




# HyPlan: An Open-Source Python Library for Planning Airborne Remote Sensing Campaigns

Ryan Pavlick <sup>1</sup> ¶

<sup>1</sup> National Aeronautics and Space Administration, Washington, DC, USA ¶ Corresponding author

DOI: [10.xxxxxx/draft](https://doi.org/10.xxxxxx/draft)

## Software

- [Review](#) 
- [Repository](#) 
- [Archive](#) 

Editor: [Open Journals](#) 

## Reviewers:

- [@openjournals](#)

Submitted: 01 January 1970

Published: unpublished

## License

Authors of papers retain copyright and release the work under a Creative Commons Attribution 4.0 International License ([CC BY 4.0](#))

## Summary

Airborne remote sensing—using instrumented aircraft to collect data over the Earth’s surface—is a critical tool for Earth science, enabling observations at spatial and temporal scales that bridge ground-based measurements and satellite data. Airborne platforms carry imaging spectrometers, lidars, radars, and other instruments over study areas at altitudes ranging from a few hundred meters to over 20 kilometers, producing data products across Earth science disciplines. Planning these campaigns requires simultaneously reasoning about aircraft flight dynamics, sensor characteristics and ground sampling, environmental conditions, airport logistics, airspace restrictions, satellite overpass coordination, and more. Scientists typically address these constraints using ad hoc spreadsheets, manual calculations, and institutional knowledge, producing workflows that are error-prone, hard to reproduce, and hard to transfer between campaigns.

HyPlan is an open-source Python library that provides a unified, reproducible, and extensible framework for planning airborne remote sensing missions. It encodes the physics of sensor–platform–environment interactions into composable building blocks covering the full mission-planning lifecycle, from study-area definition and flight line generation through swath and GSD calculations to multi-day mission plans with line-ordering optimization. It includes pre-configured models for a range of NASA airborne instruments (e.g., AVIRIS-3, PRISM, LVIS, UAVSAR, HyTES) and multiple research aircraft, along with tools for solar geometry, terrain-aware swath modeling, cloud climatology, airport selection, and satellite overpass prediction.

HyPlan’s core technical contribution is **terrain-aware swath modeling using ray–terrain intersection**, which captures terrain-induced variations in swath width and position that flat-Earth approximations miss; the effect can be substantial in mountainous regions, where swath width may vary by hundreds of meters along a single flight line. A complementary contribution is **wind-aware trajectory modeling subject to aircraft performance constraints**, combining Dubins paths (trochoidal under non-zero wind) with altitude-dependent climb, cruise, and descent models to produce physically realistic flight paths and segment-by-segment timing. The same wind and aircraft-performance machinery powers **wind-aware reachability isochrones**, which compute the boundary of operationally feasible study areas from a given base airport given a time budget, optional refuel candidates, and a wind field—turning “is this site reachable?” into a one-call query that drops directly into the rest of the planning workflow.

## Statement of Need

Airborne science campaigns are major investments—a single ER-2 deployment costs tens of thousands of dollars per flight hour, and campaigns typically span weeks to months—yet the planning process has remained largely manual and fragmented across disconnected tools.

42 Planning these campaigns is a coupled problem spanning sensor performance, solar geometry,  
43 environmental conditions, logistics, and satellite coordination. Sensor requirements such as  
44 ground sample distance and swath overlap depend on altitude and terrain; solar elevation  
45 and glint constrain when observations are viable; clouds and ecosystem state determine data  
46 usability; and aircraft endurance and airport availability constrain how flight lines can be  
47 scheduled.

48 These constraints are interdependent: altitude affects both sensor performance and coverage,  
49 wind alters trajectory and timing, and environmental conditions determine when planned flight  
50 lines can be executed. In practice, existing workflows treat these elements separately, making  
51 it difficult to evaluate trade-offs or ensure reproducibility.

52 HyPlan addresses this need by treating airborne mission planning as a **coupled physical and**  
53 **spatiotemporal problem**, unifying aircraft dynamics, sensor modeling, environmental constraints,  
54 and mission-level scheduling within a single reproducible system.

## 55 State of the Field

56 Several categories of tools address subsets of the airborne mission planning problem, but none  
57 in the integrated, programmatic manner that science campaign design requires.

