University of Colorado
Department of Aerospace Engineering Sciences
Senior Projects - ASEN 4028

Weather Resistant Autonomous Imaging for Tracking HEOs (WRAITH)

Project Final Report

Monday 4th May, 2020

Information

Project Customers

Name: Jonathan Aziz Email: jonathan.d.aziz@aero.org Phone: 719-375-6501	Name: Eric A Eiler Email: eric.a.eiler@aero.org Phone: 571-304-7854

Group Members

Project Manager Name: Julian ”Jurkoic” Email: julian.jurkoic@colorado.edu Phone: 970-218-1205	Systems Name: Jared Bartels Email: jaba9634@colorado.edu Phone: 402-972-5410
Imaging, Image Processing, & OD Name: Alex Kirk Email: alex.t.kirk@colorado.edu Phone: 253-230-4341	Environmental Systems Name: Shannon Chott Email: shannon.chott@colorado.edu Phone: 480-244-0847
Inputs, Propagation, & Scheduling Name: Noah Crisler Email: noah.crisler@colorado.edu Phone: 970-618-3527	Safety & Test Name: Matthew Barrett Email: matthew.barrett@colorado.edu Phone: 303-960-0772
Software Systems Name: Maya West Email: maya.west@colorado.edu Phone: 773-709-1128	Hardware Name: Austin Cyrus Email: aucy9911@colorado.edu Phone: 973-738-3884
Finance Name: Trevor Willson Email: trevor.willson@colorado.edu Phone: 720-245-3200	Software Hardware Integration Name: Cal Van Doren Email: caelen.vandoren@colorado.edu Phone: 303-437-6973
Manufacturing Name: Dalton Turpen Email: dalton.turpen@colorado.edu Phone: 505-239-8564	

Contents

List of Figures

List of Tables

Table of Acronyms

Table of Acronyms	
Acronym	Definition
HEO	Highly Elliptical Orbit
LEO	Low Earth Orbit
MEO	Medium Earth Orbit
GEO	Geostationary Orbit
SSA	Space Situational Awareness
GHOST	Ground-Based Hardware for Optical Space Trakcing
WRAITH	Weather Resistant Autonomous Imaging for Tracking HEOs
FOV	Field of View
TLE	Two Line Elements
COTS	Commercial Off-the-Shelf
FOD	Foreign Object Debris
OD	Orbit Determination
OSHA	Occupational Safety and Heath Administration
I/O	Input/Output
BMS	Battery Management System
SNR	Signal to Noise Ratio
Ra	Right ascension
PPP GPS	Precise Pointing Positioning Global Positioning System

Mission Statement

WRAITH's mission aims to demonstrate the feasibility of a low cost, autonomous, deployable space situational awareness system.

1. Project Purpose

Julian Jurkoic

The modern near earth space environment consists of a statistically estimated 750,000 pieces of debris having a size 1 cm or greater.[4] Given the extremely high velocity of this debris, it poses a serious risk to expensive orbital assets in the form of a potential collision. In fact, orbital collisions are identified by the European Space Agency as the second largest risk to all orbital assets.[5] In order to mitigate this risk, Space Situational Awareness (SSA) systems are utilized to identify and track potentially dangerous objects in orbit. SSA systems consist of a series of optical and radar sensors on the ground around the globe and in orbit. Currently, these systems only track objects of size 10 cm or greater, less than 20,000 of the 750,000 objects that measure 1 cm or greater[6], despite being technologically capable of tracking these smaller objects. This lack of coverage is due to the limited number of SSA stations, of which there are currently 20 very expensive installations. This poses a serious problem with risk of a mission threatening collision from debris which has not been tracked. This risk will only increase as more satellites are launched, especially with the reduction in cost of payload to orbit and the realization of large mega constellations.

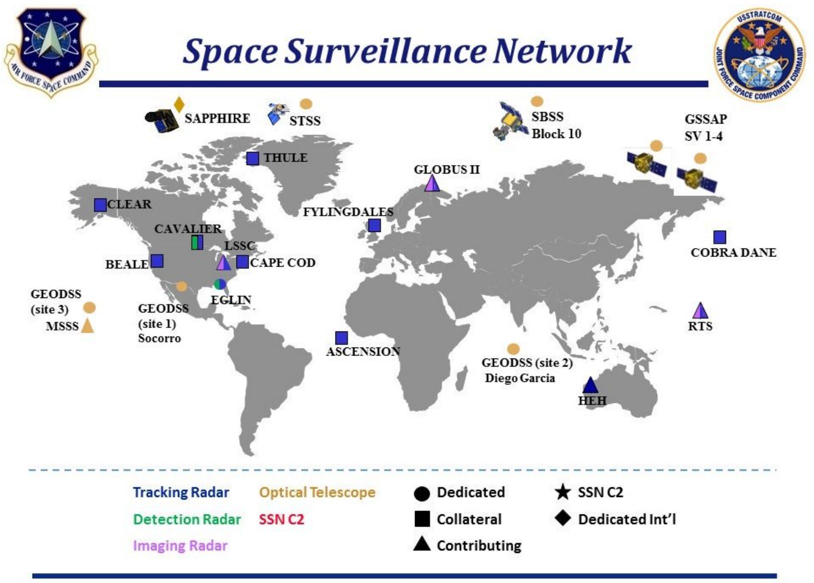


Figure 1. Current Space Surveillance Network

One method of addressing this risk would be to develop a less expensive, less capable SSA system that would be designed to track the larger, more easily tracked objects. This would free up time for the more capable, more expensive, existing SSA systems to track those smaller objects which are currently not tracked. To this end, the Aerospace Corporation commissioned the GHOST and, subsequently, the WRAITH projects. The GHOST (Ground Based Hardware for Optical Space Tracking) project laid the groundwork for the WRAITH project in the development of systems which could individually and with human input schedule an objects predicted position, image an object, get data from an image, and get an orbital determination for an object with that data for objects in circular low earth orbits (LEO) and medium earth orbits (MEO). WRAITH expands upon this project in three key manners. First and foremost, WRAITH will be fully autonomous after deployment, requiring no human input. As such, all of GHOST's individual systems must be integrated fully and further developed. Second, the system must be able to be deployed over a 12 hour period (a whole night). This creates a need for the system to be able to protect itself autonomously from environmental factors. Finally, the Aerospace Corporation has expressed a need for the tracking of highly elliptical orbits (HEO), and, as such, the WRAITH project will focus on tracking these objects.

2. Project Objectives and Functional Requirements

Jared Bartels

2.1. Success Description

The WRAITH team will provide an autonomous, palletized system that is capable of tracking LEO, MEO, GEO, and HEO space objects for a deployment of up to 12 hours. This system will also be capable of detecting adverse weather conditions in its immediate area and protecting itself from adverse weather during this deployment. The system will store the output of the updated orbit state, and, additionally, the system will also store all the information used to derive the updated orbit states. Along the way the team will document all design for hardware, software, and verification and validation for delivery to the customer and the project advisory board.

2.2. Levels of Success

Table 1. WRAITH Levels of Success

	Level 1	Level 2	Level 3
Scheduling	Accept list of SatIDs and sort based on FOV, time, and visibility constraints. Capability for up to 6 objects per hour.	Prioritize objects according to human input or probability of image capture.	Adjust schedule to search for a missing or maneuvered object, and issue an alert when this occurs.
Image Processing	Extracts endpoints of streaks at photometric SNR of 30 or less	Level 1	Determine Missing Space Object or Maneuver
Orbital Determination	Accurate orbit determination using Batch filter for HEO.	Level 1	Predicting possible orbits for missing objects.
Pointing	Tracking HEO orbits near apogee (GEO).	Tracking HEO orbits near perigee (LEO).	Search for missing objects in predicted possible locations.
Environmental Control	WRAITH retracts environmental protection in accordance to on-board sensors. Safety hardware will protect against moderate weather conditions (Appendix A) including light rain, wind, and limited temperature range.	Level 1	WRAITH retracts environmental protection in accordance to remote override from operators. WRAITH updates ground station with environmental state and safety hardware status.

2.3. Concept of Operations

WRAITH aims to further the functionality of the legacy GHOST senior project system by incorporating autonomy and weather resistance into the existing system. An illustration of the predicted WRAITH Concept of Operations during standard operating conditions is shown below in Fig. 2.

1. WRAITH will begin its 12 hour deployment by being unloaded from a transport vehicle in its palletized traveling form. Two operators will deploy WRAITH to the field and power the system on. Once powered on, WRAITH will take over and operate autonomously for the duration of the deployment (12 hours maximum).The system will be pre-loaded with a list of two line elements (TLE's) from NORAD IDs which WRAITH will use in its autonomous scheduling algorithm to create a list of prioritized object imaging locations for the deployment.

2. WRAITH will stay in safe mode when the system is in stand-by with no object passes occurring. Before the environmental shield is opened to track an object's pass, WRAITH will use its on-board weather sensors from the weather sensor suite to determine if conditions are safe for operation.

3. If the weather conditions are deemed safe, WRAITH will begin its imaging sequence. If the weather is deemed to be too dangerous to operate in, WRAITH will stay in safe mode until the next object's window where it will check the weather again before exiting safe mode. With safe weather, WRAITH will maneuver the lens into position by the locations given from the scheduler and take a long exposure image of the sky.

4. WRAITH will immediately analyze the captured image to determine if the object was present.

5. If the object was present, two more exposures of the same object will be taken at certain times of the night given by the scheduler, as three total exposures are needed for orbit determination.

6. If the object was not present in the first exposure, WRAITH will begin a searching pattern to look for the missing object over the rest of the pass as seen in the Missing Object CONOPS in Fig 3.

7. Once the pass is over, WRAITH will analyze the three exposures and provide an updated orbit estimation of the object. WRAITH will complete this autonomous process by returning to safe mode until the next object's pass.

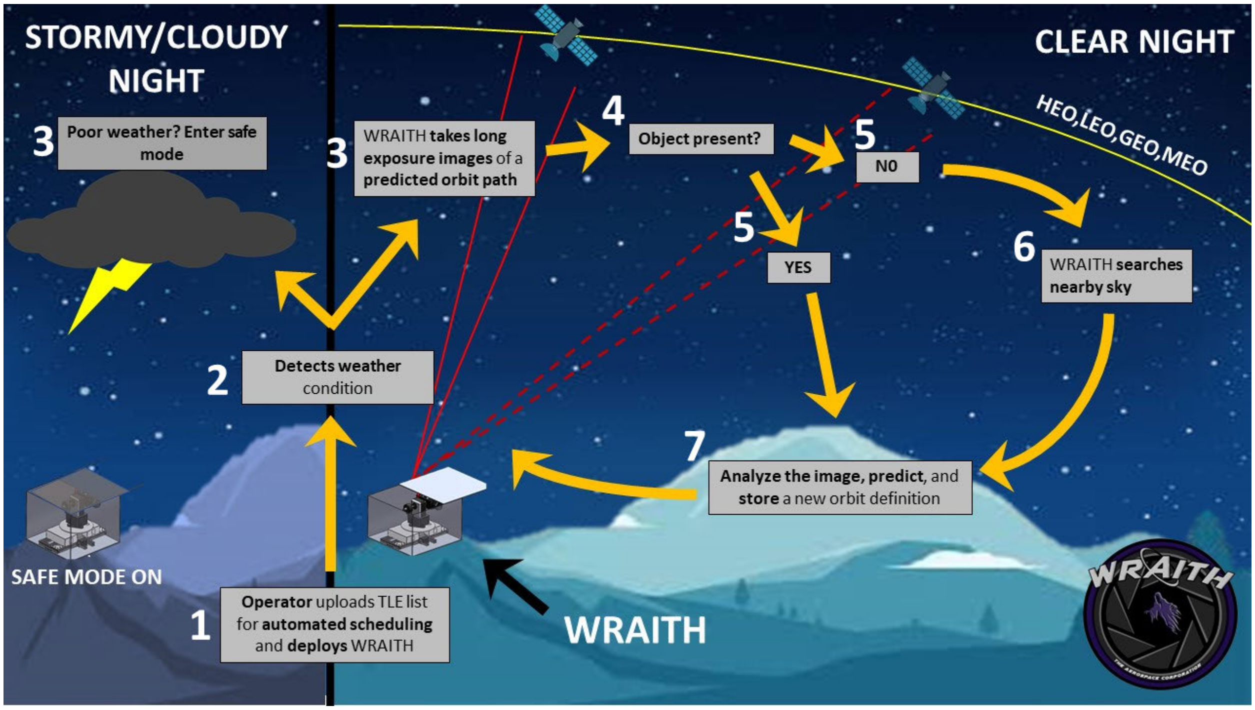


Figure 2. WRAITH Concept of Operations.

To go into more depth on the sequence of events in the case of a missing object during image processing look at Fig. 3 below. If after exiting safe mode to take an image of the currently queued space object the image processing software recognizes the space object was missed then the algorithm described in 3.1.2 is initiated. This algorithm goes through the process of imaging ahead of the objects predicted path to try and locate the object in the night sky. Whether the object is found or not, that information is stored and WRAITH will return to safe mode until the next viewing window given by the live updating scheduler.

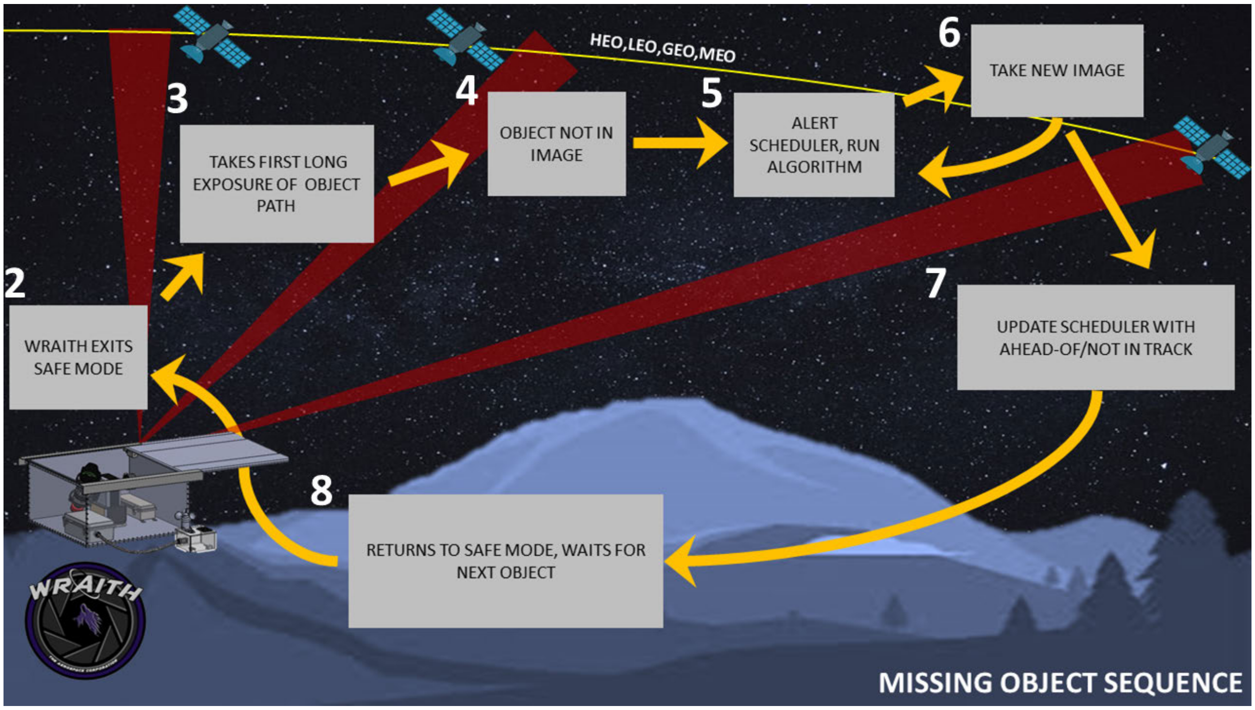


Figure 3. WRAITH Missing Object Concept of Operations

2.4. System Functional Block Diagrams

2.4.1. Hardware Functional Block Diagram

The overall functional block diagram (FBD), seen in Fig. 4 allows visualization of how each separate system on WRAITH is communicating with the others, and how the flow of information occurs. The sequence of events is as follows:

1. The system begins with an operator uploading a list of Two Line Elements (TLE's) to the main processor on WRAITH.
2. Once the TLE's have been uploaded to the on-board processor it takes over scheduling while creating actuation stage and camera commands. This will be discussed more in depth in the software functional block diagram and the later software sections of the report.
3. At all times the weather status is being checked via an incoming data stream from the weather detection sensor suite. Note that at any time if an operator deems the weather unsafe for operation, a manual override can be issued via cellular network. The signal is received on-board the weather detection suite and passed along to the on-board processor.
    (a) If the weather status is deemed safe, lid actuation commands are sent to the active weather protection system, opening WRAITH's lid.
    (b) If the weather status is deemed unsafe the weather protection system is then sent closing lid actuation commands.
4. Immediately prior to an object image window Right ascension and Declination commands are sent to the actuation stage in the form of serial port ASCII communication.
5. When an object's image window has started the on-board processor then sends imaging commands to the camera through serial port communication.
6. The camera transfers the captured image back to the on-board processor for processing.

7. The on-board processor calculates and produces an updated orbit state of the object to be stored in system memory and retrieved by an operator after deployment.

