Development of the HERMES Ground Station GUI for Antenna Pointing by NEBP Engineering Teams

Jesse Cook[a]

Abstract

During the Nationwide Eclipse Ballooning Project (NEBP), "engineering" teams were charged with live video streaming – a capability that requires reliable line-of-sight radio communication between a stratospheric payload and a ground station potentially a dozen or more miles below. HERMES is a purpose-built graphical user interface (GUI) designed to track a stratospheric balloon flight and display a livestream from onboard camera(s) in real time. HERMES can receive GPS location data from three independent radio channels simultaneously and, using a weighted filter, calculate pointing angles for the ground station's antenna. GPS information from two such tracking systems, Iridium and APRS, are picked up by HERMES through the Internet, while the RFD900 payload transmits directly to a tethered receiver. The HERMES system was designed with redundancy in mind; should one tracking system fail (or otherwise become unreliable), GPS information from another can be used in its place. In addition, manual controls allow the user to adjust the antenna pointing angles, correct small errors, and point at payloads without radio input. HERMES includes a suite of visual elements, itemized data-logging, numerous housekeeping features, and can automatically control an NEBP vent to "float" payloads in the stratosphere. HERMES was launched in open beta in 2023 and officially released shortly before the total solar eclipse in April 2024. HERMES remains in development, with new updates in the pipeline to allow users to more-effectively track stratospheric ballooning flights and interact with payloads via line-of-sight telemetry.

NEBP | Video Streaming | Ground Station | GUI

1. Introduction

The concept of real-time communication with a ballooning payload is nothing novel; in fact, the ground station standardized for the 2024 nationwide project was identical to the hardware used in 2017, the previous year the United States experienced a total solar eclipse. While other onboard cameras captured high-quality footage (such as GoPro cameras and Insta360 cameras), live video streaming offered teams and general observers the opportunity to watch the eclipse from the balloon's vantage point as it happened without worry of cloud cover or poor visibility.

During the 2023 and 2024 eclipses, NEBP ballooning teams flew live-streaming video payloads, continuously downlinking footage from a Raspberry Pi computer during ascent and float. This required each ground station, whether set up at the launch site or somewhere downrange, to accurately track the payload across the sky. Assuming reliable line-of-sight communication, the ground station could then receive data from a Ubiquiti modem and stream the real-time video footage either locally or on YouTube for others to view.

[a]Undergraduate student, University of Minnesota - Twin Cities, cook0690@umn.edu

Named for the Greek god of messengers, HERMES was built as an “all-in-one” program to automate the pointing of the ground station’s tracking dish while performing other essential flight tasks. Whereas before, with preexisting ground station software, users would have to access their livestream from a separate window, HERMES can display the video feed on a central screen amidst data readouts and other helpful visuals. In addition, teams flying an Iridium-commanded vent can send instructions directly from HERMES to bring their balloons to a controlled float in the stratosphere, and later to terminate the flight.

This project has been in development since early 2023, with the first official version having been released on June 28 of that year. Since then, HERMES has seen twelve major updates leading to the culmination of the Nationwide Eclipse Ballooning Project during the total solar eclipse on April 8, 2024.

2. Development

HERMES was designed first and foremost as a means of automating the ground station; every other feature was secondary to this goal. This essentially reduced the first version of the code to a dynamics problem: How can basic GPS information transmitted from the balloon – latitude, longitude, and altitude – provide the ground station with enough information to point its antenna (mounted to the tracking dish) directly at the payload, while accounting for error-inducing factors like the curvature of the earth?

The ground station itself is defined relative to an Earth-fixed inertial frame; as far as the tracking dish is concerned, it sits on a flat plane with the balloon located, and usually in motion, somewhere overhead (see Fig. 1).

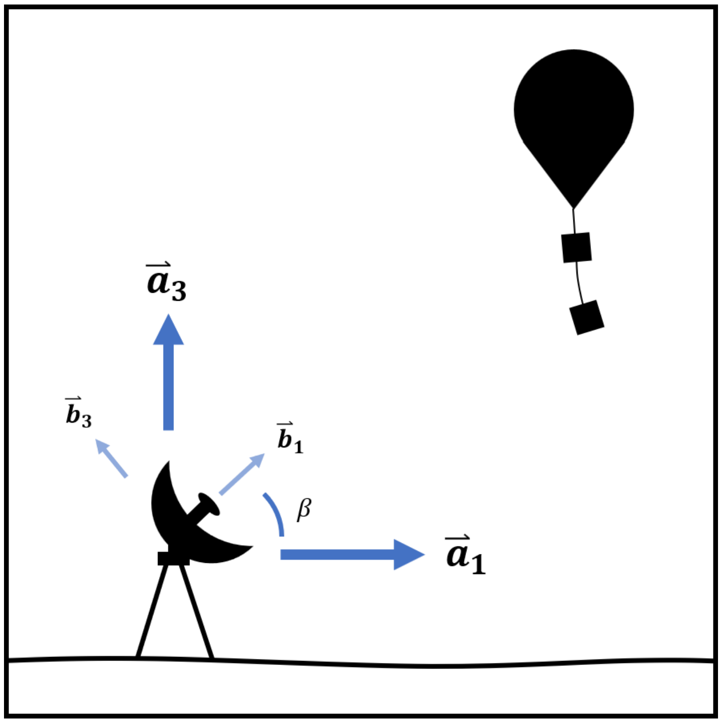


Fig. 1. Two-dimensional view of ground reference frames relative to a balloon.