58 **Commercial flight planning software** (ForeFlight, Jeppesen FliteStar) targets pilot navigation,  
59 fuel planning, and regulatory compliance, not scientific objectives such as GSD, spectral  
60 coverage, or sensor-specific swath overlap. **GIS platforms** (QGIS, ArcGIS) can visualize  
61 flight lines and study areas but lack domain-specific calculations for sensor modeling, aircraft  
62 performance, and mission timing.

63 **UAS mission planners and agency-internal tools.** Tools such as Mission Planner and  
64 QGroundControl ([Dronecode Project, 2024](#)) target small drones at low altitudes and short  
65 ranges, and do not model the sensor physics, solar geometry, or logistics constraints relevant  
66 to crewed research aircraft. Various NASA centers maintain internal planning tools tailored to  
67 specific instruments or aircraft, but these are typically proprietary, undocumented, and tightly  
68 coupled to particular missions.

69 **Moving Lines.** The most directly comparable tool is Moving Lines ([LeBlanc, 2018](#)), a Python-  
70 based application developed for NASA airborne science campaigns. Moving Lines excels at  
71 real-time, interactive flight plan creation and modification during campaign operations, with a  
72 graphical user interface combining an interactive map, spreadsheet-based waypoint editing, and  
73 overlays of satellite imagery and weather model output. It has been used operationally across  
74 numerous NASA campaigns including ORACLES, IMPACTS, and ARCSIX. Moving Lines is  
75 designed as a GUI for manual, interactive planning rather than as a programmatic library, and  
76 does not provide sensor-specific swath and GSD calculations, terrain-aware analysis, automated  
77 flight line generation, or ordering optimization. HyPlan and Moving Lines are complementary:  
78 HyPlan addresses the pre-campaign science planning phase—determining how many flight  
79 lines are needed, what altitude and speed satisfy sensor requirements, when solar conditions  
80 permit data collection, and how to schedule lines across multiple flight days—while Moving  
81 Lines supports the tactical, day-of-flight planning and replanning that occurs during campaign  
82 execution.

## 83 Software Design

84 HyPlan is designed as a **programmatic, physics-based flight planning library** for airborne remote  
85 sensing campaigns. The architecture reflects a set of deliberate design choices aimed at  
86 supporting reproducible, science-driven mission planning while accommodating the complexity  
87 of aircraft performance, sensor geometry, and environmental constraints.

## 88 Programmatic workflows over interactive tools

89 A central design decision is to expose all functionality through a **Python API** rather than  
90 a graphical user interface. While interactive tools such as Moving Lines are well-suited for  
91 real-time, tactical flight planning, they make it difficult to reproduce, audit, or systematically  
92 explore planning decisions.

93 HyPlan instead represents mission design as a sequence of function calls operating on explicit  
94 data structures such as FlightLine, FlightBox, and flight plan GeoDataFrames built on  
95 NumPy (Harris et al., 2020), pandas (McKinney, 2010), GeoPandas (Jordahl et al., 2020),  
96 and Shapely (Gillies & others, 2007). Modules such as flight\_plan and flight\_optimizer  
97 combine these abstractions to produce complete multi-segment sortie descriptions, including  
98 takeoff, climb, transit, data collection, and landing.

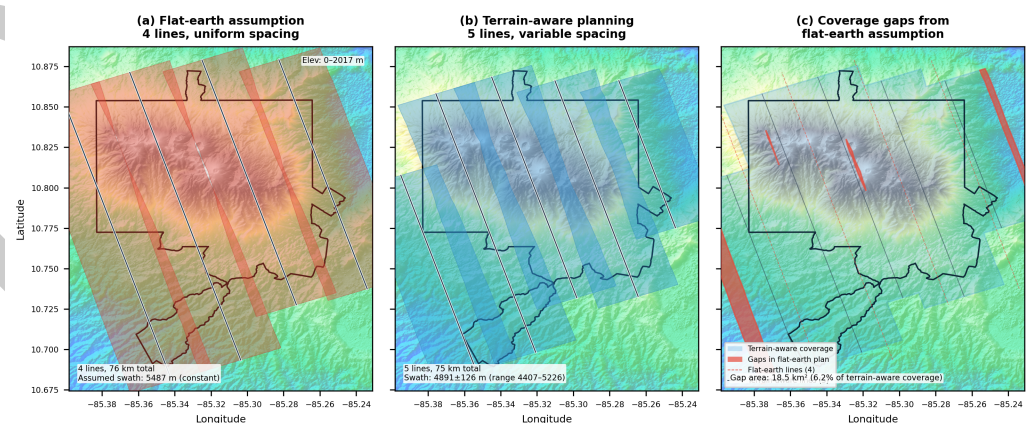
99 The trade-off is reduced interactivity in exchange for **reproducibility and automation**, enabling  
100 version-controlled workflows and systematic exploration of planning parameters.