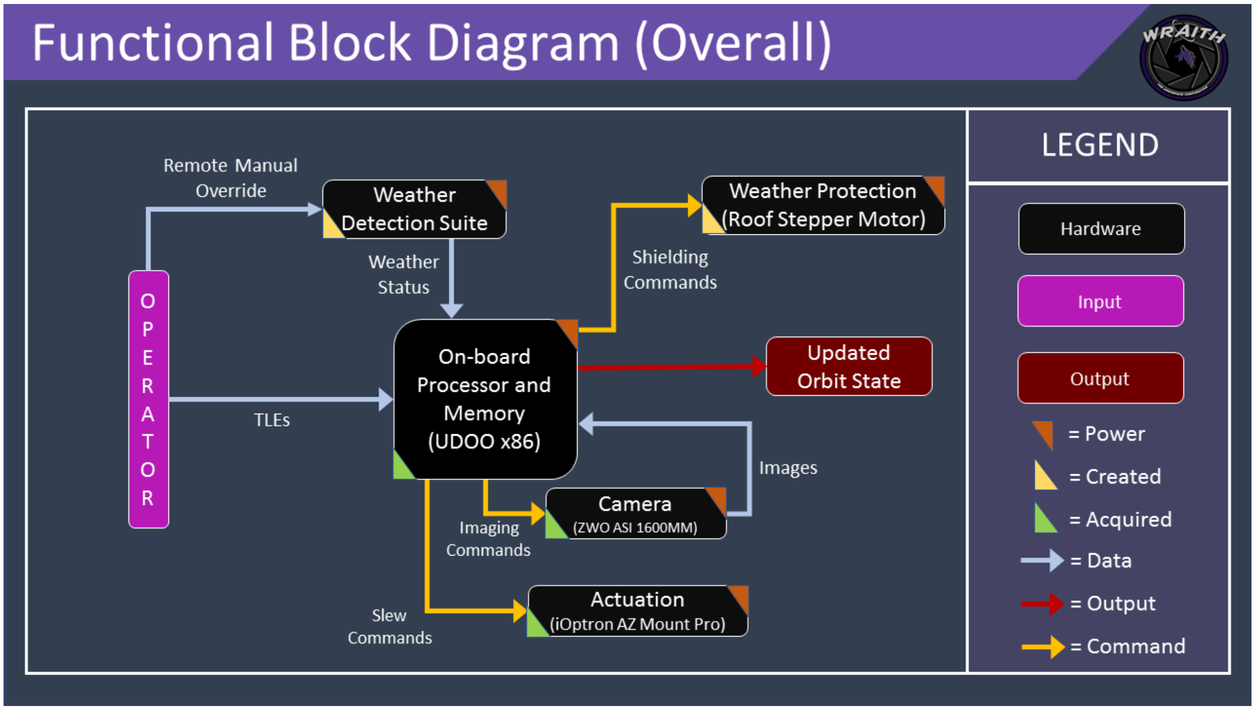


Figure 4. WRAITH Hardware Functional Block Diagram

2.4.2. Software Functional Block Diagram

The on-board processor and weather detection suite seen in Fig. 4, houses the entirety of the software modules running on WRAITH. The interactions of each of the sub modules inside the processor and weather suite, and how the data flows through the system is seen in Fig. 5. The data flow, while always running on a continuous loop, follows the sequence of events listed below. The in depth description of each module is saved for the detailed design sections, instead only the high level functionality of each module will be presented here.

1. The data flow begins with the inputs into the visibility scheduler, consisting of its location and time via the GPS module, the NORAD IDs of all desired objects to be tracked via an external USB drive, and the current state of the weather via the safety override module.

2. The visibility scheduler then creates and propagates all the objects TLEs to create viewing times and positions of each object. The viewing time and position of each object are next transferred to the hardware command module.

3. Receiving the viewing time and position of each object the hardware command module translates these into language the actuation stage and camera can read. Directly prior to sending these commands to the respective hardware, the module reads the current state of the safety override module to determine if its safe to continue on with operations.

4. If the weather was deemed safe for operation the hardware commanding module first sends actuation commands to the room motor, then slewing commands to the actuation stage to position WRAITH in the right location for imaging, and finally imaging commands to the camera to capture the object.

5. The captured image from the camera is then moved to image processing in order to determine if there was a successful capture of the object in frame.

    (a) If the image was deemed successful it can be passed on to orbit determination

   (b) If the image was deemed unsuccessful a notification is sent to the visibility window scheduler to trigger the reschedule algorithm.

6. The orbit determination receiving the successful image capture calculates new Keplerian orbit parameters and moves it to on-board storage where it can be offloaded completing a single iteration of the data flow.

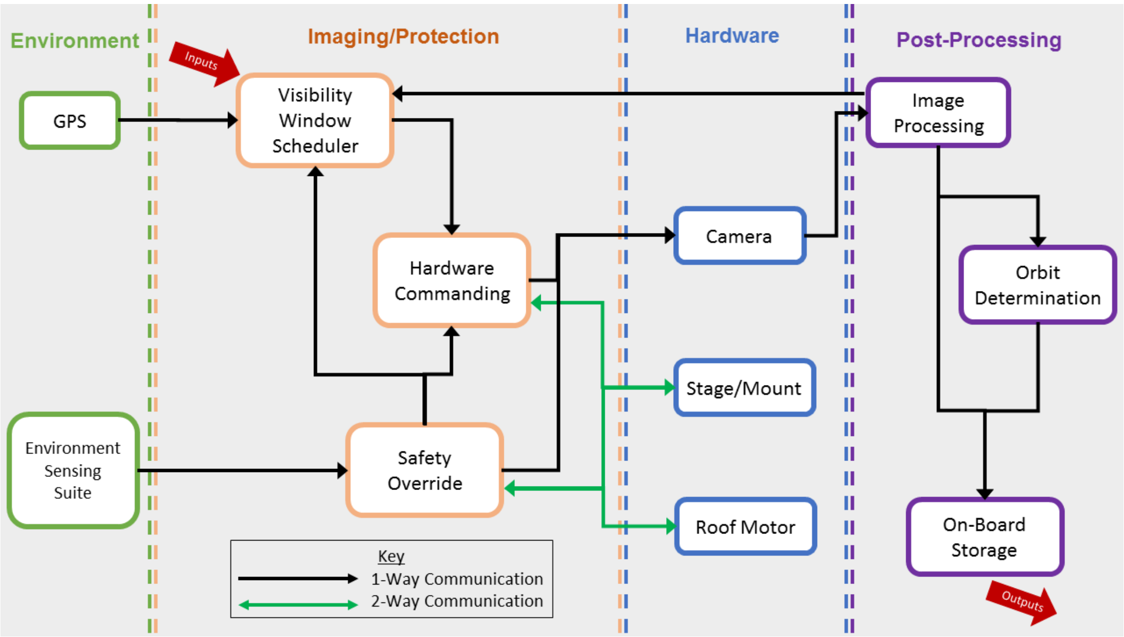


Figure 5. WRAITH Software Functional Block Diagram

2.5. Functional Requirements

WRAITH has 7 functional requirements that define the overall functionality of the system. These functional requirements will drive the design requirements and ultimately the trade study processes. The rationale behind the functional requirements stems from wanting to gather and create an exhaustive, abstract list of requirements from the objective and threshold requirements that were given to the team by the customer, The Aerospace Corporation. These functional requirements are listed below:

FR 1: The system shall schedule predicted locations and visibility windows for objects in LEO, MEO, GEO, and HEO.

FR 2: The system shall function as defined by the *Standard Operating Conditions* document[a] with no human intervention for 12 hours.

FR 3: The system shall autonomously enter and exit a safe mode to protect itself from adverse weather.

FR 4: The system shall autonomously point and track objects in LEO, MEO, GEO, and HEO.

FR 5: The system shall image objects with an apparent magnitude of light less than 10[b] and process images within 10 seconds.

FR 6: The system shall create and save an orbit estimate for each object imaged within five minutes of the end of the visibility window.

---

[a] For *Standard Operating Conditions* reference Appendix B

[b] Note that apparent magnitude of a star is a number that tells how bright that star appears from Earth. The scale is logarithmic with larger magnitudes correspond to fainter stars.

FR 7: The system shall be deployed in 30 minutes and broken down in 30 minutes by two operators.

3. Design Process and Outcome

3.1. Software

3.1.1. Legacy Software

Alex Kirk

As mentioned in the conceptual design overview, the baseline software was designed by the GHOST team. This section will describe in detail the most critical elements of the GHOST software, as well as discuss the final functionality the team was able to achieve. Once again, the most detailed description of the GHOST software, including details of code segments within the higher level modules, can be found in their project final report.[c]

3.1.1.1 Scheduler

On any given night of deployment, multiple objects will need to be imaged. These objects may have visible passes that overlap, meaning the processor needs to know the order in which it should attempt to image objects. The scheduler exists to automate this process and ensure that an imaging order is well-defined throughout the deployment. The scheduler pipeline is as follows:

1. **Receive NORAD IDs**: The user inputs the list of NORAD IDs containing all the objects they wish to image during the deployment to an .obs file. The file also includes the observation start and end times determined by the user. At this point, the user can assign their own weighting to objects, which essentially sets the prioritization of the objects.

2. **Reading TLEs**: Once the NORAD IDs are input to the system, the scheduler pulls the most recent Two-Line-Element (TLE) from Space-track.org for each ID. A backup catalog is stored on a USB flash drive in case the system is not connected to the internet and the scheduler needs to retrieve a TLE. This catalog will not be the most up-to-date, so it is best to pull the TLE's from the internet and to also frequently update the USB.

3. **SGP4 Propagation**: The TLE's need to be propagated forward in time to predict the object location past what is given in the TLE. GHOST completed a trade study and determined a semi-analytic propagation algorithm, SGP4, was an appropriate method to generate the orbit predictions. SGP4 models the problem with two body effects, in addition to drag and solar radiation. The final algorithm results in a position and velocity vectors in the True Equinox, Mean Equator coordinate from which can be easily converted to the Earth Centered Inertial Frame.

4. **Constraining Observation Opportunities**: The scheduler takes all objects position as a series of points in time and determines those which will be visible to the GHOST module for imaging. Only the points that are deemed to be visible are saved. The constraints on visibility include the satellite elevation, sun elevation, moon location, brightness, and the Earth's shadow among others.

5. **Weighting Passes**: As mentioned earlier, visible passes may occur at the same point in time. If this occurs the scheduler uses assigned weights to add priority to all objects and determine the imaging order. Weights can be determined based on a passes' maximum elevation, brightness, and the age of the TLE being used.

6. **Scheduling**: The scheduler then assigns three imaging sequences for an object during its pass. These sequences occurring at the beginning, middle, and end of a visible pass. If two, or more, windows conflict then the lower weighted objects sequence is adjusted until there is no longer any conflict. Adjustments can also occur if the scheduler determines there is not enough time to slew between two windows.

7. **Output**: The final output of the scheduler pipeline is a .cmd file. This file contains the times and pointing angles necessary for the actuation system to capture the objects in its pass, and also tells the system when to initiate image processing.

---

[c] https://www.colorado.edu/aerospace/current-students/undergraduates/senior-design-projects/past-senior-projects/2018-2019/ground-based

In its final design, GHOST's scheduler was able to successfully accomplish steps 1-5. However, successfully being able to command the stage is functionality the WRAITH team will need to complete the implementation of. In addition, the WRAITH team will be adding functionality to live schedule during a deployment based on missing objects, this is discussed in detail in 3.1.2.

3.1.1.2 Image processing

Once an object has been imaged the software needs to extract useful data. The data of most interest to both the GHOST and WRAITH teams are the right ascension and declination measurements of an object. These terms are passed into the orbit determination to define a new orbit. When an image is taken, it appears in the form of a streak, shown in Fig. 6. This is the result of taking a long exposure image. The streak endpoints are the points from which the RA/DEC is determined. The GHOST team broke down image processing into 3 main components.

Figure 6. ISS Image Capture

1. Identification of image boresight orientation.
2. Identification of object streaks
3. Conversion of streak pixels (x,y) to right ascension and declination measurements.

Boresight identification is knowing where the mount was actually pointing and what portion of the sky was actually imaged. GHOST completed a trade study on the best way to do this and decided to use astronomic calibration. This being the use of a star catalog to determine pointing. They utilized the Astrometry.net software package for this astronomic calibration. Astrometry.net being an open-source star registration algorithm which identifies an image location based on known star patterns [7]. This software also retrieves the right ascension and declination of the center of the image, while providing the baseline information for a streak template. Results of the Astrometry.net package are shown in Figures (7) and (8).

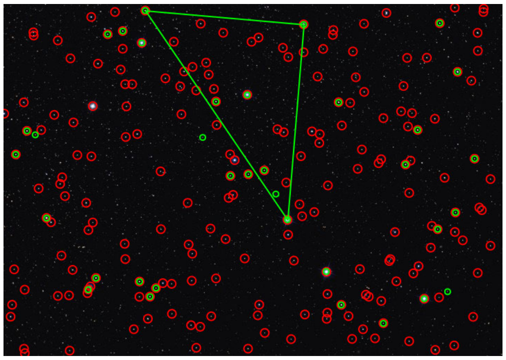


Figure 7. Star Identification

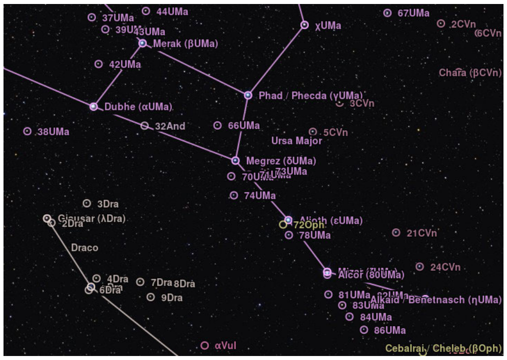


Figure 8. Identified Star Cluster

The best method for detecting streaks was another GHOST trade study. Ultimately, a match filter was the best option. A match filter is a SNR filter that uses a priori estimate of a streak's orientation and size to identify the streaks in an image. Using predictions of the objects rate of right ascension and declination, combined with the imaging time results in a streak estimate. This estimate is combined with the image containing the streak to determine the streak which most closely matches the estimate. The filter also determines a spread σ representing the accuracy of the steak and real image.

Finally, when a streak has been processed the location must be mapped from (x,y) coordinates to right ascension and declination pairs. After significant research, the GHOST team decided to use the WCS file generated by Astrometry to directly convert pixel location to right ascension and declination. In its final state, the GHOST image processing is able to determine the right ascension and declination for streak when processed manually. Meaning the image processing software was called by a user with specific inputs, not as an integrated call from the main processor. The WRAITH team will integrate this module to the scheduler and other software components. The baseline functionality will not be modified, but some research may be conducted on the effects of multiple streaks in one image. This question was not addressed by GHOST. Also, the GHOST team did not use Astrometry.net on the local processor, rather they used the online web version of the software. WRAITH has already downloaded the package and added the functionality for it to run directly on the processor, removing any reliance on the internet to run the image processing.

3.1.1.3 Orbit Determination

The entire purpose of both GHOST and WRAITH is to determine orbits of objects. While neither team has a requirement on the specific accuracy of the orbit estimates, the ability to take images and determine an objects orbit is the defining deliverable of the project. Orbit determination is the subject of research across both industry and academics. There is no method that is widely accepted as the optimum filtering method and there are many options that exist with varying pros and cons. GHOST analyzed both batch filtering and Kalman filtering to determine which is most useful for their project. As mentioned in the conceptual design, GHOST ended up using a batch filter. A batch filter is a least squares regression filter with both linear and non-linear capabilities. The filter begins with a physical model that represent the equations of motion for a given body. The model then creates an expectation of where the body will be at a given time. By comparing the estimate and the physical data from image processing, the co-variance of the body location is found. Using the time derivatives of the model states, an estimate is formed for the location of the object at the next data point. GHOST implemented a non-linear batch filter, which needs a total of 6 data points to define an orbit. These 6 points come from the 3 images taken during an object pass and the 2 resulting endpoints of each streak. The GHOST software is able to determine the orbit of an object in its current state, but once again needs to be integrated with the rest of the system. As a result, WRAITH will be treating the main structure as a black box, while integrating it with the rest of the system. Time will be spent quantifying the accuracy of the orbit determination during the testing phase next semester. Details of this testing can be found in the verification and validation section of this report. This concludes the discussion of the GHOST software, the report will now move into the specific changes and functionality that the WRAITH team will be adding.

3.1.2. Live Scheduler

Noah Crisler, Maya West

The need for a live scheduler, meaning a scheduler that is able to adapt and change the throughout the deployment, stems from two scenarios that could cause WRAITH to miss a previously scheduled object image. The first case causing a missed image opportunity is WRAITH's active weather sensing/protections system deeming the current conditions unsafe for operation, ultimately causing an aborting of the image sequence. The second case for a missed object would be the object's propagated orbit location and its true location not being accurate (due to a change in orbit elements caused by drag encountered in the atmosphere), this would mean when WRAITH takes the image at the pre-determined right ascension and declination no object will be present in the captured image. If either of these two cases were to happen, WRAITH needs to posses the ability to alert the scheduler of the missed image/object and request that it be rescheduled following an algorithm per **FR 2 and 4**. The sequence of events described above is presented below in list form:

Live Scheduler Sequence of Events

1. WRAITH begins deployment and runs the scheduler for the entire deployment, creating imaging times for each object.
2. During the deployment an Image/object is missed:

    I Object is missed due to bad weather
        i. Hardware command module, writes to a file that the object was missed due to weather.
    II Object is missed due to discrepancies in propagated and true location
        i. The image processing module, detects the blank image and writes to a file the object was missed to due orbit propagation

3. The scheduler running on indefinite loop, detects that the missed object file has been written to.
4. The scheduler runs the missing object algorithm (discussed in detail below)
5. A new schedule is created, and new commands are sent to the hardware module
6. The scheduler deletes the missing object file and waits until a new one is detected

The sequence of events and how one function interacts with the others can be visualized with the following FBD of the live scheduler. In the FBD below the function colored orange would only be altered for the specific missed object, the functions colored red are then altered for all missed objects as the entire schedule is re-optimized.