From the payload’s point of view, however, the spatial relationship is more complex. As the balloon ascends higher into the atmosphere, the Earth’s curvature becomes more apparent, and the assumption of a relatively-flat body breaks down (see Fig. 2). At great distances, it is not practical to roughly calculate distances between two coordinate pairs as one might do on a small portion of a Cartesian map.

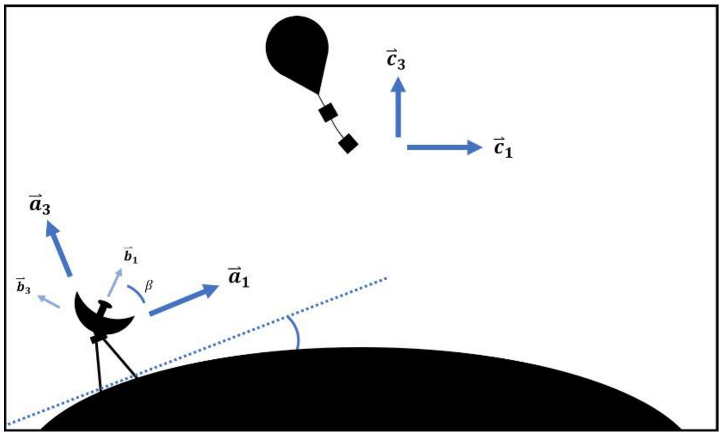


Fig. 2. Two-dimensional view of the ground station on a curved Earth surface.

These illustrations depict only two-dimensional space for visual simplicity, but the calculations must be treated as a 3D problem. This means that conversions between reference frames must account for rotations about multiple axes, yielding some combination of direction cosine matrices:

$$C_1(\theta) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta \\ 0 & \sin\theta & \cos\theta \end{bmatrix} \tag{1}$$

$$C_2(\theta) = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix} \tag{2}$$

$$C_3(\theta) = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3}$$

Referencing Figs. (1) and (2), the expression for the frame $\vec{\mathcal{F}}_b$ in terms of $\vec{\mathcal{F}}_a$ is:

$$\vec{\mathcal{F}}_b^T = C_2(-\beta)\vec{\mathcal{F}}_a^T = \begin{bmatrix} \cos(-\beta) & 0 & \sin(-\beta) \\ 0 & 1 & 0 \\ -\sin(-\beta) & 0 & \cos(-\beta) \end{bmatrix} \begin{bmatrix} \vec{a}_1 \\ \vec{a}_2 \\ \vec{a}_3 \end{bmatrix} \tag{4}$$

where β is the "tilt" angle of the tracking dish relative to the ground station. This single rotation is performed clockwise about the $\vec{a}_2$ axis (hence the negative angle), which is equivalent to the $\vec{b}_2$ axis (both extend into the page from the observer's point of view). The "pan" angle of the dish would instead consider $\vec{a}_3$ the axis of rotation as follows:

$$\vec{\mathcal{F}}_b^T = C_3(\alpha)\vec{\mathcal{F}}_a^T = \begin{bmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \vec{a}_1 \\ \vec{a}_2 \\ \vec{a}_3 \end{bmatrix} \tag{5}$$

In this expression, α is defined as the angle of pan rather than tilt. To calculate the oriented reference frame $\vec{\mathcal{F}}_b$ given both angles, the two rotation matrices would be multiplied.

$$\vec{\mathcal{F}}_b^T = C_3(\alpha)C_2(-\beta)\vec{\mathcal{F}}_a^T = \begin{bmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos(-\beta) & 0 & \sin(-\beta) \\ 0 & 1 & 0 \\ -\sin(-\beta) & 0 & \cos(-\beta) \end{bmatrix} \begin{bmatrix} \vec{a}_1 \\ \vec{a}_2 \\ \vec{a}_3 \end{bmatrix} \tag{6}$$

This method can likewise be applied to calculate the payload's frame of reference $\vec{\mathcal{F}}_c$ relative to the earth-fixed inertial frame $\vec{\mathcal{F}}_a$ and the rotated "dish frame" $\vec{\mathcal{F}}_b$:

$$\vec{\mathcal{F}}_c^T = C_{cb}\vec{\mathcal{F}}_b^T = C_{ca}\vec{\mathcal{F}}_a^T \tag{7}$$

where the above matrices are constructed using multiple axis rotations (see Fig. 3).