## 101 Physics-based modeling over geometric approximation

102 HyPlan explicitly models the physical relationships between aircraft motion, sensor geometry,  
103 and the environment. This is reflected across several modules:

- 104 ■ The swath module performs **terrain-aware swath modeling** by tracing rays from the  
105 sensor to the terrain surface using digital elevation models accessed via rasterio (Gillies &  
106 contributors, 2024) and GDAL (GDAL/OGR contributors, 2024), rather than assuming  
107 flat-Earth geometry (Figure 1).
- 108 ■ The flight\_plan module incorporates **wind-aware trajectory modeling**, computing crab  
109 angles and ground speeds from airspeed and wind vectors (Figure 2).
- 110 ■ The flight planner models **aircraft-constrained motion** by combining a 2D Dubins solver  
111 (Dubins, 1957) for horizontal geometry with vertical profiles integrated against horizontal  
112 distance from each aircraft's calibrated climb and descent rates, so that top-of-climb  
113 and top-of-descent land at physically realistic positions for non-trivial vertical profiles.  
114 Under non-zero wind, the air-relative Dubins arcs become **trochoidal ground trajectories**  
115 (Sachdev et al., 2023) (Figure 2).

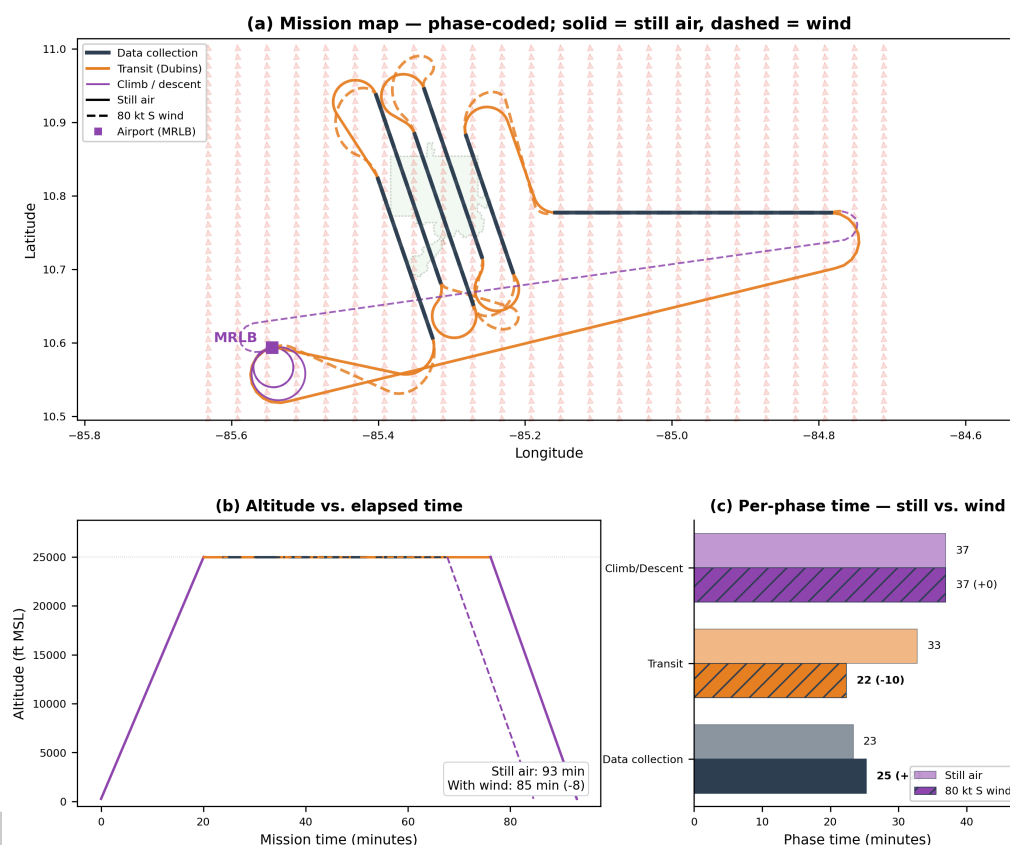
AVIRIS-3 @ 25000 foot, 20% overlap — Rincón de la Vieja, Costa Rica