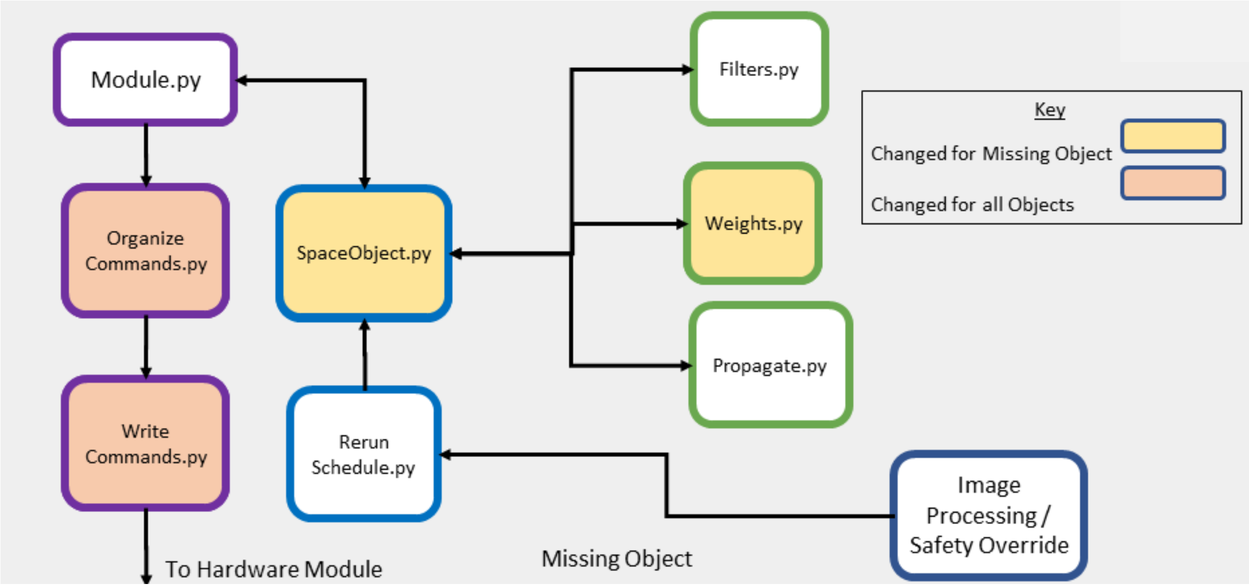


Figure 9. FBD of live scheduler

Following the above sequence of events, the scheduler employs an algorithm when it reschedules a missed event. The algorithm was designed with the goal of giving WRAITH the best chance of imaging the object later in the deployment with out interfering with other already scheduled objects. The algorithm is broken into two categories and is implemented differently depending on weather the object was missed due to bad weather or discrepancies in its predicted location. The steps for each case are presented below:

Bad Weather Algorithm

1. 1st miss of object
    - Do nothing, don't alert scheduler
2. 2nd miss of object
    - Alert scheduler for a rerun, automatically increase the objects weight.
3. 3rd miss of object
    - Alert scheduler for a rerun, further increase the objects weight.
4. 4+ misses of object
    - Alert scheduler, further increase object weight

Discrepancy in Location Algorithm

1. 1st miss of object
    - Do nothing, don't alert scheduler
2. 2nd miss of object
    - Alert scheduler, Move Camera one FOV ahead in track of predicted position
3. 3rd miss of object
    - Alert scheduler, Move Camera two FOV ahead in track of predicted position
4. 4+ misses of object
    - Alert scheduler, remove all weight from object, allowing for other objects to imaged in its place

In both algorithms the first miss does not trigger any action by the scheduler, one miss was deemed not important enough to alter the already optimized schedule. For the case of further misses due to bad weather, the scheduler is rerun and with each additional miss the object weight (which determines whether or not it will be imaged when there are conflict object image windows) is increased by an amount that will be determined through testing. Now the algorithm designed for objects missed due to an error in the object's propagated and true location works similarly, once the object has been missed twice the algorithm will change the third image location to be one camera field of view (FOV) ahead in track of the object (seen below in Fig. 10). The camera is moved ahead in track due to the likelihood that a blank image has occurred due to drag interactions between the object and the atmosphere; because the objects location was found by propagating forward an old TLE, there is a chance that the orbital elements of the respective object have changed since it was last observed. Specifically, the object could have experienced high amounts of drag causing its orbit to decrease in altitude. This decrease in altitude is coupled with an increase in the objects velocity, ultimately resulting in the object being ahead of the predicted location. Once this is complete the scheduler updates the hardware commands and waits for the objects next image time. If the object is missed a third time, the process outlined above is repeated. Now the fourth image opportunity is changed from its predicted location and instead placed two camera FOV ahead in track. If the object is missed a fourth time, it was determined that the location of that object is not known well enough for it to warrant further imaging. The objects weight is set to zero, allowing all other possible objects to be imaged ahead of it in the event of conflicts in the schedule.

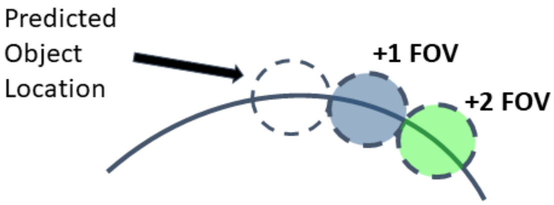


Figure 10. Visualization of discrepancy in object location algorithm

In order to move the camera field of view accurately from its predicted location to either 1 or 2 FOV ahead in track the following algorithm was developed, and is depicted below in Fig. 11:

Algorithm for Moving Camera FOV

1. The objects right ascension and declination vectors are curve fitted in order to increase the resolutions of each.
2. Using the camera FOV angle of 4.47^o a distance function is employed between the original Right Ascension and Declination to two points on the line, forward and backward in the track.
3. The point corresponding to the forward direction is chosen, and the new right ascension and declination replace the old ones for the respective image.
4. The updated hardware commands are passed to the hardware module.

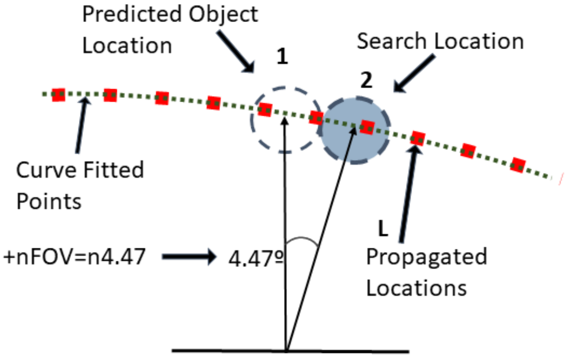


Figure 11. Discrepancy in object location algorithm math logic

3.1.3. Software Hardware Integration

Cal Van Doren

3.1.3.1 Hardware Commanding Module

One of the python modules that will be developed for the WRAITH project is the hardware commanding module. The goal of this piece of code is to coordinate hardware commands to the camera, stage, and roof motor in order to image in nominal conditions and protect by closing the roof during non-imaging times. It is important to note that this piece of code will not include the capability to close the box when adverse weather is detected from the environmental sensing suite however, that capability will be built into the safety override module which will be discussed later on. There are four key capabilities that will be part of this hardware commanding module: command scheduling at absolute times, camera shutter commanding, stage actuation commanding, and roof motor commanding. The overall hardware commanding architecture is shown in Figure 12 in which each of the four key capabilities is clearly identified by a different color.

The first capability that will be outlined is the hardware command scheduling capability. This will be completed using the scheduler package that is available through python. What this package does is allow the user to designate commands to get executed at specific absolute times, these commands are then built into a python object that is referenced each time the on-board clock iterates, which occurs often, if there is a command matching the on-board system clock time then that command will be executed. This functionality allows the software to run through and schedule every command that is required for a visibility window well before that window occurs. For a nominal visibility window, the scheduler module will be used to schedule all of the commands for that window at the completion of the previous visibility window; this means that at any given time there will only be one visibility window's worth of commands in the scheduler and thus if that window is canceled then all of the commands will be wiped out and there will be no accidental commands sent to any piece of hardware. This commands scheduling method will simply repeat at the end of each visibility window and the only condition that would stop this from occurring is if the safety override module has raised the safety override flag which will be discussed in a later section. Now that the methodology used to schedule the nominal commanding has been outlined, the command pipeline for each piece of hardware will be defined.

The first, and most simple, piece of hardware to command is the camera. Ultimately, the only commands that need be sent to the camera are shutter open and shutter close commands. There will not be a closed feedback loop for the shutter status back to the hardware commanding module because the shutter status is not critical to the safety of the box as will be seen with the stage and roof motor commanding. The camera will be commanded by establishing a Local Area Network (LAN) connection between the camera itself and the UDOO main processor on-board the WRAITH box. Establishing this LAN connection is accomplished through the use of the indi server package in python and the commands themselves that will be sent are part of the user-defined python package ZWO_ASI_LIB which has been used by Steve Marple at MIT to accomplish a similar project using the same camera that the WRAITH team will be using. The relevant commands that will be used for WRAITH are shutter open commands accompanied by exposure times and gain values which will be determined by the scheduler based on the angular rate of the object across the sky and its anticipated brightness.

The next hardware commanding path that must be defined is the stage commanding path. As shown earlier in this report, the stage chosen for this project is the iOptron AZ Mount Pro. It's important that the stage chosen comes from iOptron because they support a built-in ASCII commanding language that can be used to communicate through serial communications straight from the main processor on the WRAITH box to the stage itself. Similarly to the camera, the stage will be interfaced through a LAN connection that will allow two-way serial communication between the UDOO and the stage. Commands for stage actuation will be sent to the stage while the stage's actual pointing will be verified back to the UDOO after each command has been sent to the stage. Important commands that will be used to control the stage are: defining the target right ascension, defining the target declination, commanding the stage to slew to the target, commanding the stage to slew to the home position, and querying the stage for its current pointing. The ability to query the stage for the pointing will be crucial for ensuring the safety of the stage because the roof cannot be closed until the stage has been verified to be pointing such that the stage will not be in the way of the roof's closure path.

The final piece of hardware that must be commanded via software is the roof stepper motor. This interface will not have to be constructed by the WRAITH team because the stepper motor can be commanded by any standard stepper motor driver. Stepper motor commands will consist of the direction to actuate (clockwise or counter-clockwise) and the number of steps to actuate in that direction. While the number of steps to open or close the roof will be known, it is possible that the stepper motor could slip in which case the step count would not be accurate. In order to counter this risk, the roof will be equipped with open and closed limit switches that will be able to verify the status of the roof. These switches will be utilized by creating a while loop anytime the roof is commanded opened or closed. Within the while loop will be a command to move one step in the desired direction to open or close the roof and that single command will be executed continuously until the switches read the desired roof location. This created a very simple software design that is only at the mercy of the failure of the limit switches, and the risk of that event can be mitigated by also counting the steps on the motor and creating a hard stop after a certain number of steps. Again the closed loop feedback will be vital because the stage will not be allowed to move until the roof location is verified.

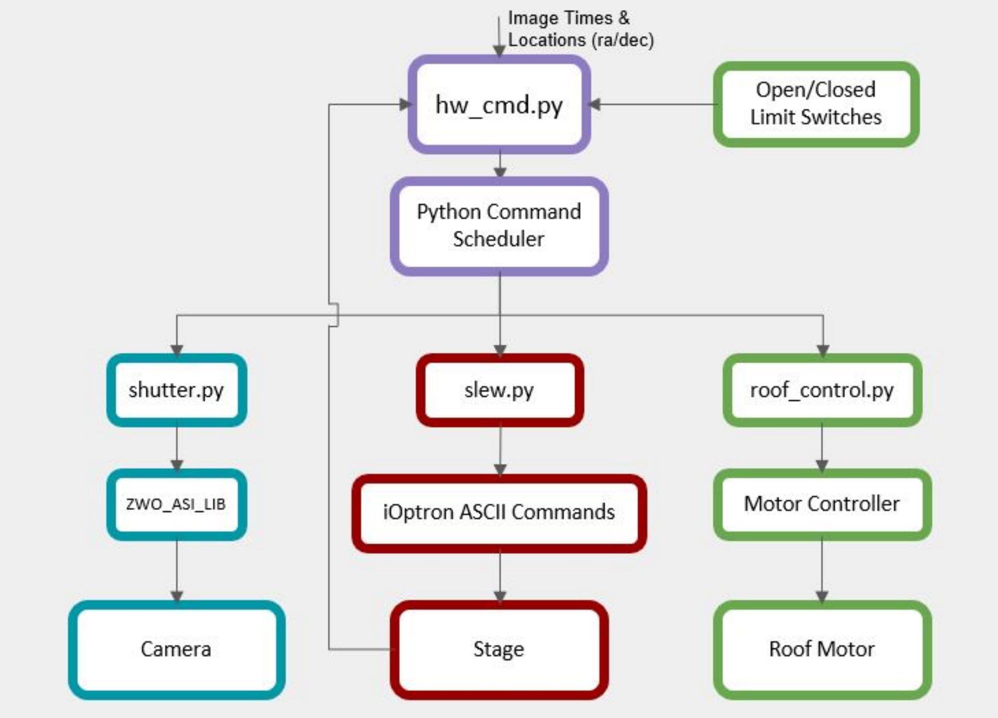


Figure 12. Hardware Commanding Architecture

3.1.3.2 Safety Override Module

The safety override module, as alluded to in the previous section, will have the capability of wiping all of the currently scheduled commands and taking over the hardware commanding to close the box in the event that the environmental sensing suite communicates that the weather conditions are outside of the acceptable weather conditions as defined in the environmental sensing suite software section. The only piece of code that will be checking the pin status from the environmental sensing suite will be this safety override software; there will be a stream of binary 0's and 1's coming from the environmental sensing suite at a rate of 1 Hz. A 1 value means that the weather is adverse and the box should be closed while a 0 value means that the box can be open and the weather is acceptable. If there are three consecutive 1 values detected from the environmental sensing suite then the safety override module will raise the safety override flag which will be a simple global variable, this will stop the hardware commanding module from scheduling any new commands. From that point, the safety override module will clear any scheduled commands in the python scheduler module and implement immediate commands to slew the stage home and close the roof to protect the internal components. Once the roof had been confirmed to be closed, the time will be noted in the safety override module and the box will not be commanded open for at minimum 30 minutes from the closure time; this is because the WRAITH team wants to be conservative with the weather safety of the box and if the weather conditions are deemed adverse it is unlikely that the weather will be acceptable for at least 30 minutes. This condition does not mean that the box will open 30 minutes later, but it means that the 1's and 0's from the environmental sensing suite will not be evaluated for 30 minutes. Once the 30 minutes has elapsed, three consecutive 0's from the environmental sensing suite will allow the hardware command module to function nominally again and the safety override flag will be lowered.

Another important functionality of the safety override suite will be to command the scheduler to reschedule the commanding windows for any objects that are missed during the box closure. When the box closes, a reschedule command will be sent to the scheduler to reschedule as if the box were to open again in exactly 30 minutes. When the 30 minutes has elapsed, if the sensing suite still says the box must be closed then another command will be sent to the scheduler to again reschedule as if the next 30 minutes will be missed. This will be continued until the weather is deemed safe again and the box is reopened and WRAITH is returned to nominal operations.

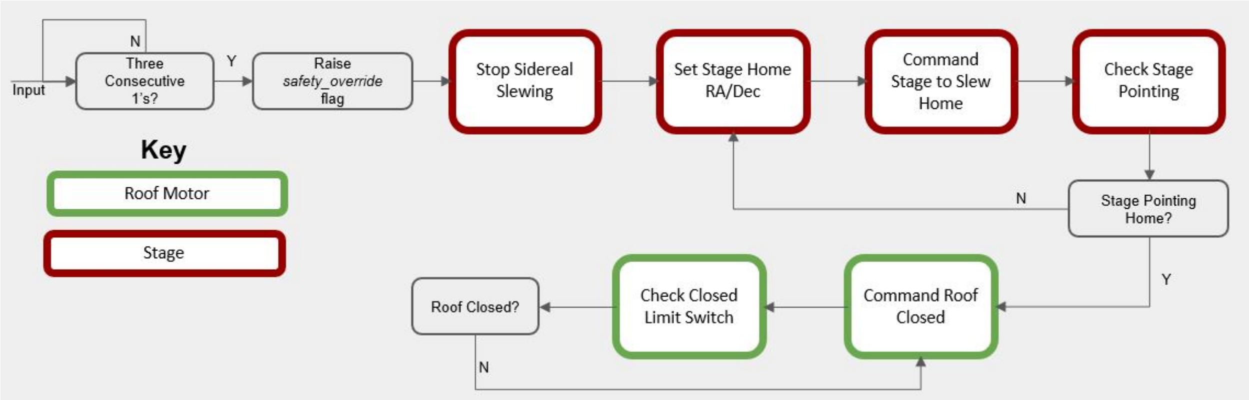


Figure 13. Safety Override Architecture

3.1.3.3 Nominal Commanding Concept

It is important to define a nominal operations concept for what the hardware commands will look like during a nominal visibility window uninterrupted by the safety override suite. Figure 14 shows the logic and a high level definition of this sequence.