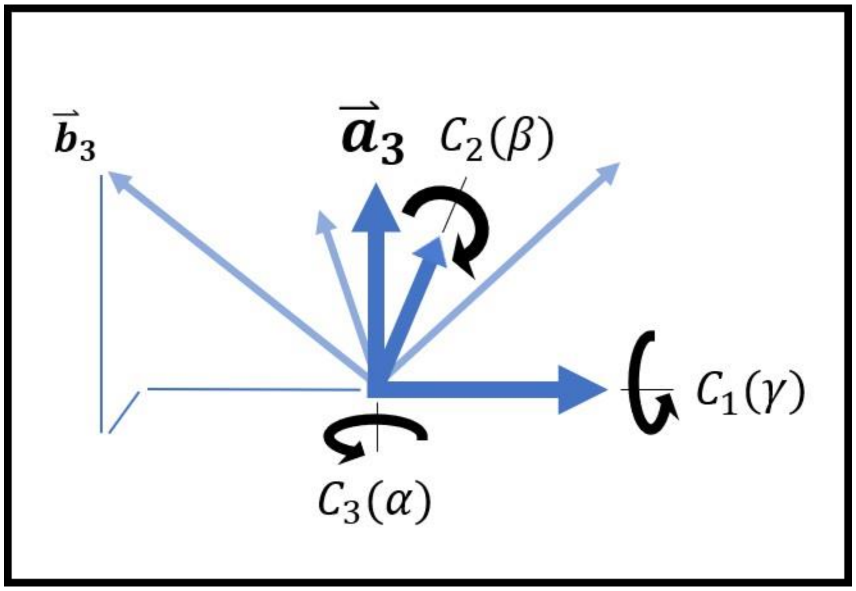


Fig. 3. Three-dimensional rotation of a basic reference frame.

This is the underlying theory of solving two-body distance problems given dynamic reference frames. Once the pair of coordinates are resolved in a common frame, the distance calculations become trivial; simply use the previously-defined frame $\vec{\mathcal{F}}_b$ and measure the length along the $\vec{b}_1$ axis, which by definition always points directly at the payload from the tracking dish. The trigonometry is already handled elegantly in matrix form by multiplying the rotation matrices when changing reference frames. From this, it is possible to solve for the two pointing angles, as illustrated in Fig. 4.

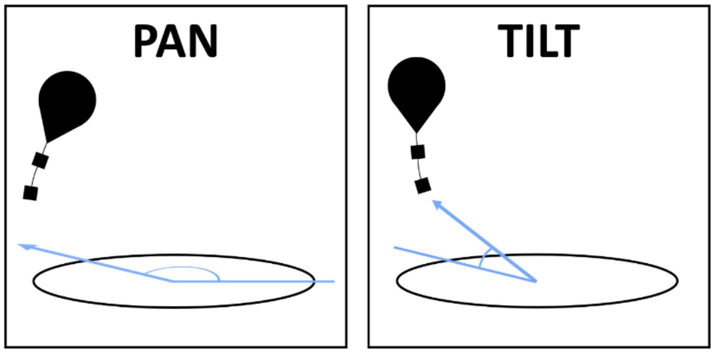


Fig. 4. A depiction of the pan and tilt angles relative to the ground. Tilt is the angle up from horizontal (i.e. from the horizon, if on flat terrain); pan is the heading angle measured from a fixed direction (like true north).

In early versions of HERMES, these matrix operations were executed in a loop each time new GPS data was received from the payload. This became quite overwhelming, especially as radios like the RFD900 were introduced, allowing far more frequent pings than was possible with just an Iridium modem tracking unit. The strain of these calculations quickly became evident, and a more-streamlined solution was sought. The code was later updated to rely on the Haversine formula, a powerful mathematical tool used to determine the "great circle" distance between two points on a sphere. Haversine calculations are far less computationally resource-intensive compared to direct matrix methods, especially when GPS information is received from the payload as often as once a second. As such, this approach is commonly used in large-scale GIS applications as well as navigation [1]. The name Haversine is a portmanteau of "half versine", referencing the trigonometric function equal to one minus the cosine of an angle. Relevant trigonometric functions, including "versine," are depicted in Fig. 5.

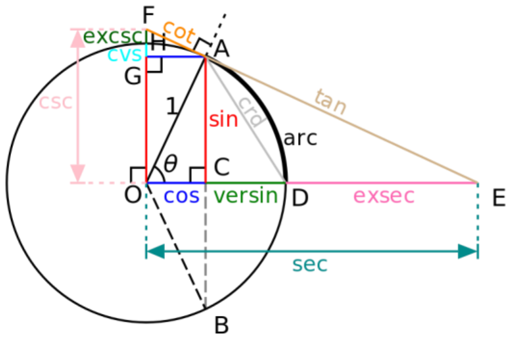


Fig. 5. A unit circle with various trigonometric functions labeled. [2]

The derivation of the formula itself is fairly complex, but it can be reduced to a few simple equations that HERMES can calculate efficiently in a loop. First, the two most recent GPS pings are received, and the difference in latitude, longitude, and altitude is recorded. Note that the coordinates must be converted to radians and the altitude to meters above sea level.

$$\begin{bmatrix} \Delta Lat \\ \Delta Lon \\ \Delta Alt \end{bmatrix} = \begin{bmatrix} Lat_2 - Lat_1 \\ Lon_2 - Lon_1 \\ Alt_2 - Alt_1 \end{bmatrix} \tag{8}$$

Next, as an intermediate step, the half-chord is determined between the two locations. At this stage, altitude is not factored into the computations, so the coordinate sets are considered to be at sea level – on the surface of the spherical Earth model.

$$a = \sin^2\left(\frac{\Delta Lat}{2}\right) + \cos(Lat_1)\cos(Lat_2)\sin^2\left(\frac{\Delta Lon}{2}\right) \tag{9}$$

The angular distance between the two points is then calculated using a two-argument arctangent function.

$$c = \operatorname{atan2}\left(\sqrt{a}, \sqrt{1-a}\right) \tag{10}$$

Using these values, the two-dimensional "great circle" distance can be obtained, and from this, the three-dimensional line-of-sight distance is found simply with a Pythagorean calculation. The conversion factor of 3958.8 yields measurements in miles.

$$d_{2D} = 3958.8c \tag{11}$$

$$d_{3D} = \sqrt{d_{2D}^2 + \Delta Alt^2} \tag{12}$$

Now that these distances are known, previously-defined variables can be reused to compute the two pointing angles for the ground station.

