AeroHack Technical Report Draft - Page 1/5

AEROHACK TECHNICAL REPORT (SUBMISSION DRAFT)

1. PROBLEM STATEMENT

We built a single mission planning + simulation framework that solves:

1. Aircraft constrained waypoint mission under wind, energy/endurance, maneuver, and geofence constraints.
2. Spacecraft 7-day observation/downlink mission under visibility, slew, battery, and duty-cycle constraints.

The same shared planning engine is used for both domains.

2. UNIFIED FORMULATION

2.1 SHARED DECISION ABSTRACTION

Both domains are represented as a directed task graph:

- Task `i`: time window `[t_i^{min}, t_i^{max}]`, duration `d_i`, value `v_i`, metadata.
- Transition `(i,j)`: travel/slew feasibility, transition time `\tau_{ij}`, and energy/resource effect.
- State `x_k`: mission time + resource vectors after k-th action.

2.2 SHARED CONSTRAINT API

Each candidate step is validated through the same interface:

- `Constraint.evaluate(state_before, state_after, from_task, to_task, transition, step_meta)`

Hard constraints (violations rejected):

- Time-window feasibility
- Resource bounds
- Domain safety/feasibility (aircraft geofence/turn limits, spacecraft slew/duty/power proxies)

2.3 SHARED OBJECTIVE

For each step:

$$
\Delta J = w_v v_i + w_d \Delta D - w_t \Delta t - w_e \Delta E
$$

Terminal adjustments reward mission completion and penalize unfinished/non-terminal plans.

2.4 SHARED SOLVER

The same solver stack is used for both domains:

- `UnifiedPlanner`: discrete resource-constrained sequence search (beam/greedy/multistart).
- `UnifiedHybridOCPEngine`: unified hybrid optimal-control refinement on top of mission sequences:
  - direct-shooting continuous control variables per transition,
  - robust multi-scenario objective (`worst_case`),
  - discrete sequence mutation and candidate selection.
  - backend path: `scipy` coordinate refinement by default, with optional `casadi/ipopt` refinement under the same interface when available.

In `planner_strategy: auto_best`, both are executed and the better feasible plan is selected.

3. AIRCRAFT MODEL

State:

$$
[x, y, h, \psi, t, E]
$$

Transition model:
- Geofence-aware routing using visibility-graph shortest paths with safety margin.
- Wind-aware ground speed projection on each route segment.
- Turn-time from heading change with combined turn-rate and bank-angle feasibility.
- Climb feasibility from altitude delta and climb-rate bound.

Hard constraints:
- Geofence/no-fly polygons (segment rejection)
- Mission horizon
- Energy reserve
- Altitude bounds
- Turn-rate bound

Output artifacts:
- `outputs/aircraft/uav_flight_plan.csv`
- `outputs/aircraft/uav_route_segments.csv`
- `outputs/aircraft/uav_path.png`
- `outputs/aircraft/uav_path.kml`
- `outputs/aircraft/uav_energy_profile.png`
- `outputs/aircraft/uav_constraint_summary.json`
- `outputs/aircraft/uav_constraint_certification.csv`
- `outputs/aircraft/uav_constraint_certification.json`

4. SPACECRAFT MODEL

State:

$$
[t, B, Q, S, D]
$$

where `B` battery, `Q` data buffer, `S` science buffer value, `D` delivered science.

Opportunity generation:
- 7-day horizon
- Two-body circular orbit propagation in ECI with J2-driven RAAN precession
- ECI/ECEF conversion with Earth rotation
- Epoch-aware Sun vector for eclipse-aware charging
- Elevation-based downlink windows and LOS/off-nadir observation windows
- Observation tasks + ground-station downlink tasks

Transition model:
- Slew time from pointing-angle delta and slew-rate
- Nominal slew precomputation uses orbit-time-dependent pointing vectors (not static target-

target geometry)
- Task start = `max(current_time + slew_time, window_start)`
- Battery proxy integrates eclipse-aware charging, solar capture efficiency, and loads over idle/task intervals
- Observation science uses diminishing returns on repeat target visits (configurable decay + floor)

Hard constraints:
- Battery min/max bounds
- Data-buffer bounds
- Max operations per orbit
- Slew feasibility
- Time windows

Output artifacts:
- `outputs/spacecraft/spacecraft_7day_schedule.csv`
- `outputs/spacecraft/visibility_windows.csv`
- `outputs/spacecraft/spacecraft_gantt.png`
- `outputs/spacecraft/spacecraft_resources.png`
- `outputs/spacecraft/spacecraft_schedule.kml`
- `outputs/spacecraft/spacecraft_constraint_summary.json`
- `outputs/spacecraft/spacecraft_constraint_certification.csv`
- `outputs/spacecraft/spacecraft_constraint_certification.json`