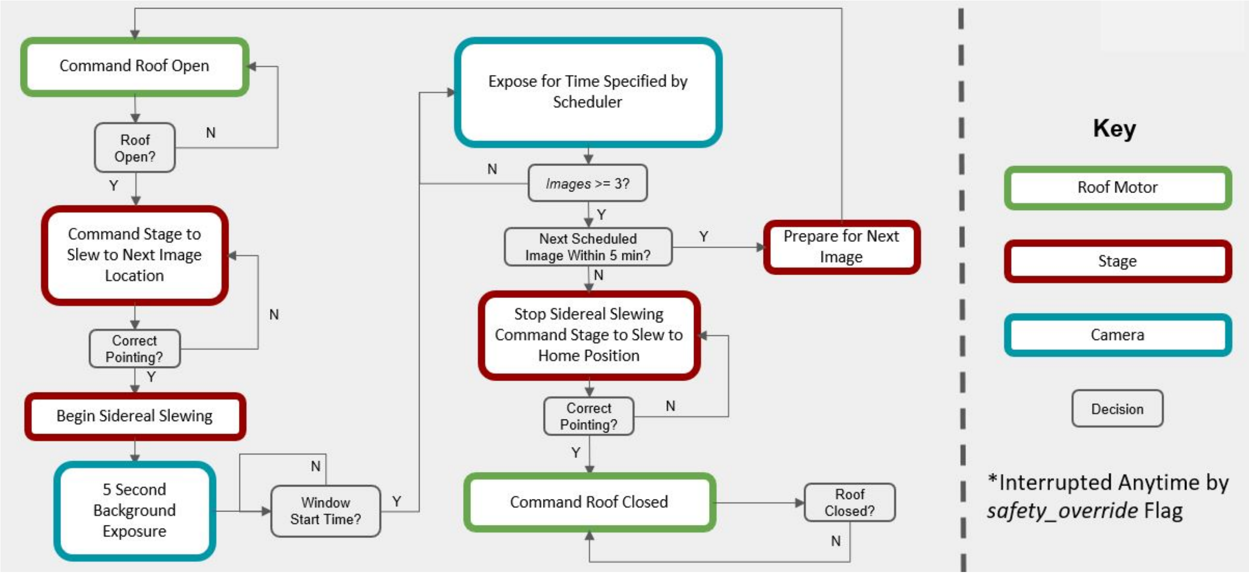


Figure 14. Nominal Commanding Sequence

The sequence shown in Figure 14 is spelled out in more detail in the following list of hardware commands that would be sent during this nominal visibility window. This process would begin two minutes before the scheduled start of a visibility window.

1. Command stepper motor to actuate one step in the open direction while the limit switches don't indicate an open box
2. Command target right ascension to the value specified by the scheduler
3. Command target declination to the value specified by the scheduler
4. Command the stage to slew to the target coordinates
5. Query the stage for its current point, if it is not pointing at the target return to step 2
6. Command the stage to enter sidereal slewing mode
7. Set camera exposure time to the value specified by the scheduler
8. Command the shutter open to take background exposure of stars
9. Wait until the scheduled start time of the visibility minus the length of an exposure
10. Command the camera shutter open three consecutive times
11. Check the time of the next visibility window, if it is within 5 minutes then go to step 2
12. Command that stage to stop sidereal slewing
13. Command target right ascension to the value of the home position
14. Command target declination to the value of the home position
15. Command the stage to slew to the target orientation
16. Query the stage for it's current pointing, if it does not match the target return to step 11
17. Command the stepper motor to actuate one step in the closed direction until the limit switched verify the roof is closed.

3.2. Weather Detection

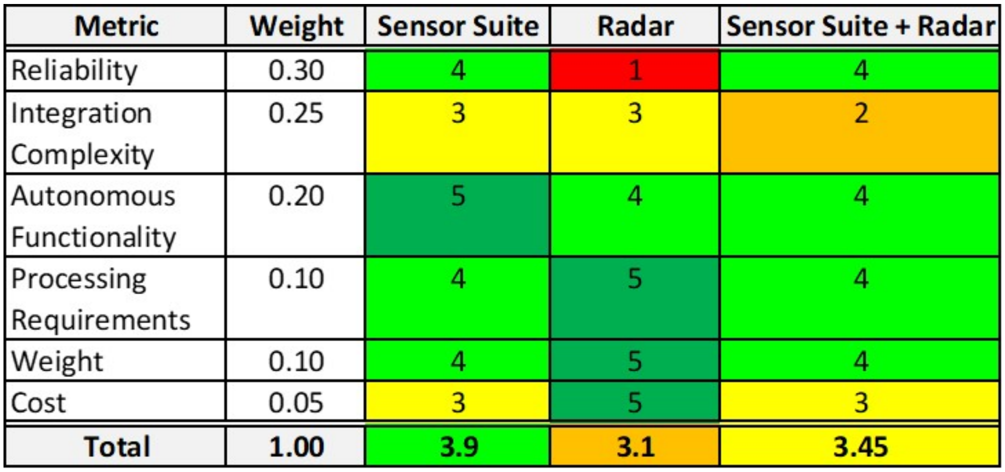
Shannon Chott, Trevor Willson

The weather detection system will be used to detect the weather at and near WRAITH's deployment location so it can autonomously protect itself from any adverse conditions in accordance with **Functional Requirement 3**. Since there are multiple factors that can contribute to adverse weather, the weather detection system is comprised of a suite of sensors including an anemometer, a barometer, a precipitation sensor, a lightning detector, a relative humidity sensor, and an infrared thermometer. This system will also incorporate a microcontroller to drive the various weather sensing devices while also analyzing the outputs of these sensors. This particular design is driven by **Functional Requirements 2 and 3**. These requirements establish that the system shall operate in the Standard Operating Conditions (Appendix B) and that the system must operate with no human intervention for 12 hours. In addition, the system shall receive start-up and shut-down commands from an offsite operator. Due to these operational conditions, software must be in place to properly determine the weather conditions to ensure the main enclosure is open during standard conditions and properly protected during adverse conditions. The sensor suite software serves the purpose of determining the current weather conditions and acts as the medium between the main processor and the operator. This will be done through a C++ based structure that consists of one continuous loop; the first step in the loop will receive commands from the off-site operator over the LTE connection, while the second part of the loop will operate the sensors and analyze current weather conditions. The microcontroller will then pass inputs to the main UDOO processor to pass the desired state of the system. The following sections will discuss the trade studies for this weather detection system, the detailed design that resulted from the trade study, and the overall structure of the logic for the microcontroller while also outlining the LTE and serial connection interfaces for the sensor suite.

3.2.1. Weather Detection Trade Study

Accurate weather detection is vital to WRAITH, as a false-positive in adverse weather conditions would impact observation windows. Conversely, a failure to detect adverse conditions could jeopardize the integrity of the entire system. An accurate system will be able to quickly detect the following: high winds, any precipitation, lightning, clouds, pressure changes, and relative humidity changes. These parameters are indicative of most adverse weather conditions expected at a typical WRAITH deployment. To ensure that the weather detection was accurate while remaining within the scope of the project, multiple design ideas were considered. The first design the team explored was the implementation of an On-Board Sensor Suite which would house a variety of sensors to monitor the weather conditions at the deployment site. These sensors would include a barometer, humidity and temperature sensors, anemometer, lightning detector, IR thermometer, and a precipitation detector. In order to best minimize risk by avoiding false-positives, the shutdown logic for the system would consist of a few single-trigger shutdown sensors, as well as a few multiple-trigger shutdown sensors. A further breakdown of the risk assessment process and the impact of the individual sensors is provided in the following **Conditions** section. The second design consideration was radar-based Weather Communication by Radio/Cellular communication which would utilize Doppler radar measurements through web applications. This method would strictly rely on the Doppler readings for the deployment location and would not require any live information from the deployment site. The final consideration was a combination of the two aforementioned methods which would provide added redundancy to the system by analyzing the Doppler radar readings in addition to the live measurements at the site. A trade study was performed to optimize this system, utilizing the following metrics: Reliability, Complexity/Integration, Autonomous Functionality, Processing Requirements, Weight, and Cost.

Table 2. Weather Detection Trade Study Results



Metric	Weight	Sensor Suite	Radar	Sensor Suite + Radar
Reliability	0.30	4	1	4
Integration Complexity	0.25	3	3	2
Autonomous Functionality	0.20	5	4	4
Processing Requirements	0.10	4	5	4
Weight	0.10	4	5	4
Cost	0.05	3	5	3
Total	**1.00**	**3.9**	**3.1**	**3.45**

As seen in Table 2, it was determined that the best design method would be the On-Board Sensor Suite, as it performed the best in comparison to the Doppler radar or the combination method. Notably, the study found the Sensor Suite method was more reliable than the Doppler radar method and had better autonomous functionality than both the Doppler radar and the combination method. A full breakdown of this study, along with the individual weight and score assignment, can be found in Appendix A.1.

3.2.2. Sensor Selection

Once the sensor suite design was chosen, multiple types of sensors were selected to provide sufficient observations of the weather at the deployment location. The types of sensors that were selected are outlined below:

- **Anemometer:** Anemometers measure the wind speed which is an important environmental parameter for a few reasons. First, dust and other particles need to be prevented from getting inside the main box so they don't coat the lens and decrease the image quality. Additionally, too high of wind speeds could shake the main box and decrease the image quality, negatively impacting the system's ability to accurately image object passes. Lastly, increased wind speeds are often indicative of changing weather. This is because winds follow the pressure gradient force, so the wind flows from high pressure to low pressure. When wind speeds increase in an area, it's a good indicator that the pressure is dropping and inclement weather is moving in. To avoid dust and debris pick up as well as possibly predicting stormy conditions, the weather detection system needs to be able to measure the wind speed.

- **Relative Humidity Sensor:** Relative humidity (RH) is a measure of how close the dew point temperature is to the ambient air temperature. In other words, it is a measure of how much moisture is in the air. While a high relative humidity percentage does not necessarily mean there's precipitation, it could mean other wet conditions such as fog or morning dew. Additionally, a high relative humidity percentage combined with low pressure would give a greater confidence that precipitation was in the area, since low pressure means the air is rising. As the air rises, it cools and condenses. If there's a lot of moisture in the air, there will be more condensation and rain may fall. Since many of the components on-board WRAITH are electronic and thus sensitive to water, it's critical that the moisture content is measured to attempt to predict wet conditions that could compromise the electronics and hardware.

- **Barometer:** Barometers measure the surface pressure in millibars or other similar units. As stated briefly in the above sections, low surface pressure is indicative of stormy conditions. Rising air leads to low pressure at the surface. This is because as air rises, it cools and condenses onto particles in the air such as cloud condensation nuclei. If the particles become heavy enough, they will fall through the cloud as precipitation. If there's higher pressure than usual at the surface relative to the surrounding areas, the air is sinking and thus the skies will be clear. For example, Boulder is generally around 835mb, but when a low pressure system moves in there can be pressures recorded as low as 825mb depending on the system. A drop in pressure of about 3mb over the course of a day is generally indicative of stormy weather. Measuring the pressure and watching for any significant pressure drops is critical for predicting adverse weather conditions such as rain or snow.

- **Infrared Thermometer:** Infrared (IR) thermometers measure the temperature of an object without the need to contact it. They generally have a wide field of view and they take in everything in that field of view to determine the temperature. An IR thermometer pointing at the sky will give a good indication of whether or not there's cloud cover overhead. A clear sky will typically read about 1°C while a cloudy sky may read somewhere around 18°C[21]. This is because when the Earth emits longwave radiation into the atmosphere, the water vapor that makes up the cloud will absorb and re-emit some of that longwave radiation. Since the cloud is emitting more IR radiation than the clear sky, the IR thermometer will read a higher temperature. Knowing if the sky is clear or not is critical for a couple of reasons. Cloudy skies could mean precipitation, which could compromise the safety of the electronics and hardware. Additionally, WRAITH needs a mostly clear sky to be able to image object passes. An IR thermometer is an easy way to autonomously determine cloud cover.

- **Lightning Detector:** Lightning detectors can detect when there's a lightning strike nearby and estimate the distance to the storm head. Many commercially available lightning detectors monitor RF disturbances to suggest lightning in the area. This particular method is effective since the resonant frequency of the sensor is 500KHz, which can usually catch lightning without receiving false-positives. This weather condition is important in satisfying Functional Requirement 2 since where there's lightning, there's most likely a thunderstorm with heavy precipitation. Again, precipitation would greatly compromise the safety of the system, so it's critical to know if there's a storm around and lightning is a strong indicator of this.

- **Precipitation Sensor:** Precipitation sensors detect how much moisture is present on the board of the sensor. This is important because predicting the weather is difficult and involves a lot of dynamic elements in the atmosphere, however detecting the current weather at a location is much more reliable. A precipitation sensor will be able to read if it's raining or snowing at that moment in time at the deployment location. This will serve as a last line of defense if the other sensors fail to detect inclement weather. The precipitation sensor will detect that there's active precipitation and WRAITH will enter safe mode before too much damage is done to the more sensitive components.

Once the weather detection sensors had been selected, specific choices had to be made to comply with the driving requirements: **Functional Requirements 2, 3, and 7**. Functional Requirement 2 dictates that the system shall operate in Standard Operating Conditions (detailed in Appendix B) with no human intervention for 12 hours, so the sensors for the weather detection system must be able to detect conditions within these specified nominal ranges. Functional Requirement 3 states that the system shall autonomously enter and exit a safe mode to protect itself from adverse weather, so the sensors must be constantly running and detecting the weather for the duration of the deployment. Lastly, Functional Requirement 7 dictates that the system must be deployed in 30 minutes or less by two operators, so the weather detection sensor suite cannot exceed the bounds of operable size or complexity. Additionally, the sensors must be able to start up and reach steady-state values in 30 minutes or less. The sensors also must be able to detect a range of conditions that contain the defined shut-down conditions for the main box to enter safe mode. These shut-down conditions are discussed in a later section, seen in Fig. 18. The layout of each sensor in the weather detection system can be seen in Fig. 15.

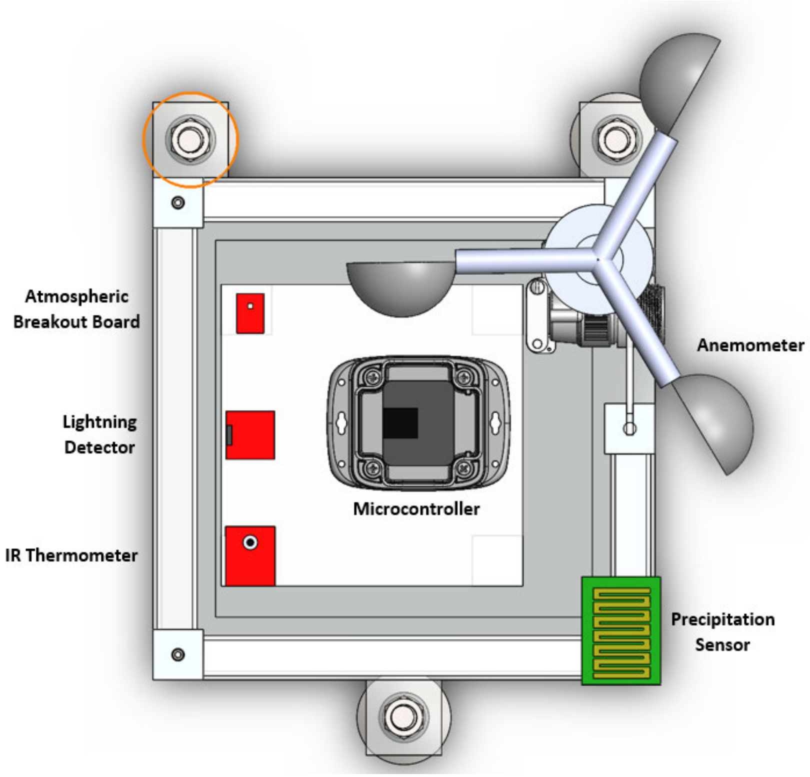


Figure 15. Layout of Weather Detection Sensor Suite

It is important to note that for each of the sensors outlined in the following sections, the shut-down conditions were chosen conservatively and with a built-in margin of error to account for the different accuracies of the sensors and in compliance with the Aerospace Corporation who advised that safety be valued over all else. For example, the anemometer has a shut-down condition of $8\frac{m}{s}$. This wind speed is where dust and debris begins to be picked up, however it's on the lower end of that interval so if the anemometer reads just below $8\frac{m}{s}$, it will not cause any significant damage to WRAITH. None of the sensors require an extremely high degree of accuracy since the shut-down conditions were selected to be as safe as possible as opposed to on the edge of where WRAITH would critically need to be in safe mode.

3.2.2.1 Anemometer

The specific anemometer chosen for the weather detection system is the **Adafruit Cup Anemometer**[15]. This anemometer is capable of measuring wind speeds from $\mathbf{0\frac{m}{s}}$ **to** $\mathbf{32.4\frac{m}{s}}$. The Standard Operating Condition for wind speed is less than $8\frac{m}{s}$ given that the shut-down condition is $\mathbf{8\frac{m}{s}}$ or higher, so this particular anemometer meets Functional Requirement 2. The wind speed of $8\frac{m}{s}$ was chosen as WRAITH's shut-down point because according to the Beaufort Scale, $8\frac{m}{s}$ is roughly the point where dust and debris start to get picked up by the wind[22]. It is also above the average wind speed in Boulder, CO (the expected deployment area for WRAITH) of about $4\frac{m}{s}$ so there should be minimal lost imaging opportunities due to this specific case. In addition, this anemometer is small and lightweight, about 112g, corresponding to Functional Requirement 7. The anemometer also has a simple pin out of 12V of power, a ground, and an analog signal, so it will be easy to integrate with the chosen microcontroller.

One concern for the anemometer measurements was the influence of the moon-roof box on the wind velocity. In order to determine the necessary placement of the anemometer in relation to the main WRAITH enclosure, the team conducted a CFD simulation to quantify the perturbations from the main enclosure. This model is presented in Fig. 16 where the nominal wind velocity is $8\frac{m}{s}$. As seen in the model, the WRAITH enclosure has significant impact on the wind velocity, therefore prohibiting placement close to the main box. The result of this model is that the anemometer must be placed at least 8 feet from the main unit since the perturbations become negligible beyond this distance.