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} \cos(Lat_1)\sin(Lat_2) - \sin(Lat_1)\cos(Lat_2)\cos(\Delta Lon) \\ \sin(\Delta Lon)\cos(Lat_2) \end{bmatrix} \tag{13}$$

$$\alpha = \operatorname{atan2}(y, x) \tag{14}$$

Note that this angle α is <u>not</u> equivalent to the alpha defined earlier as a pan angle. That variable can be discarded, as α now represents the calculated azimuth. Finally, a modulus operator is applied to ensure that this azimuth falls within $[0,2\pi]$.

$$\alpha = \left(\frac{3\pi}{2} - \alpha\right) \ \% \ 2\pi \tag{15}$$

The desired pointing angles, in degrees, are as follows:

$$\begin{bmatrix} \theta_{pan} \\ \theta_{tilt} \end{bmatrix} = \begin{bmatrix} \left(\frac{3\pi}{2} - \alpha\right) \\ \operatorname{atan2}(\Delta Alt, 2R_{Earth}c) \end{bmatrix} \cdot \left(\frac{180°}{\pi}\right) \tag{16}$$

This method proved considerably simpler and more practical than multiplying rotation matrices, which taxed even the more powerful computers running early versions of HERMES. Switching distance calculations to use the Haversine formula freed up processing power to allow for added functionality beyond simple ground station pointing. These updates enhanced HERMES' usability, and feedback from other NEBP teams preparing for the October 2023 and April 2024 eclipse flights provided valuable insight into what features users desired most from this new GUI.

3. Features

HERMES was initially developed to serve as a functional pointing GUI. Once the calculations were implemented (and the tracking logic sufficiently flight-tested), the software was expanded to include a multitude of requested features ranging from non-Iridium radio integration to manual inputs and better user operability. The main HERMES GUI screen is shown in Fig. 6.

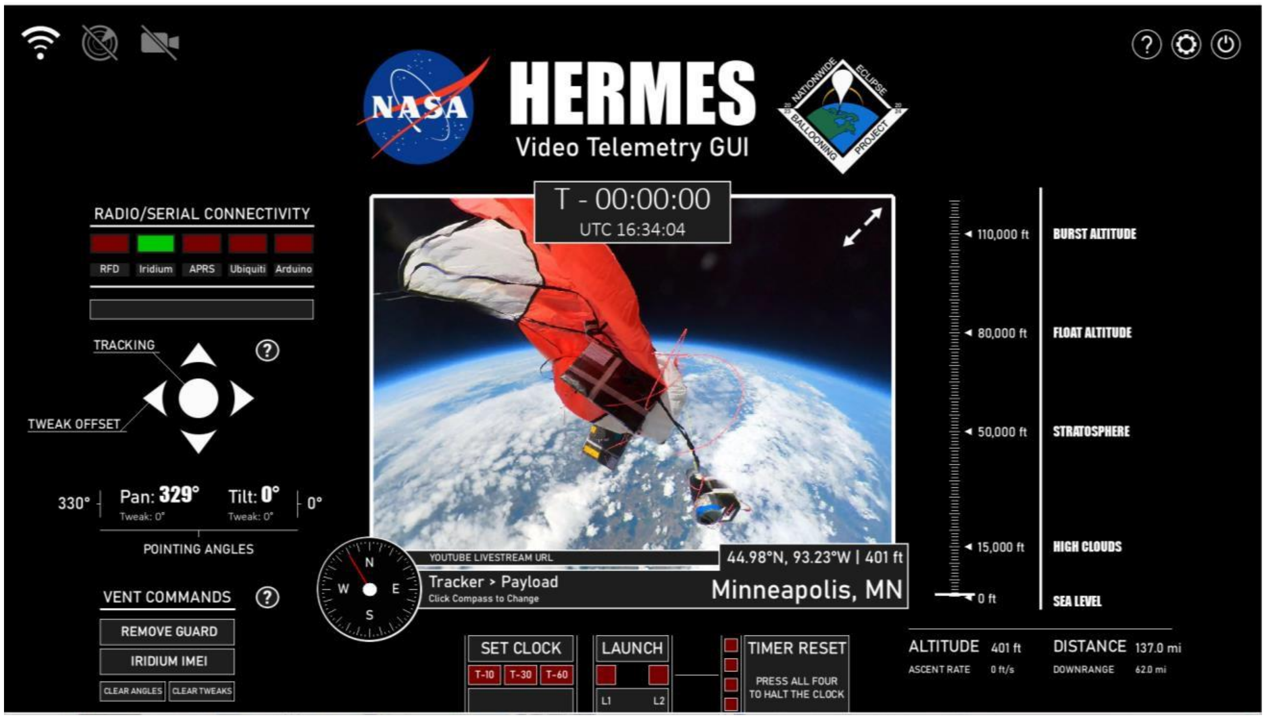


Fig. 6. Main screen of the HERMES Video Telemetry GUI with a default image, not an actual video feed, displayed in the middle.