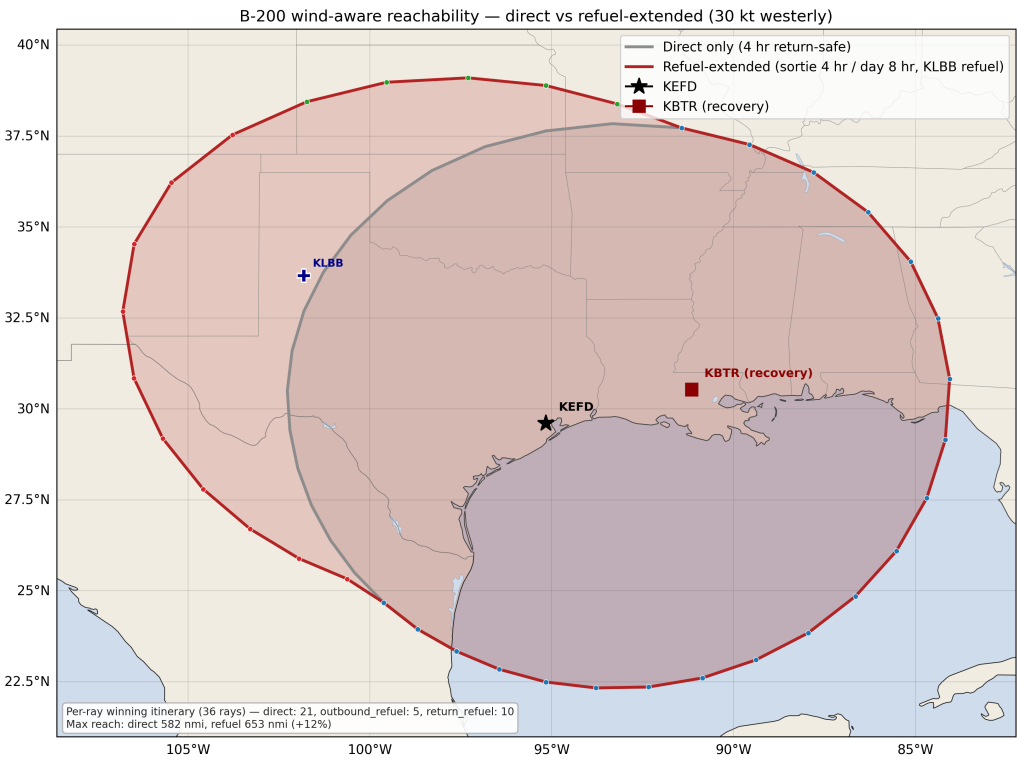
**Figure 1:** Flat-earth vs terrain-aware flight planning over Rincón de la Vieja National Park, Costa Rica (elevation 234–2,072 m). (a) A flat-earth planner assumes constant swath width and produces uniformly spaced lines. (b) Terrain-aware planning uses ray–terrain intersection to measure actual swath width at each line position, requiring additional lines where terrain narrows the swath. (c) Coverage gaps (red) show areas that would be missed by the flat-earth plan but are covered by the terrain-aware plan. Quantitative callouts on each panel report line count, total line length, swath statistics, and the resulting gap area.

116 The trade-off is increased complexity compared to planners that assume straight-line motion  
117 or constant ground speed. The benefit is **physically realistic trajectory, timing, and coverage**  
118 **estimates**—particularly when wind speeds are non-negligible relative to aircraft speed or altitude  
119 changes are required—ensuring that planned trajectories are both kinematically feasible and  
120 scientifically valid.

