10×10 urban grid. It combines:

- **Physics-based energy models** (ODE-driven take-off/landing, analytical/simulated cruise power)
- **Routing via Dijkstra's algorithm**
- **Battery-aware trip planning** with per-trip recharging
- **Custom numerical tools** such as linear interpolation and incremental root-finding
- **Extensive plotting utilities** for visualising energy usage, paths, and performance metrics

The simulation demonstrates how payload mass, path geometry, and flight energetics constrain a drone's ability to complete multiple delivery trips.

Project Structure

File	Description
main.py	Entry point; runs grid setup, ODE tables, root-finding demos, routing, and all major plots.
ode.py	Contains physical constants, energy/flight models, and ODE definitions for take-off, landing, and forward flight. Also includes cruise-energy solver and mass-scaled energy helpers.
root_finding.py	Custom incremental search + bisection solver; linear regression; exponential fitting; used for various numerical tasks.
interpolation.py	Simple piecewise linear 1D interpolator used for smoothing ODE outputs and trajectory plotting.
plotting.py	All visualisation utilities: take-off profile, payload vs. delivered mass, range/endurance, grid paths, and continuous interpolation.
read me.py	Original short explanation of project purpose (superseded by this README).

How to Run the Simulation

1. Install Requirements

This project uses:

- numpy
- matplotlib
- scipy

2. Run the Simulation

Execute:

python main.py

You will see console output (trip summaries, root-finding results) and several pop-up plots.

3. Optional: Reproducible Runs

In **main.py**, set:

seed = (an integer value)

to make the randomly generated grid layout repeatable.

Simulation Workflow

This is the sequence executed in main.py:

1. Grid Generation

- Random warehouse location
- Random delivery cells with payloads (1–3 kg)
- Two random no-fly zones
- Printed ASCII grid map for reference

2. Energy Model Demonstrations

- Print table of takeoff/landing energy vs payload
- Compute and print crossover speeds (where drag = induced power)
- Plot takeoff altitude and energy profile (uses ODE + interpolation + root finding)
- Plot endurance and range vs payload (with exponential curve fits)

3. Routing & Trip Planning

The drone:

- Can carry up to **10 kg** per trip
- Starts each trip with a **fully recharged battery** (≈0.0799 kWh)
- Uses Dijkstra-based shortest paths
- Ensures energy margin for return + landing before committing to a delivery
- Performs as many delivery trips as the battery and layout allow

Console output provides step-by-step trip logs.

4. Visualisations

Includes:

- Take-off ODE profile
- Payload vs total delivered mass
- Range/endurance vs payload
- Grid paths with nodes coloured by remaining battery
- Continuous interpolated trajectory for first trip

Key Algorithms & Models

Energy Models (in ode.py)

- **Take-off & landing ODEs:** thrust control, vertical drag, induced velocity, and electrical power.
- **Cruise energy:** integrated ODE with constant cruise tilt; fallback to analytical power model.
- **Power model:** induced + profile drag + fixed power.
- **Energy scaling:** takeoff/landing/cruise energy scaled linearly with mass.

Routing

- Full-grid Dijkstra per key point (warehouse + deliveries)
- Feasible-delivery selection based on energy margin
- Greedy nearest-delivery heuristic
- Per-trip battery logging

Custom Numerical Tools

- Incremental+binary-search root finder (no SciPy)
- Linear 1D interpolation
- Exponential regression using log-linear transform

Inputs & Outputs

Inputs (implicit/randomised)

- Grid size (10×10)
- Random warehouse and delivery locations
- Delivery payloads (1–3 kg)
- Random no-fly cells
- Drone parameters (mass, drag, battery, etc.) in ode.py
- Cruise speed and battery capacity in main.py

To customise, edit variables at the top of main.py.

Outputs

Console Output

- Grid layout
- Energy table (take-off/landing for sample payloads)
- Root-finding speed crossover results
- Detailed per-trip logs:
    - Moves, distances, loads, energy costs
    - Battery state after each step
    - Delivered payload
    - Full-trip summaries

Plots

1. Take-off altitude & battery profile (ODE + interpolation)
2. Payload vs total delivered mass per battery

3. Range & endurance vs payload (plus exponential fits)
4. Grid map with coloured drone paths
5. Interpolated continuous flight trajectory

Extending the Project

Potential enhancements:

- Replace simple control laws with more advanced autopilot modelling
- Implement multi-drone coordination
- Add stochastic wind or weather models
- Add 3D obstacles instead of 2D no-fly cells
- Use A* instead of Dijkstra for speed
- Model package pickup/return payload changes per delivery