5. VALIDATION
Validation pipeline (`src/validation/run_validation.py`) includes:
- Aircraft Monte Carlo with wind/battery perturbations
- Spacecraft Monte Carlo with battery/solar perturbations and configurable epoch-timing perturbations on a subset of runs
- Baseline comparison: beam vs greedy with same unified solver and data
- Stress scenarios for both domains (tight wind/energy and low solar/battery cases)
- Pareto trade studies for both domains
- Deterministic replay hash certification
- Independent post-simulation audits (separate from planner constraints) for both domains
- Constraint certification tables (max violation, slack, active fraction, pseudo-multiplier) for both domains

Artifacts:
- `outputs/validation/aircraft_monte_carlo.csv`
- `outputs/validation/spacecraft_monte_carlo.csv`
- `outputs/validation/baseline_comparison.csv`
- `outputs/validation/stress_tests.csv`
- `outputs/validation/stress_downlink_windows.csv`
- `outputs/validation/aircraft_pareto_trade_study.csv`
- `outputs/validation/spacecraft_pareto_trade_study.csv`
- `outputs/validation/deterministic_replay.json`
- `outputs/validation/spacecraft_monte_carlo_traces.csv`
- `outputs/validation/stress_spacecraft_low_solar_trace.csv`

- `outputs/validation/robustness_max_deviation.csv`
- `outputs/validation/validation_summary.json`
- `outputs/validation/aircraft_hybrid_vs_beam_variance.png`
- `outputs/validation/aircraft_scenario_sensitivity.png`
- `outputs/validation/spacecraft_hybrid_vs_beam_science.png`
- `outputs/validation/spacecraft_scenario_science.png`
- `outputs/validation/spacecraft_stress_resource_evolution.png`
- `outputs/validation/stress_margin_plot.png`
- `outputs/validation/stress_downlink_vs_constraints.png`
- `outputs/validation/aircraft_pareto_frontier.png`
- `outputs/validation/spacecraft_pareto_frontier.png`
- `outputs/aircraft/independent_checks.json`
- `outputs/spacecraft/independent_checks.json`

6. RESULTS (DEFAULT CONFIG)
From `outputs/*` on default reproducible run:
- Aircraft: solved, hard violations = 0, mission time = 6993.66 s (116.56 min), energy remaining = 588.05 Wh.
- Spacecraft: solved, hard violations = 0, delivered science = 229.81, battery minimum = 65.03 Wh, final data buffer = 0.0 MB.
- Validation mode: `fast` profile Monte Carlo and stress runs execute with 100% solve-rate for both domains.
- Deterministic replay check passes for both domains (hash-equal repeated solves).
- Pareto trade-study frontiers are generated for aircraft (time-energy) and spacecraft (science-battery).

CONSTRAINT CERTIFICATION MARGINS
The independent constraint checker evaluates Slack (limit - actual), creating pseudo-multipliers identifying the tightest system boundaries:
*   **UAV Turn-Rate:** `Active Fraction: 1.00`, `Observed: 0.069813 rad/s` at the configured limit (binding maneuver constraint).
*   **UAV Geofence Margin:** `Min Slack: 5.0 m` with 450 m safety margin.
*   **Spacecraft Battery:** `Min Slack: 0.026 Wh`, active fraction `0.17` (battery constraint is active near lower bound).
*   **Spacecraft Downlink Elevation:** `Min Slack: 0.015 deg` above 10 deg minimum.
*   **Spacecraft Off-Nadir:** `Min Slack: 0.148 deg` below 55 deg maximum.

7. LIMITATIONS AND NEXT STEPS
- Orbit model uses two-body + first-order J2 secular drift and a low-order solar ephemeris; it still excludes drag and higher-order perturbations.
- Aircraft model is point-mass kinematics with wind harmonics and energy proxy (`P ~ v^3`), without full aero force decomposition.
- Ground contact network uses two high-latitude stations; adding equatorial stations would improve downlink diversity and resilience.
- Next steps:
  1. Add full SGP4/TLE drag integration for highly-accurate opportunity window predictions.
  2. Implement comprehensive BADA-style lift/drag components.
  3. Introduce dynamic weather-front boundary integration into routing algorithms.

REPRODUCIBILITY

```bash
python -m pip install -r requirements.txt
python run_all.py --config configs/default.yaml
```

All outputs regenerate into `outputs/` and are packaged into `outputs/results_bundle.zip`.