**King Air B200 — MRLB → Rincón de la Vieja → MRLB, AVIRIS-3 @ FL250**



**Figure 2:** Wind-aware mission planning for the Rincón de la Vieja survey using a King Air B-200, with a single 30 km east-west relocation flight line offset 10 nautical miles east of the survey box. (a) Mission map; solid lines are still-air Dubins arcs and dashed lines are trochoidal arcs under an 80 kt south wind, both color-coded by phase. (b) Altitude vs. elapsed mission time, both conditions overlaid. (c) Per-phase time totals (still vs. wind) for climb/descent, transit, and data collection, with absolute deltas annotated. The N–S survey box’s alternating-direction lines mostly cancel wind effects, but the relocation strip and the asymmetric out-and-back transit to it do not: with the wind aligned along the dominant transit direction, the planner reroutes the relocation strip to its wind-favorable traversal and the mission completes 8.5 min faster than the 93.0 min still-air baseline (~9.1%), with savings split between the transit and data-collection phases. The same per-phase decomposition propagates into endurance budgeting, scheduling constraints, and the isochrone solver shown in Figure 3.



**Figure 3:** Wind-aware reachability isochrone for a King Air B-200 based at KEFD (Ellington Field, Houston) recovering at KBTR (Baton Rouge), with KLBB (Lubbock) as a refuel candidate, under a synthetic 30 kt westerly wind at FL250. The gray polygon is the *direct-only* boundary at a single 4-hour fuel cycle; the firebrick polygon is the refuel-extended reach at sortie 4 hr / day 8 hr / 30 min refuel. Per-ray dots are colored by the winning itinerary — blue for direct, green for outbound\_refuel, red for return\_refuel. Refuel-extended reach contains the direct boundary by construction (refuel is optional, so each ray takes the maximum-distance template); the extension shows where the optional fuel stop pays off.

**Composable abstractions rather than monolithic workflows**

HyPlan decomposes the planning problem into **composable abstractions** that can be combined into flexible workflows. Core abstractions include flight lines, survey patterns, sensor models, swath generators, glint analyses, and sortie/mission planners.

Each component is designed to operate independently while maintaining compatibility with others. For example, *FlightLine* objects can be generated independently of sensor choice, sensor models can be applied to arbitrary flight geometries, and swath or glint analyses can be layered onto the same flight lines without modifying the underlying planning objects.

The trade-off is that users must assemble workflows themselves rather than relying on a single “one-click” planner. However, this modular design allows HyPlan to support a wide range of research scenarios, from simple coverage estimation to complex multi-day campaign optimization. It also enables extensibility: new aircraft, sensors, flight patterns, or analysis methods can be introduced without modifying core modules.

**Explicit units and geodesic accuracy**

All physical quantities in HyPlan carry explicit units using the Pint library (Grecco, 2024), and all spatial calculations are performed on the WGS84 ellipsoid using geodesic methods implemented via pymap3d (Hirsch, 2018) and pyproj (Snow & others, 2023).



This design avoids the class of unit conversion and projection errors that commonly arise in mixed-domain workflows—the kind of error famously responsible for the loss of Mars Climate Orbiter (NASA, 1999). The trade-off is additional implementation complexity compared to unitless or planar approaches. The benefit is **numerical correctness and consistency across domains**, allowing seamless integration of aviation conventions (feet, knots, nautical miles) with scientific conventions (meters, m/s, kilometers).

For airborne campaigns spanning hundreds to thousands of kilometers, these geodesic considerations are essential to maintaining spatial accuracy.

#### Integration of flight execution and environmental timing

A distinguishing design feature of HyPlan is the integration of **flight execution modeling** (“how to fly”) with **environmental timing constraints** (“when to fly”) within a single framework.