3.1. *Peripherals*

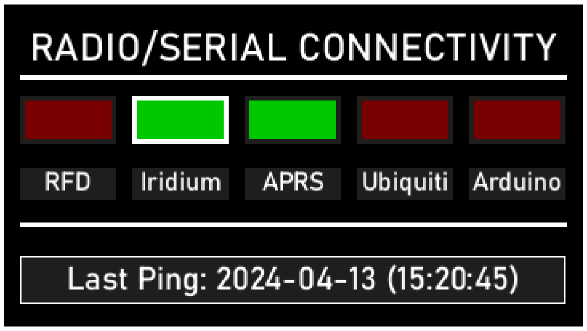
HERMES bundles a multitude of visual elements that continually update as the program receives new flight data. GPS information is displayed on the bar altimeter and on the location display, the latter of which also produces the name of any town or county that the balloon is overflying courtesy of OpenStreetMap's API. A compass widget points from the ground station to the payload, indicating the pan angle used to aim the tracking dish laterally. A clock indicates the time since launch (MET) as well as the current time (UTC). The pointing angles (and offsets) are shown next to an interactive control pad. The altitude, ascent rate, line-of-sight distance, and downrange distance of the payload relative to the ground station are displayed at the bottom of the GUI.

3.2. *Automatic Screen Capture*

When the flight timer is active, HERMES automatically records and timelapses the entire screen. This allows for post-flight fast-replay analysis by observing the compass, altimeter, clock, and location display, among other visual indicators, compressing a several-hour balloon flight into just a few minutes.

3.3. *Multi-Radio Tracking*

In addition to the Ubiquiti and Arduino connections, HERMES offers support for RFD900, Iridium, and APRS radios. These three GPS tracking systems can send latitude, longitude, and altitude information from the payload, which are then processed by the GUI to update the peripherals and calculate the pointing angles for the ground station antenna. As of version 1.4, HERMES can communicate with multiple radios in tandem, applying a filter to compare the data and reconcile any discrepancies. Iridium is the most accurate of the three, but its updates are the least frequent. If Iridium is active, all other radios will defer to it, but their GPS pings will be used to interpolate pointing angles between Iridium updates. Connectivity status is shown on the left portion of the screen (see Fig. 7).

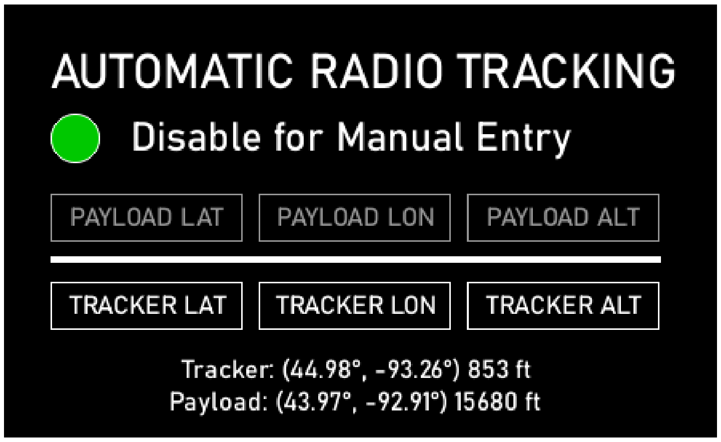


Last Ping: 2024-04-13 (15:20:45)

Fig. 7. Connection indicators for the three radios, video streaming, and the Arduino Uno in the ground station.

3.4. *Manual Tracking*

When any radio is active, the ground station will not point to the input coordinates unless the user specifically commands HERMES to begin tracking. However, the program will still update the position data from the payload by default. GPS updates can be disabled in the settings menu (or the radio connections turned off individually), and payload coordinates can be entered manually (see Fig. 8). Manual pointing inputs will take precedence over all incoming radio telemetry data.



Tracker: (44.98°, -93.26°) 853 ft
Payload: (43.97°, -92.91°) 15680 ft

Fig. 8. Manual input fields for the ground station and payload locations.

3.5. *Angle Manual Tweak Settings*

Even while the ground station is tracking automatically, users can adjust the tilt or pan of the its tracking dish by "tweaking," or adding offsets to the respective angles. This feature can also be used for initial calibration to orient the dish to point due north with antenna axis horizontal (i.e. parallel to the ground, if on level terrain), thereby establishing the (0,0) position for the two pointing angles. When the GUI is in use, all angles are continuously displayed on the GUI's main screen (see Fig. 9).

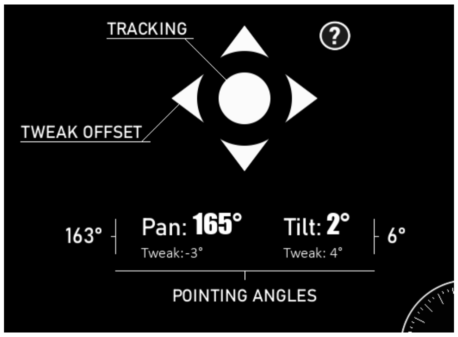


Fig. 9. Pointing angle readouts including the "tweak" offset angles.

3.6. *Livestream Display*

HERMES is capable of accessing and displaying, in the center of the GUI, a live video feed from a Raspberry Pi's RTSP address. This is the hardware that was standardized for NEBP engineering teams to downlink footage during eclipse flights. YouTube livestreams are also viewable, as are pre-recorded videos, by entering a URL in the appropriate field (see Fig. 10).