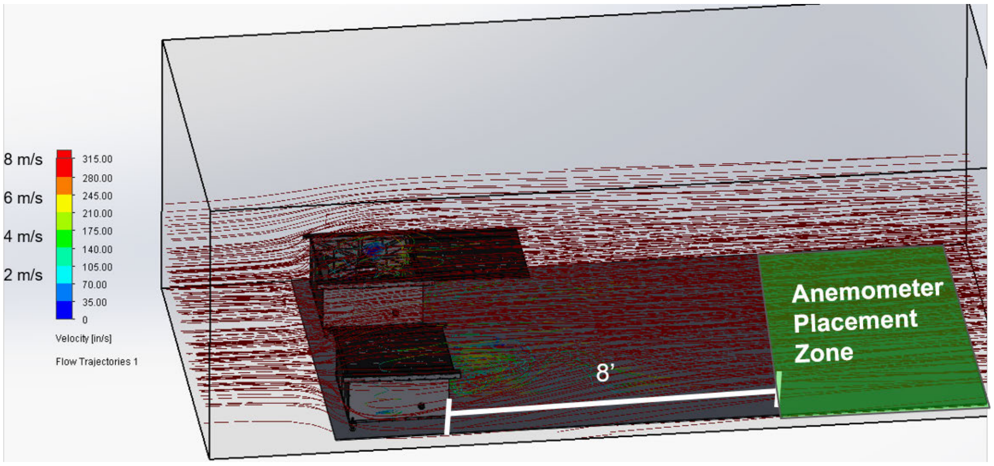


Figure 16. CFD Analysis for Anemometer Placement

3.2.2.2 Relative Humidity Sensor and Barometer

The relative humidity (RH) sensor and barometer were chosen as a combined breakout board, the **SparkFun BME280 Atmospheric Sensor Breakout**[16]. This breakout board contains both a relative humidity sensor, a barometer, and a temperature sensor. As per the Standard Operating Conditions and the chosen shut-down conditions, the relative humidity for a nominal deployment should be below 70% as the combined shut-down condition for relative humidity is greater than or equal to **70%**. This specific RH sensor is capable of measuring between **20% and 90%** relative humidity and the values WRAITH needs to detect are within this range, satisfying Functional Requirement 2.

As for the barometer component of the breakout board, the software will contain a combination shut-down condition of a **pressure drop of 3mb**. The barometer on the breakout board can measure from **300mb to 1100mb** with an accuracy of $\pm 1mb$. Boulder, CO generally has a surface pressure of about 845mb, while sea level is about 1013mb. This barometer is more than capable of detecting even significant drops in pressure due to its wide range. For reference, Mt. Everest has an approximate pressure of about 333mb, and the eye of a hurricane will drop the pressure from sea level (1013mb) down to about 920mb. A hurricane is outside of the scope of extreme weather expected to be seen by WRAITH, however the barometer chosen would be able to detect even this extreme of a drop in pressure, again satisfying Functional Requirement 2.

In addition to being able to measure well within the required ranges, this breakout board is incredibly small and lightweight, making the mass almost negligible in the total mass budget, in compliance with Functional Requirement 7. It also operates on the I^2C interface, making the integration with the microcontroller and other sensors simple as it requires only 4 pins: 5v of power, a ground, SCL, and SDA.

3.2.2.3 Infrared Thermometer

The chosen infrared thermometer for the weather detection system is the **SparkFun MLX90614 IR Thermometer**[19]. In accordance with the shut-down conditions, there is a combined trigger for safe mode of an IR temperature detected greater than **10°F** (or about -12°C). This specific IR thermometer has an object temperature range of **-70°C to 380°C** and an accuracy of $\pm 0.5^{\circ}C$. The shut-down condition is well within the capabilities of the sensor. As was determined experimentally through testing, a clear sky will often read low temperatures of about -25°F, while a cloudy sky will read higher temperatures of about 18°F. These values are experimentally determined so there is some room for variability here until the team can fully experimentally characterize values for the specific sensor. Even if these values shift greatly, which is not expected, the sensor has a wide enough range that it will be able to handle any change in shut-down conditions, corresponding to Functional Requirement 2. This sensor is also incredibly small and lightweight, so Functional Requirement 7 is satisfied. Lastly, this sensor also operates on the I^2C interface, needing the same 4 pins that the atmospheric breakout board needs, greatly simplifying the integration of this component.

3.2.2.4 Lightning Detector

The lightning detector chosen is the **SparkFun AS3935 Lightning Detector**[17]. This particular sensor uses RF centered around the 500kHz frequency to detect lightning strikes and estimate the distance to the storm head. Both the

Standard Operating Conditions and the shut-down condition for lightning state that any lightning detected will send WRAITH into safe mode. This sensor is capable of detecting lightning from as close as **1km** to as far as **40km**. 40km is far enough away that WRAITH will fully be in safe mode by the time the storm reached the deployment location. Additionally, this sensor operates on the SPI interface, so it needs the following pins: 5V of power, a ground, MOSI, MISO, CS, and SCK. This interface is widely used and SparkFun provides help with this sensor on the SPI interface, so integration with the microcontroller should be straight forward.

3.2.2.5 Precipitation Sensor

The type of precipitation sensor that WRAITH needs for the weather detection system is a simple one that reads if there is moisture present on the board. This deviates from the standard precipitation sensors used in hobbyist weather stations that are much larger, much more expensive, and measure amounts of precipitation as opposed to simply detecting moisture. These such precipitation sensors are beyond the scope of this project since the weather detection suite only cares if it's precipitating or not, not the amount it precipitates. Since the needed precipitation sensor is not widely used, there are not many options nor is there much information on others who have used them. Because of this, the team chose the **XLX Raindrop Sensor**[18] found on Amazon and preemptively bought a set of 6 to begin testing them. This sensor requires 4 pins: 5V of power, a ground, an analog output, and a digital output. It outputs an analog signal between **0 and 4095**. These values correspond to the amount of moisture detected on the board by detecting the amount of the board that is being shorted out. A value of 4095 means the board is completely dry, and the lower the value is, the more moisture is present on the board. The Standard Operating Conditions as well as the shut-down condition dictate that there should not be any precipitation while WRAITH is not in safe mode, so if the precipitation sensor drops from 4095 to any lower value, the weather detection system will command WRAITH to enter safe mode. This complies with Functional Requirement 2. If the values are detected as increasing back up to 4095, it's a good indication that the precipitation has stopped and the board is drying off, so WRAITH can resume nominal operations.

The team completed early testing with these precipitation sensors to ensure that they will work as needed. An Arduino Uno was used with a breadboard and open-source code from the Arduino website. First, the team verified that a dry board continuously outputted a value of 4095. Next, the team dripped some water from a water bottle onto the sensor and verified that the values dropped lower with more water present on the board. Lastly, the team sprinkled some day-old snow on the board and verified that this dropped the outputted analog value but not as much as the water did. With these results, seen below in Table 3, the team is confident that this sensor will work as needed to serve as a last line of defense for the weather detection system to determine if there is currently precipitation occurring or not.

Table 3. Preliminary Precipitation Sensor Test Results

Board Condition:	Analog Output:
Dry	4095
Damp	4076
Water Droplets	~ 1200
Day-Old Snow	~ 2200

3.2.3. Sensor Suite Logic

As previously mentioned, the external sensor suite will be driven by a microcontroller operating on a C++ based coding language. The particular language is an adaptation by Particle Industries which is designed to work specifically with their Internet of Things (IoT) devices. All of the code will be developed in the Particle Console which provides a simple and reliable interface between the user and the microcontroller device[13]. The specific functions of the microcontroller are, as previously mentioned, to monitor the state of each weather sensor, while also to receive overriding commands from an operator over the cellular interface. The high level logic for this function is outlined in Fig. 17.

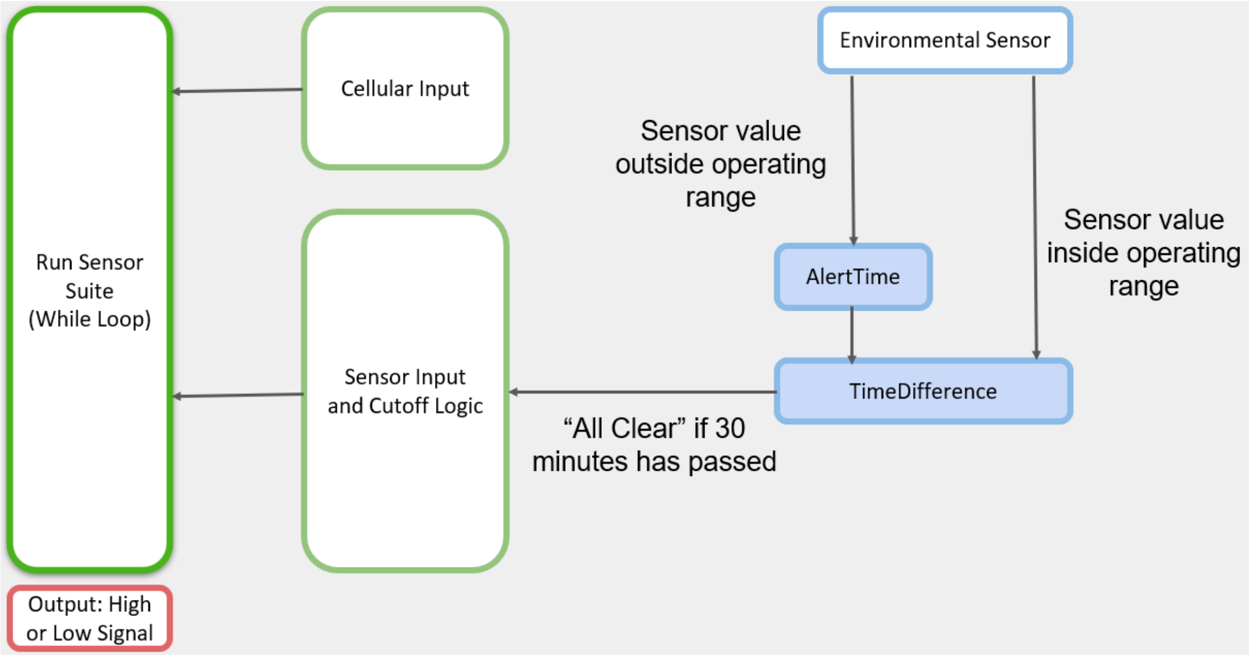


Figure 17. Environmental Software Architecture

Run Sensor Suite is the overarching architecture for the microcontroller which encompasses both the communication and processing. The nominal state of the sensor suite software is to remain in the processing loop since this loop reads the sensors, analyzes the output, and then relays the condition to the main processor. This loop will be running at approximately 1 Hz and will pass the state command to the main processor at the completion of each iteration. Each sensor will be read by its respective read function and the output of these sensors will then be compared to the shut down conditions. In the case that a measurement is outside the safe-mode conditions, the logic will write a close variable to indicate the necessary shutdown. It is important to note that this loop begins by checking the cellular input, thus any operator shut-down command which would then result in the same close variable. The purpose of this architecture is to ensure that the microcontroller relays the safe mode command until the operator gives the operational go-ahead, therefore it is necessary to assess this at the beginning of every loop. In this state, the system not only awaits the command from the operator, but also performs the nominal processes in order to ensure the system does not open to adverse weather following the go-ahead from the operator. Although these processes are being performed like the nominal phase, the outcome of this algorithm cannot override the operator's command. The microcontroller will receive the operator command over the LTE connection but the operator will upload the command by publishing an "event" on the Particle console[13]. The process of publishing an event allows the microcontroller to receive the command since the microcontroller will be subscribed to all events from the operator, thus any new event will be received by the microcontroller.

3.2.4. Conditions

The sensor suite will incorporate multiple levels of weather detection logic which dictates the impact a single sensor has on the safe mode command. Looking back at Fig. 17, inputs to the microcontroller are read and analyzed for all of the sensors, however the resulting commands from the microcontroller can differ depending on the sensor trigger. In order to minimize the likelihood of false-positives, the weather sensor software will differentiate between sensors that assist in predicting the weather versus sensors that provide the current conditions. As seen in Fig. 18, three sensors can trigger the safe mode signal independently while the remaining sensors are incorporated into a multiple sensor trigger. The single sensor trigger relies on the sensors that provide immediate feedback of the weather conditions, therefore indicating immediate risk to the system for a positive reading. As seen in the Fig. 18, the lightning detector, precipitation detector, and the anemometer are the sensors that can independently signal for safe mode. In short, lightning at any distance indicates a storm in the immediate proximity of the system, any measurement of precipitation indicates immediate risk to the system, and wind speeds above $8\frac{m}{s}$ indicates the possibility of dust in the air which can both obstruct the camera and cause harm to the system. The software logic behind these triggers is to immediately assign a ”High” value to the safe mode indicator and to bypass the loop in order to pass the indicator to the main processor as quickly as possible. These three sensors will be the first occurring processes in the nominal processing loop and will be followed by the multiple sensor trigger. It is worth noting that the cellular command will still be read prior to these three sensors.

As previously stated, the next three sensors will be analyzed in tandem as a combination trigger. These sensors are the barometer, relative humidity sensor, and the infrared thermometer since individually they do not indicate the immediate risk of adverse weather. However, in combination, they indicate a high likelihood of adverse conditions. The structure for the multiple sensor triggers will be to read each sensor individually and then sum the number of positively tripped sensors. If the sum of these sensors is greater than or equal to two this would signify an immediate risk to the WRAITH system, thus the safe mode indicator would be assigned a value of ”High”. If only zero or one of these multiple sensor triggers indicates a positive trigger, the system would continue with nominal functions.

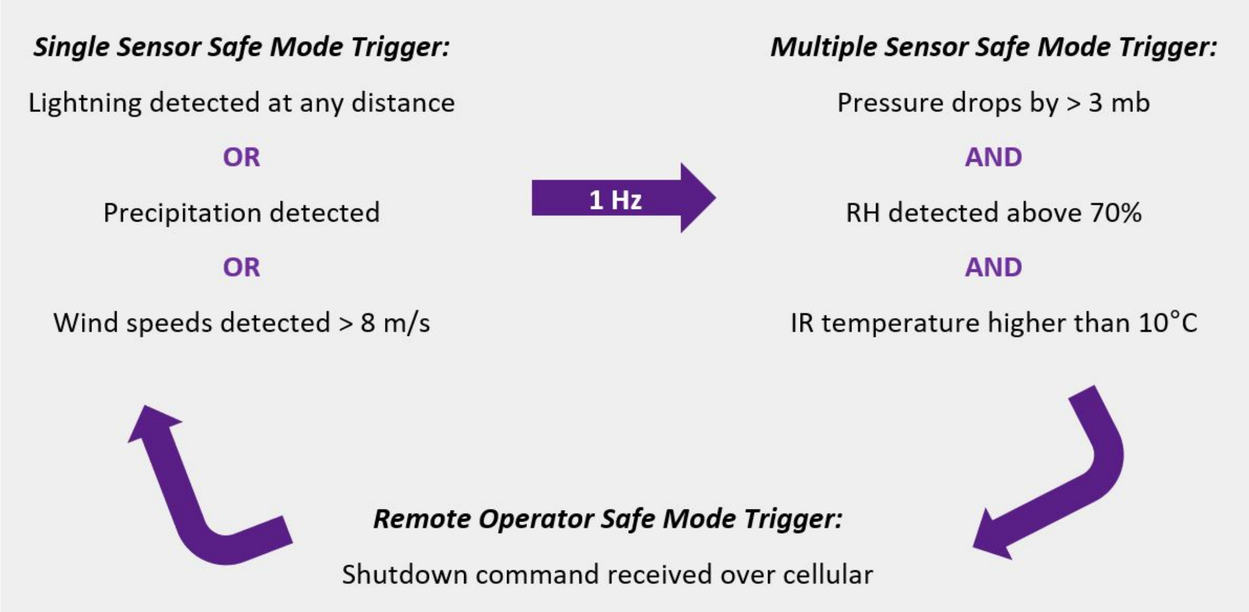


Figure 18. Environmental Software Closure Triggers

3.2.5. Microcontroller

For the weather detection sensor suite, a microcontroller is needed to run all the environmental software and communicate with the main UDOO processor on-board WRAITH. The specific microcontroller chosen is the **Particle E-Series Microcontroller**[14]. This microcontroller was chosen because it has 57 pins which is far more than the 15 pins needed to connect all the sensors as well as the umbilical to the main box and some extra pins as contingency. In addition, the Particle E Series is equipped with an LTE cellular module which gives the operator the capability of remotely shutting down the box to safe mode, in compliance with Functional Requirement 3. This functionality aligns with the design to include cellular communication capabilities as determined through a trade study comparing Cellular Communication, Pocket/Ham Radio Communication, and Satellite Communication. This entire study can be found in Appendix A.2. As previously mentioned, the microcontroller would read and interpret the data from each individual sensor, pass a system state command to the main processor, and also be capable of receiving shutdown commands from an off-site operator. The wiring diagram showing all the connections to the microcontroller is shown in Fig. 19.