In addition to geometric and aircraft modeling, HyPlan includes modules for:

- **Solar geometry and glint analysis** (sun, glint): computation of solar elevation (Reda & Andreas, 2004) and specular reflection geometry, including glint arcs that constrain flight heading and timing
- **Cloud climatology** (clouds): estimation of clear-sky probability from reanalysis data via Open-Meteo (Zippenfenig, 2024) or Google Earth Engine (Gorelick et al., 2017), with campaign simulation tools
- **Vegetation phenology** (phenology): extraction of NDVI/LAI time series and phenological transition dates from MODIS products
- **Satellite coordination** (satellites): prediction of overpass timing and swath overlap using TLE propagation via Skyfield (Rhodes, 2019) and CelesTrak (CelesTrak, 2024)

These modules enable users to determine not only where and how to fly, but also **when environmental conditions are suitable for data collection**. The trade-off is increased system scope compared to planners focused solely on geometry or navigation. However, airborne campaigns are inherently constrained by environmental conditions, and optimal data acquisition depends on jointly satisfying spatial and temporal constraints.

#### Logistics-aware mission planning

HyPlan incorporates logistics considerations directly into the planning process. The `flight_plan` module models complete sorties, while `flight_optimizer` addresses the problem of ordering flight lines across multiple days subject to constraints such as aircraft endurance, maximum daily flight time, and the availability of refueling airports. The complementary `planning.isochrone` module answers the upstream “where *can* we observe?” question by computing wind-aware reachability boundaries around a base airport, with optional refuel-extended reach (two-clock per-cycle / per-day budget tracking) and single-target spot checks (Figure 3). Together these turn site selection, multi-day scheduling, and refuel-aware feasibility into a coherent programmatic workflow.

Airport selection is supported through integration with the OurAirports database (OurAirports, 2024), enabling filtering by runway length, surface type, and proximity. Airspace conflict detection uses FAA NASR and OpenAIP (OpenAIP, 2024) data to identify intersections with restricted, prohibited, or controlled airspace. These logistics constraints interact with solar and environmental timing, creating a coupled optimization problem that HyPlan addresses through heuristic graph-based methods using NetworkX (Hagberg et al., 2008).

The trade-off is that the optimizer uses heuristic approaches rather than guaranteeing global optimality. The benefit is **computational efficiency and practical applicability**, producing usable

186 multi-day schedules for large campaigns without requiring expensive optimization algorithms.

## 187 Design implications for research applications

188 These design choices collectively enable HyPlan to function as a **reproducible research tool**  
189 rather than a static planning utility. Researchers can encode mission design assumptions in  
190 code, evaluate sensitivity to parameters such as altitude, overlap, or cloud thresholds, and  
191 generate planning products that are directly traceable to scientific requirements.

192 This architecture supports a range of applications, including:

- 193     ▪ design of airborne calibration/validation campaigns for satellite missions
- 194
- 195     ▪ planning of ecological and biogeochemical surveys with phenology constraints
- 196
- 197     ▪ optimization of multi-day campaigns under cloud, solar, and logistics constraints
- 198
- 199     ▪ integration of airborne observations with satellite overpasses

## 200 Limitations

201 HyPlan is designed for pre-campaign planning and does not currently model real-time operational  
202 constraints such as dynamic weather avoidance, air traffic control restrictions, or in-flight  
203 replanning. These capabilities are typically addressed by operational tools such as Moving  
204 Lines during campaign execution.

205 Aircraft performance models combine published specifications, operator-provided values, and  
206 calibration from public or NASA in-situ flight-state data where available. HyPlan currently  
207 includes data-fit climb, cruise, descent, turn, and/or approach parameters for several research  
208 platforms, including the NASA ER-2, G-III, G-V, WB-57, C-130, P-3, King Air B-200, and  
209 Twin Otter. These calibrations use IWG1 or ICARTT navigation/meteorological products  
210 and are documented in per-aircraft calibration notebooks. Calibrated quantities should be  
211 interpreted as representative operational behavior for the sampled mission mix, not as certified  
212 aircraft-envelope limits; remaining aircraft use brochure-derived or sibling-airframe-inferred  
213 values until comparable flight-state data are available.

## 214 Research Impact Statement

215 Early versions of HyPlan have been applied in exploratory and pre-campaign planning workflows  
216 for multiple NASA airborne science campaigns, including the SHIFT campaign ([Chadwick et al., 2025](#)),  
217 AVIRIS deployments for the NASA Arctic Boreal Vulnerability Experiment (ABOVE)  
218 in 2022 and 2023 ([Miller et al., 2025](#)), the BioSCaPE campaign ([Cardoso et al., 2025](#)), and  
219 the AVUELO AVIRIS campaign (2025).