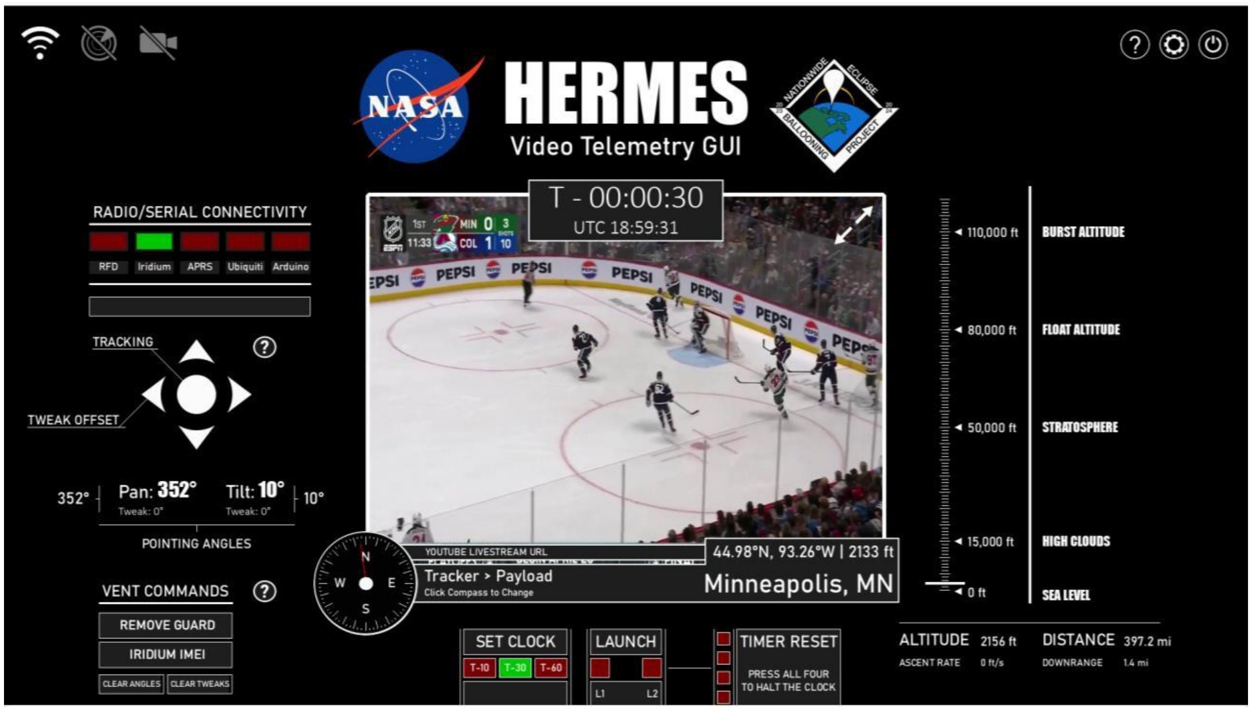


Fig. 10. HERMES' central screen, here showing NHL highlights but able to display a live-stream from a payload.

3.7. ***Data Logging***

In addition to CSV data files created for each active tracking radio, the program also outputs a map showing all used data points for the duration of the flight (see Fig. 11). This is constructed using the aggregate GPS data produced from the filter. Similar to the screen capture feature, map writing is active only while the flight timer is running.

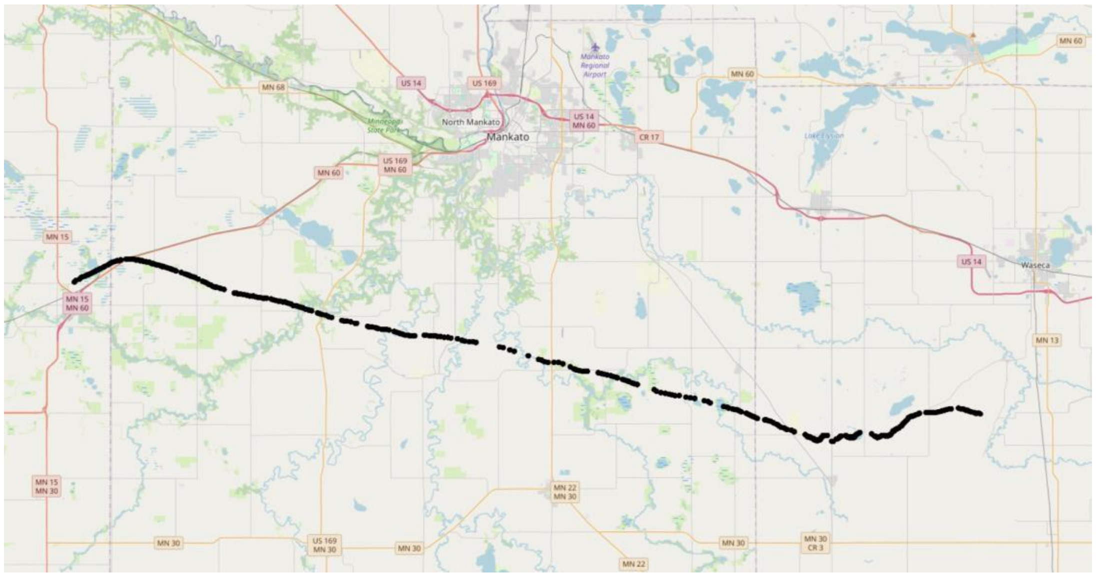
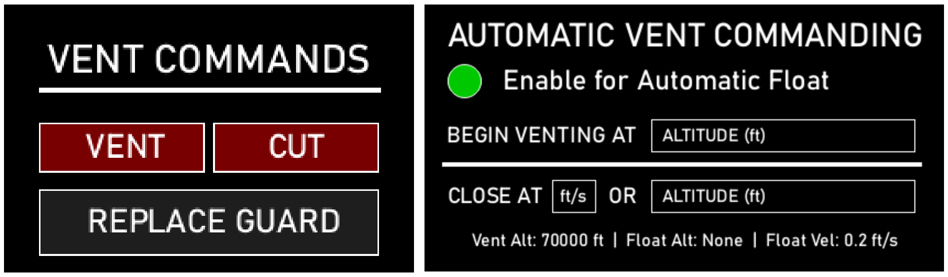