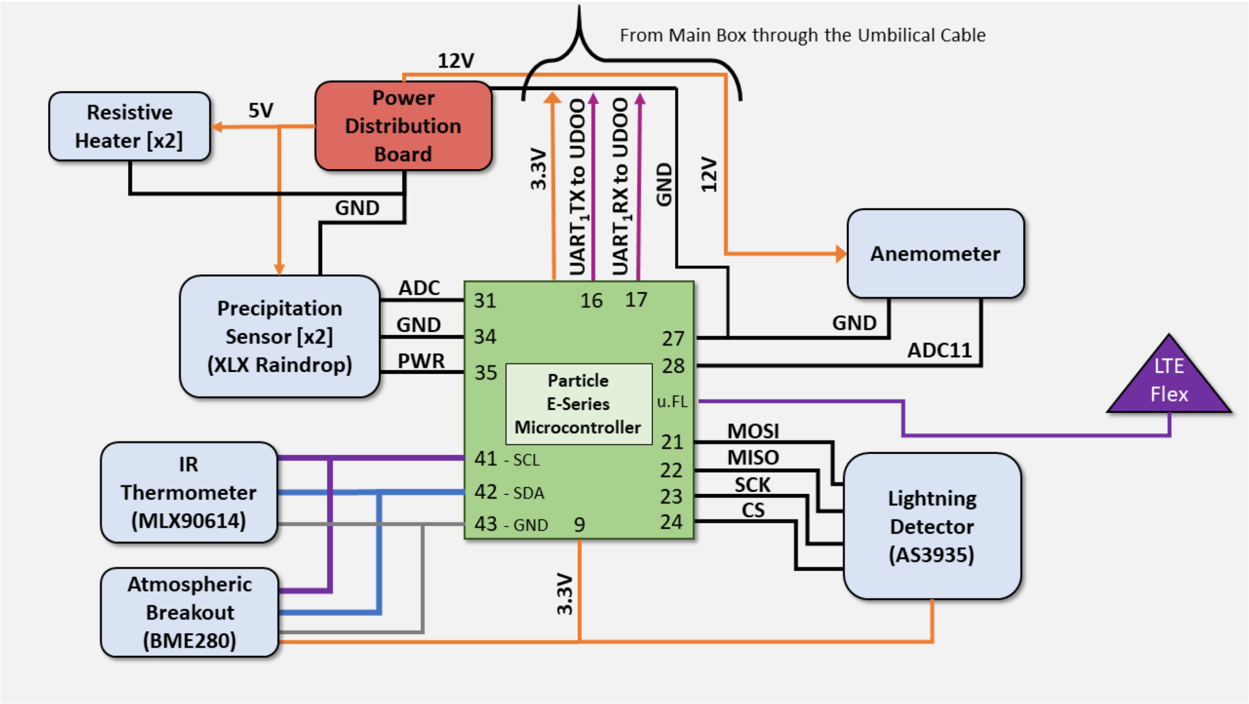
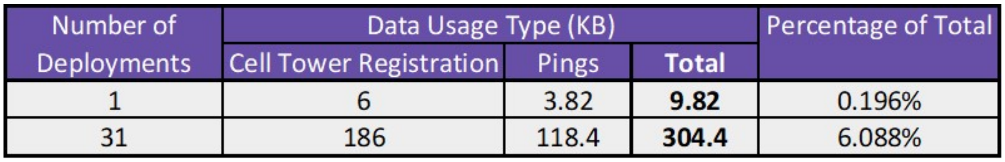


Figure 19. Weather Sensor Suite Wiring Diagram

The Particle E-Series microcontroller was chosen since the board incorporates many different communication interfaces and the amount of pins far exceeds the requirements for this subsystem. This fact allowed the team to select the most beneficial sensors since compatibility and communications was not an issue. Among the multiple interfaces provided in the E-Series microcontroller, the team will be utilizing the SPI, I^{2}C, and UART interfaces and LTE connectivity, in addition to general digital and analog inputs. This allows the team to simplify the coding process of the weather sensing suite subsystem since the manufacturers of many of the sensors provide basic coding scripts that operate on these interfacing pins. For example, the IR thermometer and the Atmospheric Breakout board will both be operating on the I^{2}C pins since this not only allows multiple sensors to send data over the same pins, but also will be built upon code provided from the manufacturer of these sensor breakout boards[16] [19]. In contrast, the lightning detector will be connected on the SPI interface since SparkFun, the manufacturer, strictly supports the SPI interface only for this particular sensor[17]. Since the E-Series microcontroller incorporates both of these interfaces, the team did not have to sacrifice sensor functionality or reliability when choosing the ultimate sensor selections.

The communication with the main UDOO X86 processor will be done through the UART receiver and transceiver pins on the E-Series. This was chosen since this opens serial communication between the devices. In between the two communication points will be a BOB-12009 Bi-Directional Level Shifter to alter the 3.3V RX and TX from the E-Series to the 1.8V that the UDOO requires. Although the adverse weather indicator from the E-Series is simply a ”0” or a ”1”, this method was chosen over a trivial digital high-low signal to provide the option for further iterations of the project to send more data between the processor and the microcontroller. In addition, the team chose the particle device for this reason since the device allows ”events” to be published to a cloud terminal over the LTE connectivity. The ”event” name covers any information that the device sends out to the off-site operator or even other devices. This provides the possibility that the WRAITH system can publish information regarding specific NORAD IDs to an off-site operator during a deployment since the information can be sent over serial from the processor to the microcontroller, and then over LTE from the microcontroller to the operator. It is important to note that both sending information from the processor to the microcontroller and sending information from the microcontroller to the operator are both beyond the scope of this project, but the hardware and routes are in place to allow this capability in the future.

As stated throughout this section, the final consideration for the selection of the microcontroller was the ability to communicate with an off-site operator over LTE connection. The Particle E-Series provides this capability through the aforementioned LTE cellular module that is incorporated into the device. The service for the E-Series is provide by AT&T and purchased through Particle. This particular plan allows the microcontroller to transmit up to 5MB of data for $3.79 which far exceeds the requirements of the system as seen by the data usage budget seen in Table 4. Due to these factors, the Particle E-Series was chosen as a viable option for the LTE transmission and the main microcontroller for the weather sensing suite.



Number of Deployments	Data Usage Type (KB)			Percentage of Total
	Cell Tower Registration	Pings	**Total**	
1	6	3.82	**9.82**	0.196%
31	186	118.4	**304.4**	6.088%

Table 4. Weather Sensor Suite Data Usage

3.3. Active Weather Protection

Dalton Turpen

3.3.1. Designs Considered

In order for WRAITH to complete its mission of a 12 hour autonomous deployment while simultaneously imaging and protecting itself from adverse weather an active weather protection system needed to be implemented. It was decided that there was to be an enclosure built that would protect WRAITH from adverse weather that could also be actuated open to allow for unobstructed viewing. To achieve this functionality, there were 5 designs considered for the active weather protection: Moon Roof, Lateral Garage, Canvas Convertible, Segmented Concentric Dome, and the Sliding Canopy.

3.3.1.1 Moon Roof

The moon roof design is an enclosed box with rigid walls and a sliding actuated lid that can both protect from adverse weather and actuate open to allow for viewing. All of the imaging hardware and actuation stage would be mounted in the box, allowing for nearly the entire system to be contained within one enclosure that could be moved by two people and transported in the trunk of a sedan.

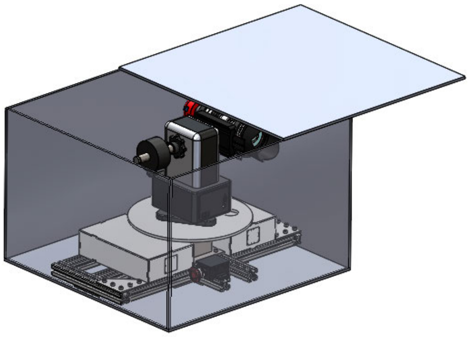


Figure 20. CAD Rendering of Moon Roof Concept.

3.3.1.2 Lateral Garage

The lateral garage design is very similar to the moon roof design with the exception of which part(s) slide. For the lateral garage 3 walls and the room actuate while the base plate and 4th wall remain static. This allows for protection from adverse weather and also actuated to allow viewing.

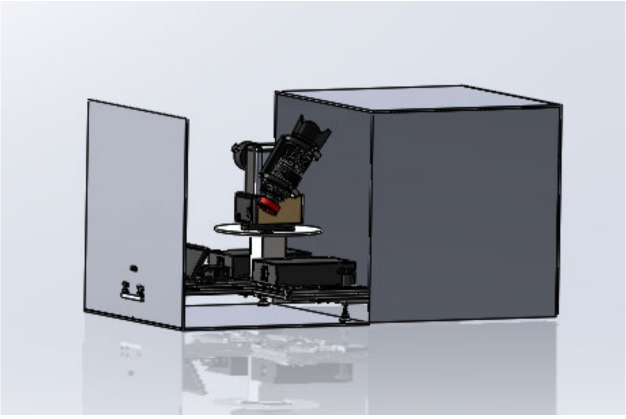


Figure 21. CAD Rendering of Lateral Garage Concept.

3.3.1.3 Canvas Convertible

The canvas convertible is a dome design with a rigid skeleton and a fabric canopy. This design works similarly to the sun shade on a baby stroller where the half hoop skeleton pieces all pivot about the same point allowing for the attached canvas to be actuated open and closed to allow for weather protection and viewing.

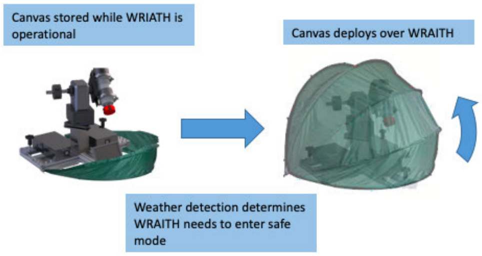


Figure 22. CAD Rendering of Canvas Convertible Concept.

3.3.1.4 Segmented Concentric Dome

The segmented concentric dome is a rigid bubble design that has pivoting segments that allow open and close actuation of the shell. This design is the most complex to manufacture but also has the capability to be the least restrictive of the field of view of the imaging hardware.

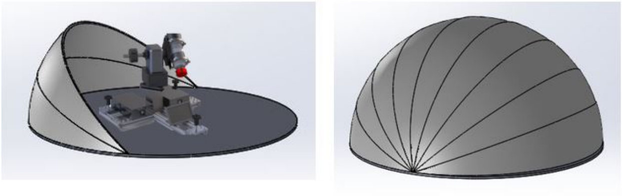


Figure 23. CAD Rendering of Segmented Concentric Dome Concept.

3.3.1.5 Sliding Canopy

The sliding canopy is the second fabric design and has an actuation system similar to an automatic pool cover. The fabric would be pulled taught between two sets of half-hoop shaped tracks and a pulley system would actuate the fabric over the hoops to allow for opening and closing of the system.

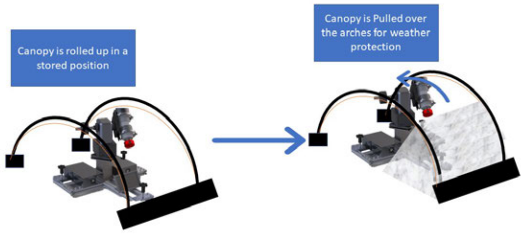


Figure 24. CAD Rendering of Segmented Concentric Dome Concept.

3.4. Active Weather Protection Summary

Dalton Turpen

Table 5 below serves as a brief summary of the benefits and risks involved in each of the design choices.

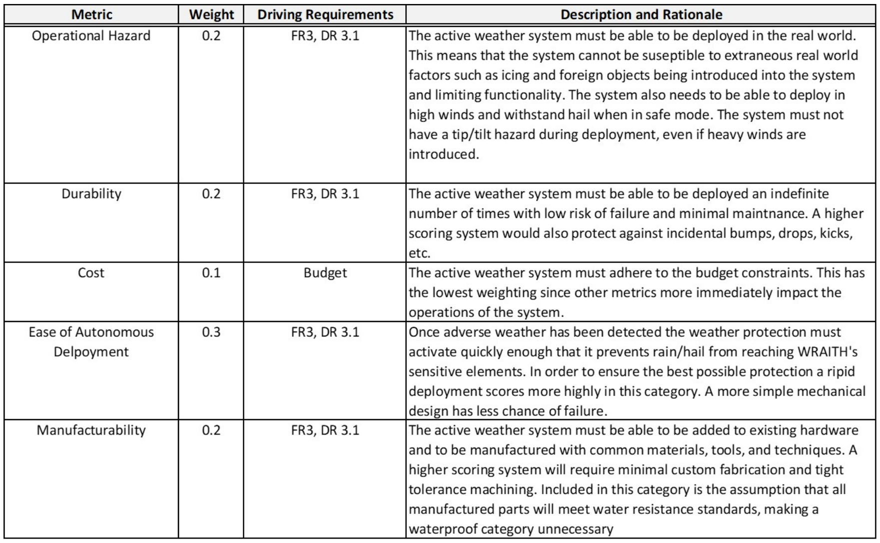
Table 5. CAD Rendering of Segmented Concentric Dome Concept.

Option	Pros	Cons
Moon Roof	• Uses commonly available material • Minimal R&D efforts to get to full functionality • Requires only one motor • Has only one moving part (not including motor) • Provides protective external shell • Very weather resilitant, least suseptible to operational hazards • Allows for easy storage and transport	• FOV Restrictions • Costly • Likely heavier than canvas and plastic options
Lateral Garage	• Uses commonly available material • Minimal R&D efforts to get to full functionality • Requires only one motor • Has only one moving part (not including motor) • Provides protective external shell • Very weather resilitant, least suseptible to operational hazards • Allows for easy storage and transport	• FOV Restrictions • Costly • Likely heavier than canvas and plastic options • More tip/tilt hazard than moon roof
Canvas Convertible	• When collapsed no FOV restrictions • Lightweight materials • Low cost	• Complex fabrication due to attachment to servo(s) • Questionable durability due to canvas • Difficult to store and transport • Unknowns for canvas/fiberglass interface and joints
Concentric Segmented Dome	• When collapsed no FOV restrictions • Most aesthetically pleasing • Most similar to commerical protection methods	• Highly complex fabrication due to segments • Waterproofing seams • Costly • Unknown joint geometeris, seam seals, servo
Sliding Canopy	• Simple mechanism • Lightweight materials • Low cost	• Possbile FOV restrictions • Questionable durability due to canvas • Complex fabrication due to rail system • Unknown rail geometry and servo

3.4.1. Active Weather Protection Trade Study

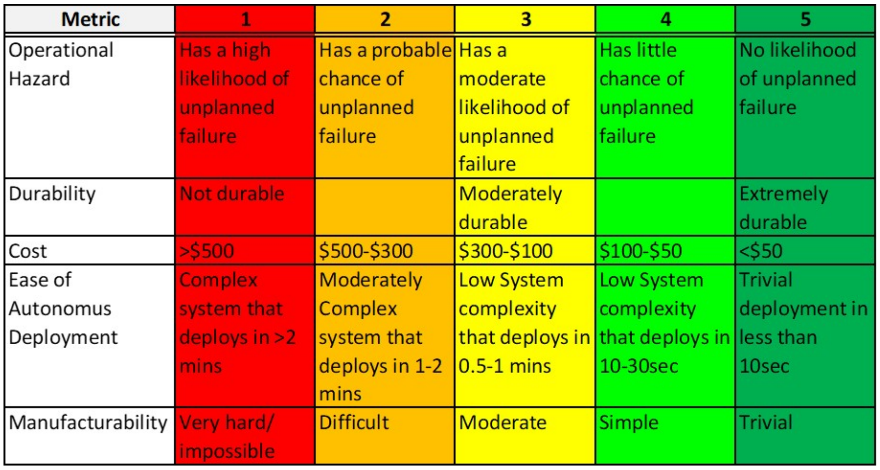
In order to make a decision on which active weather protection design to proceed with a trade study was conducted. The trade study description, weight breakdown, and scores were created and calculated.

Table 6. Active Weather Protection Trade Study Description



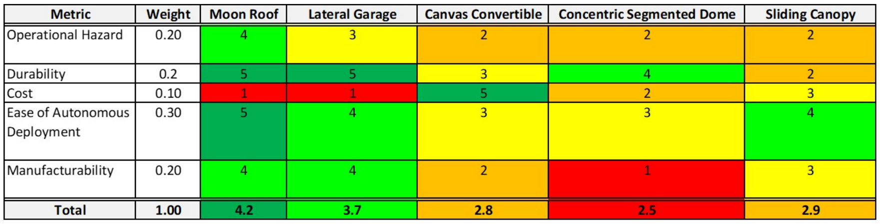
Metric	Weight	Driving Requirements	Description and Rationale
Operational Hazard	0.2	FR3, DR 3.1	The active weather system must be able to be deployed in the real world. This means that the system cannot be suseptible to extraneous real world factors such as icing and foreign objects being introduced into the system and limiting functionality. The system also needs to be able to deploy in high winds and withstand hail when in safe mode. The system must not have a tip/tilt hazard during deployment, even if heavy winds are introduced.
Durability	0.2	FR3, DR 3.1	The active weather system must be able to be deployed an indefinite number of times with low risk of failure and minimal maintnance. A higher scoring system would also protect against incidental bumps, drops, kicks, etc.
Cost	0.1	Budget	The active weather system must adhere to the budget constraints. This has the lowest weighting since other metrics more immediately impact the operations of the system.
Ease of Autonomous Delpoyment	0.3	FR3, DR 3.1	Once adverse weather has been detected the weather protection must activate quickly enough that it prevents rain/hail from reaching WRAITH's sensitive elements. In order to ensure the best possible protection a ripid deployment scores more highly in this category. A more simple mechanical design has less chance of failure.
Manufacturability	0.2	FR3, DR 3.1	The active weather system must be able to be added to existing hardware and to be manufactured with common materials, tools, and techniques. A higher scoring system will require minimal custom fabrication and tight tolerance machining. Included in this category is the assumption that all manufactured parts will meet water resistance standards, making a waterproof category unnecessary

Table 7. Active Weather Protection Trade Study Breakdown



Metric	1	2	3	4	5
Operational Hazard	Has a high likelihood of unplanned failure	Has a probable chance of unplanned failure	Has a moderate likelihood of unplanned failure	Has little chance of unplanned failure	No likelihood of unplanned failure
Durability	Not durable		Moderately durable		Extremely durable
Cost	>$500	$500-$300	$300-$100	$100-$50	<$50
Ease of Autonomus Deployment	Complex system that deploys in >2 mins	Moderately Complex system that deploys in 1-2 mins	Low System complexity that deploys in 0.5-1 mins	Low System complexity that deploys in 10-30sec	Trivial deployment in less than 10sec
Manufacturability	Very hard/ impossible	Difficult	Moderate	Simple	Trivial

Table 8. Active Weather Protection Trade Study



Metric	Weight	Moon Roof	Lateral Garage	Canvas Convertible	Concentric Segmented Dome	Sliding Canopy
Operational Hazard	0.20	4	3	2	2	2
Durability	0.2	5	5	3	4	2
Cost	0.10	1	1	5	2	3
Ease of Autonomous Deployment	0.30	5	4	3	3	4
Manufacturability	0.20	4	4	2	1	3
Total	**1.00**	**4.2**	**3.7**	**2.8**	**2.5**	**2.9**

As seen above in 8, the moon roof design scored higher than the other designs. Because of its ease of manufacturing and simplicity of actuation it was decided that it satisfied the requirements and was most likely to help WRAITH succeed in its mission.