220 HyPlan is also currently being used to support planning for the NASA Student Airborne  
221 Research Program (SARP) 2026 campaign and the joint NASA–Australia AVIRIS campaign,  
222 and to refine the concise experiment plan for a notional future NASA PANGAEA Central Africa  
223 campaign.

224 In these applications, HyPlan supported evaluation of flight line configurations, terrain and  
225 solar constraints, and mission feasibility under logistical and environmental limitations, directly  
226 informing the design of core capabilities such as terrain-aware swath modeling, flight pattern  
227 generation, and mission-level scheduling.

228 The software demonstrates strong community-readiness signals, including over 1,500  
229 automated tests with greater than 80% code coverage, continuous integration on every  
230 commit, comprehensive API documentation, and more than 20 Jupyter notebooks that serve

as both tutorials and integration tests. Core calculations are validated against independent references including Vincenty geodesic (Vincenty, 1975) test cases, NOAA solar geometry calculations (Reda & Andreas, 2004), and analytical sensor models.

HyPlan is under active development for integration into future NASA airborne campaign planning workflows, where its ability to unify sensor modeling, aircraft performance, and environmental constraints in a single framework is expected to support more reproducible and efficient mission design.

## AI Usage Disclosure

Generative AI tools (Claude and ChatGPT) were used to assist with drafting and editing this manuscript. The software itself was developed with AI coding assistance (Claude and ChatGPT). All AI-generated content was reviewed and verified by the author.

## Acknowledgements

The author thanks Samuel LeBlanc for developing Moving Lines and for contributions to the airborne science planning community that informed HyPlan's design.

This work was supported by the National Aeronautics and Space Administration.

## References

- Cardoso, A. W., Hestir, E. L., Slingsby, J. A., & others. (2025). The biodiversity survey of the Cape (BioSCape), integrating remote sensing with biodiversity science. *Npj Biodiversity*, 4(1), 5. <https://doi.org/10.1038/s44185-024-00071-5>
- CelesTrak. (2024). *CelesTrak: Current GP element sets*. <https://celestrak.org/>
- Chadwick, K. D., Cawse-Nicholson, K., Thompson, D. R., & others. (2025). Unlocking ecological insights from sub-seasonal visible-to-shortwave infrared imaging spectroscopy: The SHIFT campaign. *Ecosphere*, 16(3), e70194. <https://doi.org/10.1002/ecs2.70194>
- Dronecode Project. (2024). *QGroundControl: Intuitive and powerful ground control station for the MAVLink protocol*. <https://doi.org/10.5281/zenodo.595404>
- Dubins, L. E. (1957). On curves of minimal length with a constraint on average curvature, and with prescribed initial and terminal positions and tangents. *American Journal of Mathematics*, 79(3), 497–516. <https://doi.org/10.2307/2372560>
- GDAL/OGR contributors. (2024). *GDAL/OGR geospatial data abstraction software library*. Open Source Geospatial Foundation. <https://doi.org/10.5281/zenodo.5884351>
- Gillies, S., & contributors. (2024). *Rasterio: Geospatial raster i/o for python*. <https://github.com/rasterio/rasterio>
- Gillies, S., & others. (2007). *Shapely: Manipulation and analysis of geometric objects*. Zenodo. <https://doi.org/10.5281/zenodo.5597138>
- Gorelick, N., Hancher, M., Dixon, M., Ilyushchenko, S., Thau, D., & Moore, R. (2017). Google Earth Engine: Planetary-scale geospatial analysis for everyone. *Remote Sensing of Environment*, 202, 18–27. <https://doi.org/10.1016/j.rse.2017.06.031>
- Grecco, H. E. (2024). *Pint: Operate and manipulate physical quantities in Python*. <https://pint.readthedocs.io>
- Hagberg, A. A., Schult, D. A., & Swart, P. J. (2008). Exploring network structure, dynamics, and function using NetworkX. *Proceedings of the 7th Python in Science Conference*, 11–15.