Fig. 11. HERMES-generated map showing the location history of a flight over southern Minnesota.

3.8. ***Venting and Cutdown***

During eclipse flights, many NEBP teams used a “vent” to release lift gas and attempt to “float” the payload at a constant altitude in the stratosphere. Typically, NEBP vent devices are controlled by an Iridium modem, as they can process commands sent by email to a central server. HERMES simplifies this task by automatically sending email messages with the proper subject lines and file attachments at the push of a button. Users can choose when to open and close the vent; when to cut the payload free from the balloon so as to terminate the flight; and whether to automate the float procedure by instructing HERMES to send vent commands when it detects a certain altitude and/or ascent rate. The HERMES vent command window is shown in Fig. 12.



Vent Alt: 70000 ft | Float Alt: None | Float Vel: 0.2 ft/s

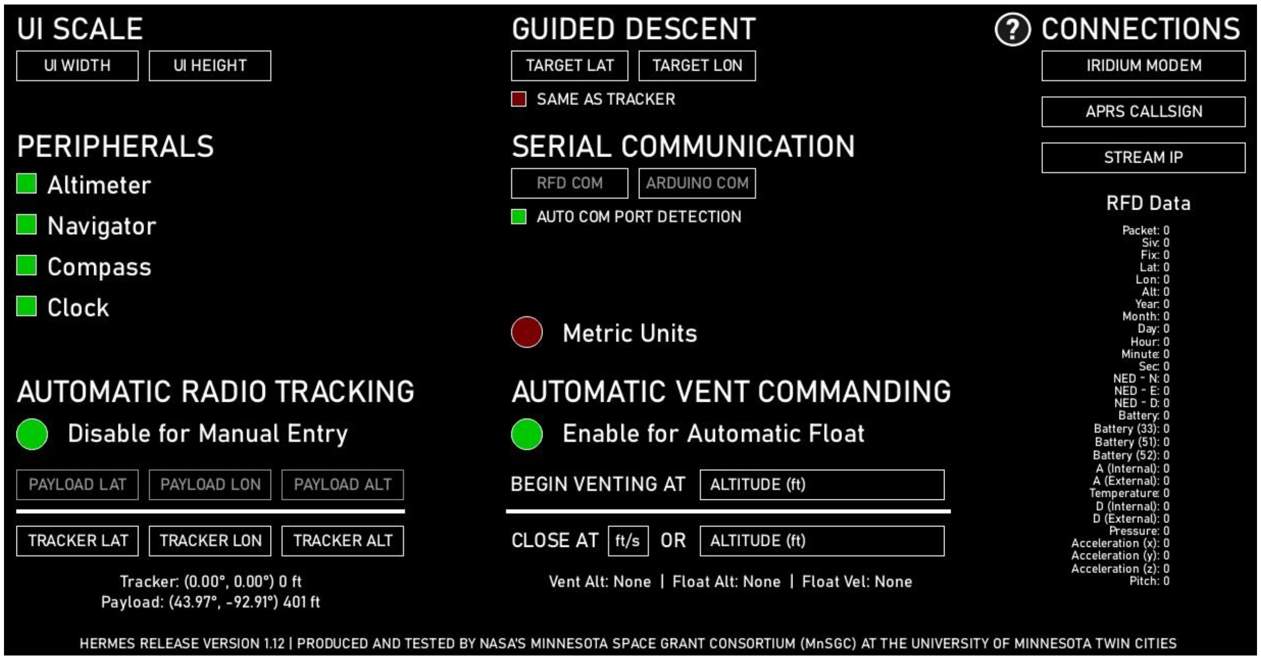
Fig. 12. Manual and automatic vent controls.

3.9. ***Housekeeping Features***

When using tethered Arduino devices like an RFD900 receiver or the ground station itself, HERMES automatically handles all COM port resets and exception handling. The user does not have to worry about reseating cables or restarting the program when switching connections, disabling radios, or changing settings. USB cords can be unplugged and re-plugged *ad infinitum* without any complaints from the software. Likewise, any radio failures, be it a sudden disconnection or prolonged unresponsiveness, will not cause HERMES any disruption. The full HERMES settings menu is shown in Fig. 13.

An Internet connection is not required to use HERMES software. If Internet connectivity is lost mid-flight, HERMES will continue executing its offline processes. Iridium and APRS data will be unavailable, but RFD900 line-of-sight tracking can be used in their stead.

If all GPS radio telemetry is lost, HERMES will continue pointing based on the last known balloon position. Should this happen, manual coordinate inputs and tweak controls can be used in tandem to continue pointing the ground station antenna at the payload to attempt to retain line-of-sight communication.