3.5.   Passive Weather Protection

Austin Cyrus, Noah Crisler

3.5.1. Designs Considered

Once adverse weather has been detected, WRAITH must be able to physically protect itself. In order to ensure a redundant and weatherproof system, it was determined that both passive and active weather protection would be utilized in the WRAITH system. The passive protection system will ensure WRAITH can survive normal operating conditions, as well as any moisture that could accumulate while the active system deploys. The main purpose of the passive weather protection is to protect the more sensitive electronic components that will be mounted on the base of the main enclosure, as well as mitigate damages that may occur in the unlikely event that water accumulates in the bottom of the box. The methods of passive protection considered were multiple commercial off-the-shelf weatherproof boxes, a single commercial-of-the-shelf weatherproof box, canvas tents, and custom built enclosures. These methods are explained further below.

3.5.1.1 Single COTS Box

The single COTS box would entail one large enclosure, sized to contain all electronic components, excluding the actuation mount and camera. This enclosure would be made out of either aluminum, polycarbonate, or stainless steel. The single box would have multiple sections installed aftermarket for the battery, on-board processor, power distribution system, and any other electronic elements. The camera and actuation mount would not be passively protected in this design due to the necessity of their external mounting and frequent movement. Holes would be drilled into the side of the box to allow for wire connections and possibly heat dissipation. The single box design minimizes the number of holes needed to be drilled and consequently re-waterproofed. The downsides to the single COTS box are the need to design the system to fit inside a predetermined space and the chance that the size of the box would limit the operations of WRAITH. The single COTS box would cut down on both time and money spent on custom manufacturing of a rigid waterproof case.

Figure 25. Example of a Waterproof *polycase* COTS box

3.5.1.2 Three COTS Boxes

In contrast to the single box design where all the essential parts would be placed in one enclosure, this design places the battery pack, on-board processor, and the power distribution system all in separate cases spread around the base of the WRAITH platform. Spreading the boxes around the platform allows for better optimization of space, but would also increase the number of holes drilled for wiring connections, therefore increasing the number of containers needing to be waterproofed. The pre-made enclosures, again, would cut down on manufacturing cost and time while providing a durable passive waterproofing solution.

3.5.1.3 Lower Tent

The lower tent passive water protection system is a soft shell option considered for passive weather protection. It would feature a waterproof polyester canvas placed around a skeleton that surrounds the specified components. The structure and polyester canvas would provide passive shielding to everything except the camera, lens, and actuation mount of WRAITH. The skeleton structure would be built out of fiberglass tent poles, and the canvas would have to be pulled over the structure prior to deployment. The system would require custom manufacturing and assembly as a tent-like structure with the necessary form-factor is not commercially available. It would also limit quick accessibility to the inner components of WRAITH as the canvas would have to be removed each time. The canvas lower tent designs allow for a relatively inexpensive ($7.00/yd), durable, and customizable weather protection solution.

Figure 26. Concept of a lower-tent soft-shell passive weather protection design

3.5.1.4 Upper Tent

The upper tent passive weather protection design is the second of the two soft-shell designs. It consists of a custom fiberglass cage with a waterproof polyester canvas fit around it. The upper tent design differs from the lower tent design as it would provide passive weather protection to the entirety of WRAITH except the protruding lens. There would be a severe risk of inhibiting the actuation mount or the lens during the deployment, as the flexible canvas would be in close proximity to both moving parts and lens opening. That being said, the upper tent design is the only design that provides passive shielding to the actuation mount and camera. As with the previous tent design, the low cost of the canvas and fiber glass poles provides a relatively cheap, durable, customizable weather protection solution.

3.5.1.5 Lower Shield

The single, custom hard-shell enclosure would encompass the lower portion of the WRAITH system. The box would protect the non-moving components of WRAITH, like the batteries, CPU, GPS, etc. from damage. By enclosing the non-moving parts, it allows for added security, especially in terms of sudden inclement weather mitigation. The advantages of using a custom box lie in the ability to create a volumetric package that is advantageous for this application, as opposed to trying to find a volumetric package that the system can fit within, which may have a poor form-factor. This passive protection system would be extremely close in size and shape to the lower tent, except the canvas would be replaced with a rigid material. The shell material would be constructed using additive manufacturing, utilizing 1 cm thick ABS or polycarbonate plastic. The joints would be self-tapped with M6 screws and waterproofing joiner at all intersections to ensure a tight, waterproof fit.

Figure 27. Concept of a lower system hard-shell passive weather protection design

3.5.1.6 Passive Weather Protection Summary

Table 9 provides some of the costs and benefits of the previously proposed methods. Many of these costs and benefits directly correlate with metrics in the Passive Weather Protection Trade Study, and thus will be analyzed in further detail in a following section.

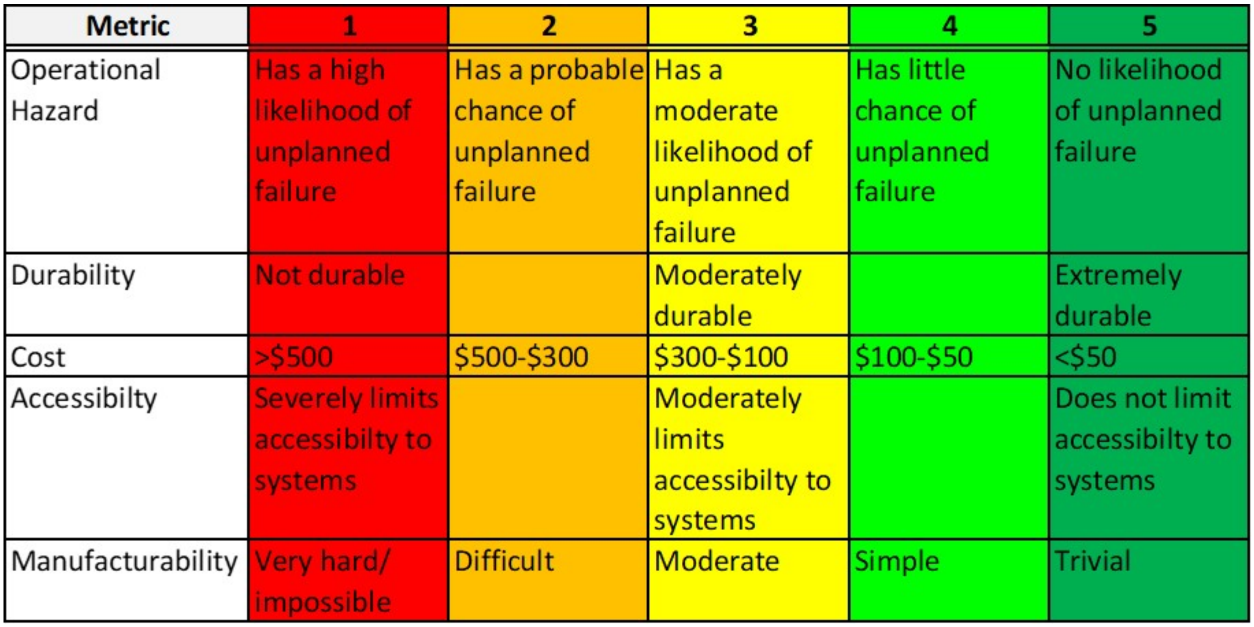
Table 9. Passive Weather Protection Method Pros and Cons

Option	Pros	Cons
Single COTS Box	• All crucial systems are in one area • May help with heating in low temps • Guarenteed water proof	• FOV Restrictions • Costly • Likely heavier than canvas and plastic options
3 COTS Box	• Allows for more freedom in placing systems • No custom parts needed • Guarenteed waterproof	• Will require lots of cable managment form box to box • More costly than the canvas options ~$200
Lower Tent	• Cheap • Allows for complete lower level protection • Allows for optimization of space	• Not as durable as COTS bpxes • Requires a custom made canvas
Upper Tent	• Allows for the most passive protection	• FOV Restrictions • Very hard to manufacture a working system • Requres a custom made canvas
Lower Shield	• Customized sheild allows for volumetric optimization • More durable than the canvas options • No need for cable management	• Would have to waterpoof manufactured parts • would limit accesbility to lower systems • Costly

3.5.2. Trade Studies

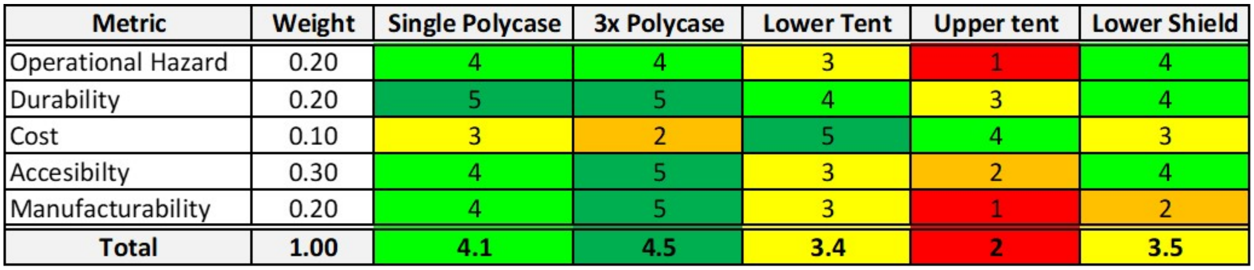
The trade studies conducted followed the rationale of providing the best possible protection for the sensitive elements of WRAITH while simultaneously allowing for maximum functionality of the system. In addition, it was important to have a durable system for multiple deployments in possibly adverse conditions. Finally, one of the most heavily weighted components of the trade study was derived from the 30 minute setup time; that coupled with the prototypic nature of this project and the mercurial state of the system meant that accessibility was one of the most important aspects of this project. The design effects considered were all related to the performance of the system as a whole, as well as, the design requirements handed down from the customer. Each of the designs of the passive weather protection were implemented in 3D designs to not interfere with any performance characteristics of the currently existing GHOST hardware, but also to incite an increase in performance in some of the system components in adverse weather. The only negative impacts identified that the passive protection would have would be an increase in manufacturing time, whether that be for waterproofing pass-throughs or for the actual protection system itself, and an increase in weight to the system. The methodology for assigning weights was straightforward and practical, and was concerned with operational hazards, durability, cost, accessibility, and manufacturability. The most important item considered was accessibility, as the changing of the design later on coupled with the need for easy access during testing provided a heavy weight for this category. The ability to maneuver and alter the interior components is critical in prototypes, and was absolutely imperative to address in this case. In addition, these systems have to be durable and have a low chance of failure, otherwise the continuous deployment aspect of this entire project would be nullified. Thus the operational hazards and durability categories were given the most weight. In addition, ease of manufacturing was weighted the same as the previous two categories, as an easily manufacturable system has a lower chance of failure, directly supplementing operational hazard and durability metrics. However, it received the same weight as those two because the resources saved by utilizing a more direct design were also considered when delegating importance. Finally, as with all projects, cost was considered, however it was weighted the lowest, as all of the options were relatively inexpensive compared to the overall budget of the project, and would not create an intense stress on the project if the cost was relatively high.

Table 10. Passive Weather Protection Metric Values



Metric	1	2	3	4	5
Operational Hazard	Has a high likelihood of unplanned failure	Has a probable chance of unplanned failure	Has a moderate likelihood of unplanned failure	Has little chance of unplanned failure	No likelihood of unplanned failure
Durability	Not durable		Moderately durable		Extremely durable
Cost	>$500	$500-$300	$300-$100	$100-$50	<$50
Accessibilty	Severely limits accessibilty to systems		Moderately limits accessibilty to systems		Does not limit accessibilty to systems
Manufacturability	Very hard/ impossible	Difficult	Moderate	Simple	Trivial

Table 11. Passive Weather Protection Trade Results



Metric	Weight	Single Polycase	3x Polycase	Lower Tent	Upper tent	Lower Shield
Operational Hazard	0.20	4	4	3	1	4
Durability	0.20	5	5	4	3	4
Cost	0.10	3	2	5	4	3
Accesibilty	0.30	4	5	3	2	4
Manufacturability	0.20	4	5	3	1	2
Total	**1.00**	**4.1**	**4.5**	**3.4**	**2**	**3.5**

The trade study provided results that were expected, as the ease and failure-mitigation aspects of the weighting really came into effect. The multiple COTS enclosure approach was deemed the best option for this project, which provided a clear answer of what path to pursue. There would be a weatherproof enclosure for every aforementioned piece of the system, and these enclosures would be linked via waterproof connectors. This would allow for flexible placement of different elements of the system and easy access during testing. The added benefit of a commercially available durable and tested product, a higher confidence against failure than an in-house part, and a conservation of resources that could be directed to the more arduous facets of this project made this decision most appealing. There would be a need for protection on the processor and power distribution boards, and a possibility of a need for battery protection. There would also be a need to protect the microprocessor in the Environmental Sensing Suite, which brought the definitive total up to three boxes, as it was determined at this point chronologically to use weather-resistant batteries.

The baseline design selected eventually settled on three different enclosures from Polycase, with the larger size being used to house the processor and the smaller size being used to house the microprocessor and power distribution board, separately. In terms of pass-throughs for wires, it was deemed appropriate that gasketed vents from the same manufacturer would be included as additions on each of the enclosures. These vents would work in tandem to act as pass-throughs for the wiring as well as heat-mitigation vents to keep all of the components at an acceptable temperature. The dimensions of the enclosures and wiring between them can be located in the figure below, to avoid confusion. These boxes will be closed and sealed before deployment and then fixtured to the baseplate with velcro, providing an easy, reliable, and non-permanent way to attach the enclosures to the active weather protection enclosure during testing. The velcro is used only to keep the boxes in place and allow easy removal and installation of these boxes during the development phase of the project. Later, if desired, the passive enclosures could be attached to flanges and then permanently fixtured to the baseplate of the active weather protection enclosure.

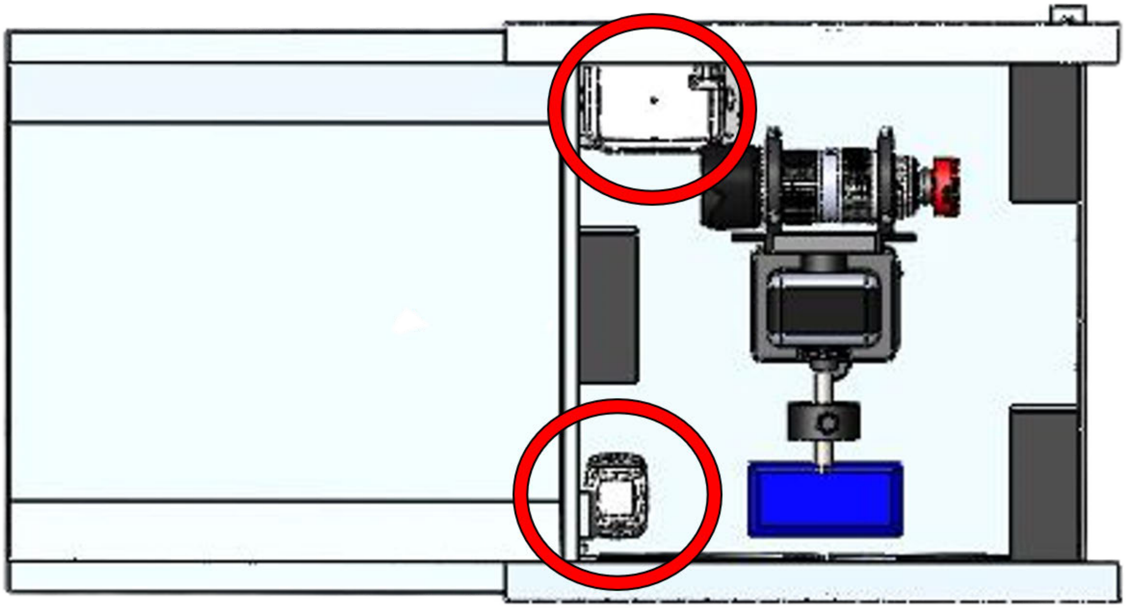


Figure 28. Diagram of Passive Weather Protection

3.5.3. Sensor Suite Enclosure

3.6. Requirements Flowdown

Jared Bartels

After definition of the functional requirements, design requirements were created as a further breakdown of what the system needed in order to achieve success. Note that all functional and design requirements are paired with their motivation for creation, and a brief, rough outline of how they will be verified. Further description of verification and validation can be viewed later in its own section of the report.

3.7. Breakdown to Design Requirements

FR 1: The system shall schedule predicted locations and visibility windows for objects in LEO, MEO, GEO, and HEO.

Motivation: WRAITH must be capable of scheduling it's own visibility windows for all of the specified space objects and this requires on-board orbit propagation. For space object reentry purposes and for tracking requirements related to WRAITH, HEO is used to describe orbits with an orbit perigee below 300 km and an apogee between 300 km and 30,000 km.

Verification: Test - WRAITH will be deployed, given input TLE's, and prompted to determine a visibility schedule. The success of the scheduling process will be measured by a human visually attempting to identify satellites at the scheduled times and locations from the WRAITH schedule.

DR 1.1: The system shall accept a series of satellite catalog numbers and TLE's as inputs from the user.

Motivation: The initial knowledge the user will have is the satellite ID's of the objects of interest which can be converted to TLEs offline and uploaded to WRAITH. Thus, WRAITH must be capable of retrieving the TLEs for these objects and using them as the only input.

