Drone Delivery with Battery Limits - Report

Course: CEN 352 - Artificial Intelligence
Done by: Klevi Vezi
GitHub Repo: https://github.com/KleviV04/AI_assignment_Klevi_Vezi

1. Original Domain

This project models a drone delivery system that works with limited battery power on a small grid map. The drone must deliver several packages to certain locations while dealing with wind zones, battery use, and recharge stations.
Each move uses a different amount of energy depending on:
→ The wind level of that cell (0–2).
→ The weight of remaining packages (heavier means more battery use).
The drone can stop at recharge stations, but each one takes a different amount of time to refill the battery. The goal is to deliver all packages using as little total cost as possible.

State Representation

(x, y, battery, delivered_mask)

x, y → drone position
battery → remaining charge
delivered_mask → tracks which packages are already delivered

Actions

MOVE → move in 4 directions (if enough battery).
DELIVER →: deliver the package if standing on a delivery spot.
RECHARGE → recharge to full capacity at a station (adds recharge-time cost).

Transition and Cost

→ Dynamic energy per move:
energy = 1 + wind[x,y] + 0.05 * remaining_payload_weight

This makes the world more realistic — wind makes flying harder, and carrying more packages drains energy faster.
→ Delivering has no cost.
→ Recharging fills the battery and adds time based on the station speed.

Goal → Deliver all packages (order doesn’t matter)

2. Algorithms Implemented

Two search strategies were implemented and compared:

Uniform Cost Search (UCS) → an uninformed search that expands by total path cost g. It's always optimal but explores more states because it doesn't use any heuristic.

A* → an informed version that expands by f = g + h, using this heuristic:

h = nearest(current → remaining) + MST(remaining)

MST = minimum spanning tree of remaining delivery points, using Manhattan distance

→ The heuristic ignores wind and recharge times, so it never overestimates cost - this makes it admissible and safe to use. UCS was used as a baseline, while A* shows how using a heuristic can make the search faster and more efficient.

3. Completeness, Optimality, and Complexity

Property	**UCS**	**A***
Completeness	Complete (finite grid + costs >= 0)	Complete
Optimality	Optimal (non-negative costs)	Optimal (admissible heuristic)
Time/Space complexity	Exponential in number of deliveries	Exponential but explores fewer states

4. Experiments

Grid size = 8×6

Deliveries = 3 locations

Battery capacity = 20

Wind = random (0–2)

Recharge stations = 3 (variable times: 2, 4, 1)

Weight per package = 5.0

Seed	**Algorithm**	**Nodes expanded**	**Nodes generated**	**Max Frontier**	**Time (s)**	**Total path cost**
42	UCS	818	1034	217	0.0746	42.8
42	A*	520	648	148	0.0948	42.8
84	UCS	581	769	189	0.0540	47.8
84	A*	401	455	89	0.0776	47.8

→ Across both tests (seeds 42 and 84), A* found the same optimal cost as UCS (42.8 and 47.8) but with far fewer node expansions - around 25–40% less.

This proves that the heuristic helps guide the search while still finding the best path.
The total cost changed between the two seeds because the random wind and recharge setup made the environment harder in the second test.
→ Both algorithms had short runtimes, but A* would perform much better as the grid, wind, or number of deliveries increase.

5. Analysis and insights

This problem mixes both pathfinding and resource management.
Key design decisions:
→ The dynamic energy use makes movement depend on weight and wind.
→ Different recharge speeds add planning choices (fast but far vs. slow but close).
→ The heuristic ignores wind so it stays safe (admissible) but still useful.
These elements make the domain original and practical - similar to how real delivery drones would plan their routes.

6. Reproducibility

→ Single-file Python implementation (drone_solver.py)
→ Deterministic random seed for fair comparisons
→ No extra libraries needed
Run command : drone_solver.py

7. References

Russell, S. & Norvig, P. *Artificial Intelligence: A Modern Approach* (3rd Ed.).
Pearl, J. *Heuristics: Intelligent Search Strategies for Computer Problem Solving.*
Original contribution → dynamic energy model + variable recharge stations integrated into grid-based search.

8. Conclusion

→ This project shows how search algorithms can be used in real-life style problems where energy and environment matter. Both UCS and A* always found the best plan, but A* did it with much less search effort. The results show how a simple heuristic can make a big difference in efficiency without losing accuracy.