Tracker: (0.00°, 0.00°) 0 ft
Payload: (43.97°, -92.91°) 401 ft

Vent Alt: None | Float Alt: None | Float Vel: None

RFD Data

```
            Packet: 0
               Siv: 0
               Fix: 0
               Lat: 0
               Lon: 0
               Alt: 0
              Year: 0
             Month: 0
               Day: 0
              Hour: 0
            Minute: 0
               Sec: 0
           NED - N: 0
           NED - E: 0
           NED - D: 0
           Battery: 0
      Battery (33): 0
      Battery (51): 0
      Battery (52): 0
      A (Internal): 0
      A (External): 0
       Temperature: 0
      D (Internal): 0
      D (External): 0
          Pressure: 0
  Acceleration (x): 0
  Acceleration (y): 0
  Acceleration (z): 0
             Pitch: 0
```

HERMES RELEASE VERSION 1.12 | PRODUCED AND TESTED BY NASA'S MINNESOTA SPACE GRANT CONSORTIUM (MnSGC) AT THE UNIVERSITY OF MINNESOTA TWIN CITIES

Fig. 13. HERMES settings menu.

4. Flight Testing

Prior to HERMES, ground station tracking relied on the preexisting BRAD software developed by Montana State University's BOREALIS program, which was originally written for the previous nationwide eclipse program in 2017. [3] Once HERMES' pointing math was deemed accurate and reliable, our team transitioned to the new software for all 2023 NEBP practice flights.

Naturally, early testing revealed a myriad of issues with the GUI. Multi-radio tracking proved a major nuisance, as disagreeing GPS data fought for priority and caused the ground station to oscillate back and forth. Loss of Internet connection while tracking with Iridium caused the entire program to freeze and ultimately crash. An inactive RFD900 radio would do the same, as HERMES attempted to access unavailable data. When automatic vent commanding was first introduced, a subtle loop structure coding error resulted in HERMES flooding the Iridium server with almost a hundred emails when the "open" command was sent, causing the command link to the payload to fail altogether.

As our team performed more flights, and as HERMES was further developed over the summer, these bugs were ironed out. Better exception handling was implemented. Other interruptions and failures were passed more gracefully. Connection timeouts were added; functions were optimized and later multithreaded; redundancy was eliminated; and every loop in the code was triple-checked before HERMES was allowed to operate on another flight. Some issues, which did not manifest on our own ground station computer, plagued other teams attempting to use HERMES in its infancy. Thanks to their feedback and error reporting, the software was made more robust and reliable in the months leading up to the 2023 and 2024 eclipses.

5. Conclusions

The HERMES ground station GUI was officially released in its earliest open version in mid-2023. What was at first a means of pointing a ground station dish for eclipse practice flights blossomed into a versatile, multifunctional tool performing a variety of tasks on top of tracking a stratospheric balloon and pointing a parabolic antenna at it. All of the information that had been scattered across several windows and independent GUIs were consolidated into a single program: updates from multiple GPS radio systems, ground station antenna pointing angles, assorted API data, vent commanding options, and even the Raspberry Pi's video stream. All of these can now be viewed and managed simultaneously within a centralized GUI.

Although HERMES was originally developed for the purpose of tracking NEBP flights, it is still being maintained and updated after the conclusion of the Nationwide Eclipse Ballooning Project. Even teams that are no longer flying video streaming payloads or operating ground stations can still use this software for monitoring flight data, generating maps, commanding a vent, or listening to the Apollo 13 soundtrack while the flight timer counts down. Many planned features have already been implemented in some capacity, if only as basic framework. Eventually, HERMES will be able to generate live, customizable plots from incoming data; expand its radio catalog to include other GPS tracking systems; and even interface with the NEBP ground station to uplink commands to the payload directly instead of via Iridium servers. The scope of this software has expanded considerably since 2023 and it will only continue to be enhanced, optimized, and upgraded to provide users with more capabilities during any type of high-altitude balloon flight.

Acknowledgments

Development of the HERMES Telemetry GUI was made possible by support from NASA's Minnesota Space Grant Consortium (MnSGC) and the Nationwide Eclipse Ballooning Project (NEBP). I would like to thank the leadership of the national NEBP effort and also Dr. James Flaten for encouraging this work. I am also grateful to the NEBP ballooning teams who tested this software in open beta and helped me troubleshoot and improve it prior to the 2023 and 2024 eclipse balloon flights.

References

1 Azdy, R. A., & Darnis, F. Use of Haversine formula in Finding Distance Between Temporary Shelter and Waste End Processing Sites. Journal of Physics: Conference Series, 1500(1), 012104. https://iopscience.iop.org/article/10.1088/1742-6596/1500/1/012104/pdf

2 Esri Community. Distance on a sphere: The Haversine formula. Number 2017, Esri Community. https://community.esri.com/t5/coordinate-reference-systems-blog/distance-on-a-sphere-the-haversine-formula/ba-p/902128

3 Eclipse Ballooning Project. Engineering Lesson 8: Payloads and groundstations. Montana State University. https://eclipse.montana.edu/education/engineering-course/eng-lesson-08.html#brad

4 Cook, J. HERMES Documentation. Minnesota Space Grant Consortium. https://docs.google.com/document/d/1PRoLkXaMrUWXbg3vbNBbG_Q1igPCRx8imRnfndy7S-0/edit?usp=sharing