- 272 <https://doi.org/10.25080/TCWV9851>
- 273 Harris, C. R., Millman, K. J., Walt, S. J. van der, Gommers, R., Virtanen, P., Cournapeau, D.,  
274 Wieser, E., Taylor, J., Berg, S., Smith, N. J., Kern, R., Picus, M., Hoyer, S., Kerkwijk,  
275 M. H. van, Brett, M., Haldane, A., Río, J. F. del, Wiebe, M., Peterson, P., ... Oliphant,  
276 T. E. (2020). Array programming with NumPy. *Nature*, 585(7825), 357–362. <https://doi.org/10.1038/s41586-020-2649-2>
- 277
- 278 Hirsch, M. (2018). PyMap3D: 3-D coordinate conversions for terrestrial and geospace  
279 environments. *Journal of Open Source Software*, 3(23), 580. [https://doi.org/10.21105/](https://doi.org/10.21105/joss.00580)  
280 [joss.00580](https://doi.org/10.21105/joss.00580)
- 281 Jordahl, K., Van den Bossche, J., Fleischmann, M., & others. (2020). *GeoPandas: Python*  
282 *tools for geographic data*. Zenodo. <https://doi.org/10.5281/zenodo.3946761>
- 283 LeBlanc, S. E. (2018). *Moving lines: NASA airborne research flight planning tool*. Zenodo.  
284 <https://doi.org/10.5281/zenodo.1478126>
- 285 McKinney, W. (2010). Data structures for statistical computing in Python. *Proceedings of the*  
286 *9th Python in Science Conference*, 56–61. <https://doi.org/10.25080/Majora-92bf1922-00a>
- 287 Miller, C. E., Green, R. O., Thompson, D. R., Thorpe, A. K., Eastwood, M. L., McCubbin,  
288 I. B., Olson-Duvall, W., Bernas, M. A., Sarture, C. M., Rios, L. M., Hernandez, M. A.,  
289 Bue, B. D., Lundeen, S. R., Pavlick, R., Chapman, J. W., Brodrick, P. G., Eckert, R. F.,  
290 Coleman, R. W., Baskaran, L., & Elder, C. D. (2025). Airborne imaging spectroscopy  
291 surveys of Arctic and boreal Alaska and northwestern Canada 2017–2023. *Scientific Data*,  
292 12, 692. <https://doi.org/10.1038/s41597-025-04898-w>
- 293 NASA. (1999). *Mars Climate Orbiter mishap investigation board phase I report*. [https://llis.nasa.gov/llis\\_lib/pdf/1009464main1\\_0641-mr.pdf](https://llis.nasa.gov/llis_lib/pdf/1009464main1_0641-mr.pdf)
- 294
- 295 OpenAIP. (2024). *OpenAIP: Open aviation data*. <https://www.openaip.net/>
- 296 OurAirports. (2024). *OurAirports: Open data for airports*. <https://ourairports.com/data/>
- 297 Reda, I., & Andreas, A. (2004). Solar position algorithm for solar radiation applications. *Solar*  
298 *Energy*, 76(5), 577–589. <https://doi.org/10.1016/j.solener.2003.12.003>
- 299 Rhodes, B. (2019). *Skyfield: High precision research-grade positions for planets and Earth*  
300 *satellites generator*. <https://rhodesmill.org/skyfield/>
- 301 Sachdev, S., Moon, B., Yuan, J., & Scherer, S. (2023). Time-optimal path planning in a  
302 constant wind for uncrewed aerial vehicles using Dubins set classification. *arXiv Preprint*  
303 *arXiv:2306.11845*. <https://doi.org/10.48550/arXiv.2306.11845>
- 304 Snow, A., & others. (2023). *Pyproj: Python interface to PROJ*. Zenodo. <https://doi.org/10.5281/zenodo.2592232>
- 305
- 306 Vincenty, T. (1975). Direct and inverse solutions of geodesics on the ellipsoid with application  
307 of nested equations. *Survey Review*, 23(176), 88–93. [https://doi.org/10.1179/sre.1975.23.](https://doi.org/10.1179/sre.1975.23.176.88)  
308 [176.88](https://doi.org/10.1179/sre.1975.23.176.88)
- 309 Zippenfenig, P. (2024). *Open-meteo.com weather API*. Zenodo. [https://doi.org/10.5281/](https://doi.org/10.5281/zenodo.7970649)  
310 [zenodo.7970649](https://doi.org/10.5281/zenodo.7970649)