Verification: Test - WRAITH will be given a series of satellite ID's and TLEs. If WRAITH accepts these and is able to create a schedule, then the requirement is satisfied.

DR 1.2: The system shall compute predicted observation windows for each input object based on input state information, environmental information, and system location.

Motivation: All factors that could influence the quality of a visibility window need to be considered in the on-board scheduler otherwise WRAITH may not efficiently capture all necessary observations with sufficient quality.

Verification: Test - WRAITH will determine its own location and environmental data and produce an observation schedule for itself. If a human can visually find the space objects at their scheduled times and locations then the requirement is satisfied.

DR 1.3: The system shall create an observation schedule that ensures at least one orbit state update is made per deployment for each input object.

Motivation: The customer wants to ensure that each input object is observed at least once before any object is observed twice.

Verification: Test - A series of satellite IDs and TLEs will be inputted to WRAITH and an operator will observe whether or not WRAITH schedules at least one viewing window for each.

DR 1.3.1: The system shall be capable of creating a new schedule if a visibility window is missed.

Motivation: It is common to fail to image a satellite for a variety of reasons. If a satellite is missed, WRAITH must be able to reorder the schedule to prioritize that object later in the deployment.

Verification: Demonstration - WRAITH will be made to create an initial schedule and be prompted to recalculate a new schedule with different priority values based off of a missed object.

FR 2: The system shall function in standard operating conditions with no human intervention for 12 hours.

Motivation: The customer would like to leave the system outside overnight without any human interaction and be able to pick it up in the morning.

Verification: Test/Inspection - WRAITH will be deployed, given its inputs, and left to run for 12 hours with no human interaction. Upon pickup, orbit outputs and image archive review will determine autonomous success or failure as well as survival of hardware components and their original functionality.

DR 2.1: The system shall operate in conditions defined by Appendix B with no impact to orbit determination capabilities.

Motivation: Since adverse weather is an important part of this project, it is important to define the standard operating conditions in which the system must operate nominally.

Verification: Test - The system will be deployed in a variety of environmental conditions that lie within the *Standard Operating Conditions* of Appendix B and sensor performance will be measured through a variety of tests.

DR 2.2: The system shall contain a battery capable of supplying operational power levels for 12 hours without recharging.

Motivation: During the 12 hours deployment the system must be capable of operating continuously which requires power draw.

Verification: Test - The battery will be subjected to an operational loading and the time to drain the battery will be measured as well as voltage versus time.

DR 2.2.1: The system's battery shall be capable of recharging from a 120V, 60 Hz power source, drawing under 20A, for 12 hours.

Motivation: The battery must be capable of charging from a standard wall outlet for ease of use.

Verification: Demonstration - The battery will be charged from a standard wall outlet.

FR 3: The system shall autonomously enter and exit a safe mode to protect itself from adverse weather.

Motivation: One of the customer's expectations is that the system will be able to operate for 12 hours, which requires protecting the system from potential adverse weather during that time.

Verification: Test - The system will be tested under simulated weather conditions to determine if the system can autonomously enter and exit safe mode.

DR 3.1: The system shall have active weather shielding, in conjunction with passive protection, to protect sensitive components from adverse weather defined by conditions worse than the Standard Operating Conditions defined in Appendix B.

Motivation: To protect the system from adverse weather, the system will require active weather protection to supplement the passive protection in the event of more serious conditions.

Verification: Test - The weather system will be tested under controlled conditions on its ability to be water resistant, to protect against light debris caused by wind, and maintain operable relative humidities.

DR 3.2: The system shall accept a safety override from the operator to activate and deactivate the weather protection.

Motivation: The system will be capable of autonomously activating and deactivating the weather protection through the weather sensor suite, but to mitigate the risks to the system an operator will have the ability to manually put the system into a safe mode.

Verification: Demonstration - The operator override will be initiated while in Standard Operating Conditions to see if the active protection deploys properly.

FR 4: The system shall autonomously point and track objects in LEO, MEO, GEO, and HEO.

Motivation: The purpose of the previous system, GHOST, was to track objects in LEO, MEO, and GEO given the rising number of satellites and space debris currently orbiting Earth. WRAITH has been expanded to include HEO orbits and to operate more autonomously.

Verification: Test - WRAITH will be deployed in the field and directed to point towards and track objects bounded within each orbit type.

DR 4.1: The system shall have a pointing accuracy within 4 arcseconds.

Motivation: This system is tracking objects in space in varying orbits, as such the pointing accuracy needs to be high in order to obtain the desired data.

Verification: Test - The system will be directed to point towards a well known object in the night sky, such as the North Star, and the accuracy of the image will be determined.

DR 4.2: The system shall slew at a rate of 2 deg/s.

Motivation: Objects passing perigee in HEO are moving quickly. Due to this, a higher slew rate is required in order to take tracking images of these objects. The value of 2 deg/s was chosen based on one of the example HEO bodies provided: satellite ID of 9927 and traveling at 0.807822 deg/s at its fastest point. Comparatively, the ISS moves at 0.0644 deg/s at its fastest point. These values were calculated using the perigee altitude and semi-major axis to obtain the velocity in km/s, which was then converted to a deg/s value.

Verification: Test - In a laboratory setting, WRAITH will be commanded to rotate a certain distance while recording the duration to determine the angular speed. This, when compared with a created control model for the actuation system, will prove success of the slew criteria.

DR 4.2.1: The system shall contain an on-board control algorithm to actuate the imaging system.

Motivation: A control algorithm is necessary to avoid overshooting or undershooting the target.

Verification: Demonstration - The actuation system shall be directed to point at a given target to successfully move the imaging system to aim at the target.

DR 4.3: The pointing and tracking sub-system shall interface with the scheduler to receive commands.

Motivation: To point at the correct target window, the actuators that control pointing will require input from the scheduler. This is also necessary when a pass is missed to re-orient the imaging system.

Verification: Test - This criteria will be satisfied when the scheduler successfully commands the actuators to point the imaging equipment at a designated target.

FR 5: The system shall image objects with apparent magnitude of less than 10 and process images within 10 seconds.

Motivation: WRAITH must be able to image the objects in the sky once it is pointed correctly at them. These images are the basis for the image processing which then feeds towards the goal of orbit determination.

Verification: Test - WRAITH will be fed correct and incorrect images to showcase abilities in measuring quality and look for missing space objects. Additionally, WRAITH will be tested in the field.

DR 5.1: The system shall provide ≥ 6 angular measurements in the inertial frame from a single orbit visibility window.

Motivation: WRAITH needs at least 6 angular measurements for the OD software to work correctly with no under-determined matrices. These can be gathered in pairs through three measurements that each have a start and end point to the streak line.

Verification: Test - The scheduler will be tested for creating a minimum of 3 unique captures for each space object pass. The image processing will be tested based off these 3 captures to ensure 6 measurements can be gathered.

DR 5.2: The system shall maintain a clock drift less than 5 milliseconds when compared with GPS time.

Motivation: Timing is key when orbital determination is the topic of interest. GHOST struggled to produce precise time latency results, which is necessary to increase the fidelity of an SSA tracker.

Verification: Test - In order to determine the camera latency, two tests can be conducted. One test is where the imaging system is set to a short exposure time (relative to monitor refresh rate) and images a screen displaying a fast-refreshing time. The second test is where the images are timestamped with GPS time from the on-board GPS receiver.

DR 5.3: The system shall process captured images and screen for quality and missing space objects.

Motivation: Quality images that include both the start and end of a streak during a single capture are necessary for the software to identify start and end points. Furthermore, if the space object is not present, the system needs to recognize this, notify the scheduler, and take action to locate that space object.

Verification: Test - WRAITH will be given quality images and bad images to determine if it can differentiate between them. Additionally, WRAITH will be given images with and without space object streaks present to determine if the space object is off of its predicated course.

DR 5.3.1: The system shall be capable of identifying and rejecting images that cannot be processed for boresight or space object inertial position.

Motivation: The system needs quality images for processing and orbital determination.

Verification: Test - WRAITH will be given good and bad images to determine if proper distinctions can be made by the system.

DR 5.3.2: The system shall be capable of identifying missing space objects within captured images and reporting that information to the scheduler.

Motivation: As part of the customer requirements, WRAITH must scan and locate missing space objects if they are not located along their predicted orbit.

Verification: Test - WRAITH will be given bad data (data without streaks or with only partial streak lines present) to showcase if the system can recognize, signal an alert, and search for the missing space object.

DR 5.4: The system shall be capable of tracking at least 6 resident space objects per hour subject to visibility constraints.

Motivation: The customer has a desired level of capability as far as the amount of objects that can be tracked per system deployed. The fewer satellites that can be tracked the more WRAITH systems must be put into the global network.

Verification: Test - The system will be deployed in a situation where 6 satellites must be tracked in an hour, if each satellite is given an updated TLE then the requirement is met.

FR 6: The system shall create and save an orbit estimate for each object imaged within five minutes of the end of the visibility window.

Motivation: The purpose of this system is to provide orbital estimates to the user after each deployment. The customer imagines WRAITH operating as a network of systems, and consistent orbit state updates would be necessary to enable fresh data is being iterated upon.

Verification: Test - The image processing and orbit estimation software will be ran on-board and timed in under five minutes.

DR 6.1: The system shall have knowledge of its own location in latitude, longitude, and altitude to within 10 meters.

Motivation: The system must know where the azimuth and elevation measurements were taken, otherwise the orbit determination will be wrong.

Verification: Test - The location reading from the on-board computer will be compared to other GPS units, like cell-phones and other hand-held devices, for accuracy.

DR 6.2: The system shall save the new orbit states, the data to construct those orbit state, as well as comparisons to previous orbit estimates for each tracked object to the on-board memory.

Motivation: The orbit estimates must be saved in the on-board memory in a format that can be read by the operator once the deployment is complete. Additionally, the customer and team agreed upon release of all orbit state construction data in the output.

Verification: Test/Demonstration - The orbit estimates will be output and read by an operator after the test deployment and compared with current orbit element data online. Additionally, it will be demonstrated that the data used to construct the orbit state on the system has been properly stored in the on-board memory.

DR 6.3: The system shall be capable of converting six sets of angular measurements into an orbit estimate within 4 minutes.

Motivation: The orbit estimates must be made no later than five minutes after the end of the pass, however with image processing taking up a small portion of time, a buffer was added in.

Verification: Test - Orbit estimates will be time-tagged and compared to the time of the end of the observation window.

FR 7: The system shall be deployed in 30 minutes and broken down in 30 minutes by two operators.

Motivation: A major requirement by the customer is to reduce deployment and tear down time of the system.

Verification: Test/Inspection - The operators will inspect to ensure proper size and weight while also running the operation process to keep under the allotted time.

DR 7.1: The system shall be palletized and contained within a 70 by 70 by 70 centimeter cube.

Motivation: To ease maneuverability, the customer desires a palletized rendition of the system.

Verification: Inspection

DR 7.2: Any system component shall have a mass less than or equal to 45.35 kilograms.

Motivation: The system is to be placed in the field by two operators who, to maintain the time constraint, should be able to lift the system with ease. OSHA standards typical cite 50lbs (22.68kg) per person for lifting. The system will likely be comprised of the batteries, sensor suite, and the main weather protection box and imaging hardware that will all be plugged together separately during the deployment time.

Verification: Test - The system segments (batteries, sensor suite, and main box with imaging hardware) will be weighed by a scale.

DR 7.3: The system shall be set up and taken down in accordance to the process document titled: *WRAITH System Operation Manual*.

Motivation: To quicken the process, a document outlining the operation of the system in the field may be of use.

Verification: Test - The setup and tear down process will be timed separately through their laid out process in the *WRAITH System Operation Manual*.

3.8. Overall Baseline Design

Julian Jurkoic

Below is a summary of all the conceptual design choices made in this section, thus making up the overall baseline design.

1. An on-board weather sensor suite will detect any adverse weather conditions which may affect WRAITH, updating the main processor with weather data on a 1 Hz loop.
2. A series of COTS boxes will be used to passively weather proof the system during operation and in the event of weather detection failure.
3. A Moon Roof design will be used to actively protect the WRAITH system from adverse weather conditions.
4. For communication purposes, the WRAITH system will utilize an existing cellular network.
5. The scheduler will be updated to a live system capable of creating new visibility windows if a target isn't captured.
6. The GHOST orbital determination system (linear batch filter) will be updated to be operable without an internet connection using an on-board star catalog.
7. The GHOST image processing system will continue to be used.
8. The GHOST imaging hardware (DSLR + Lens) will continue to be used.
9. The GHOST actuation hardware (iOptron) will continue to be used.
10. The system will use a general purpose Linux based computing board.

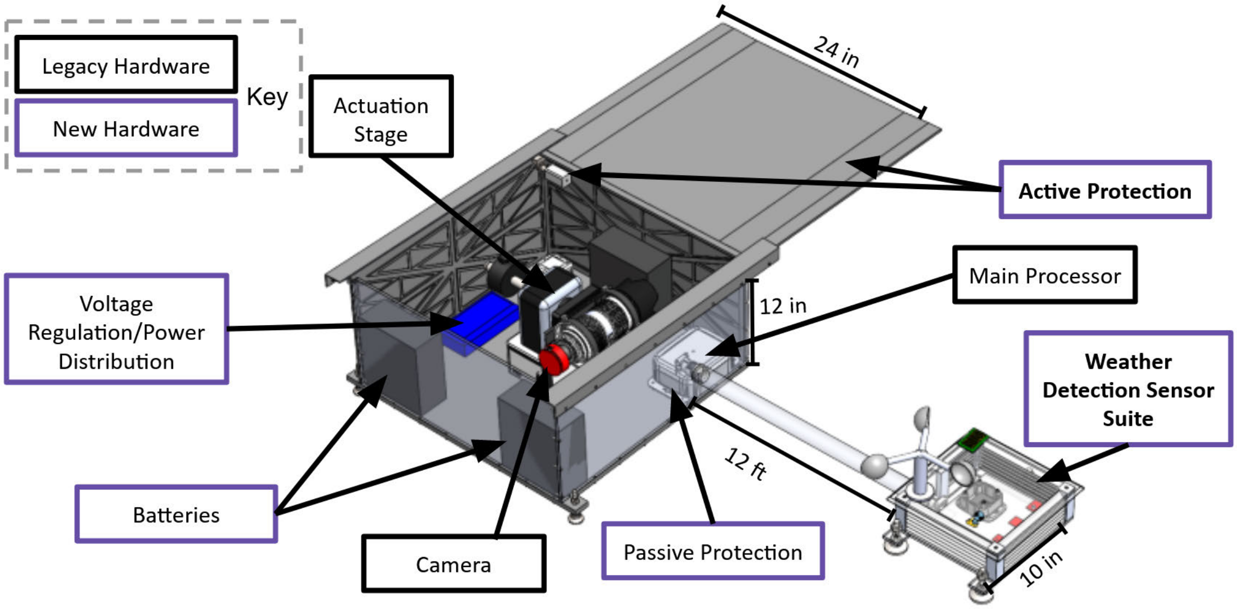


Figure 29. Overall Baseline Design

The final product is a roughly 110 lb aluminum box housing a camera mounted on an actuation stage. This box is able to open and close depending on the weather which will be monitored with an auxiliary sensor box. The camera takes long exposure images of space objects in their expected paths determined by an on-board scheduler. These images are then processed, filtered, and data is returned about the right ascension and declination of a space object at certain times. This data is then passed to orbit determination software which returns an orbit state for the space object and stores it in the on board storage.

4. Manufacturing

4.1. Software

For every piece of code being developed for this project, Github was used to manage version control and allow the team to be more confident that their software wouldn't be lost to a mistake. The general infrastructure that was followed to develop code was to create a new personal branch off of the master branch for each new piece of code to be written. Once new code had been written and tested on the individual branch, a pull request could be created to merge the new code into the master branch. All pull requests had to be reviewed and approved by another member of the team before they could be merged to the master branch. Github was also installed on the on-board UDOO processor such that hardware commanding code could be tested and so that code didn't have to be transferred via external storage and instead it could be transferred simply with an internet connection.

4.1.1. Hardware Commanding

4.1.1.1 Stage Commanding

Cal Van Doren

The software for commanding the stage is written in bash script while the commands themselves are ASCII commands that are passed directly to the serial port on the UDOO where the stage is connected. Note that these scripts are written specifically to operate with the serial port naming conventions on the UDOO and the software will not work effectively on a non-linux operating system. The scripts that were developed, that can be found in WRAITH/hardware_cmd/stage, for commanding the stage are as follows:

1. **Calibrate.sh** - Will take input right ascension and declination points and tell the stage that those are the current coordinates that it is pointing at (Note that this script would have been updated for an azimuth/elevation commanding scheme)
2. **SetHome.sh** - This command takes whatever the current pointing of the stage is and sets that as the home position
3. **SlewForCalibration.sh** - This command sets the slewing rate of the stage and slews it for an absolute time that ends with the camera pointing straight up if it began in the stowed position parallel to the ground.
4. **SlewHome.sh** - This command tells the stage to slew home immediately
5. **SlewRaDec.sh** - This command tells the stage to slew to the most recently defined right ascension and declination
6. **SlewSidereal.sh** - This command begins the sidereal slewing for imaging
7. **StopSlewSidereal.sh** - This command ends the sidereal slewing for imaging

Note that a full list of the ASCII commands that can be used with the stage are included in Appendix D.

4.1.1.2 Roof Motor Commanding

Alex Kirk

The software for roof motor commanding is split between a python script and arduino sketch. The motor is physically controlled by the arduino half of the UDOO processor and corresponding sketch, while logic is sent to the arduino by the python script running on the UDOO. The scripts can be found in WRAITH/physical_hardware/Motor_Control/Motor_final.

1. **MotorFunctions.py** - This script contains the open and close function needed to send commands to arduino. The functions are called by either a user, or another script and send a character to the arduino corresponding to the lid response needed.
2. **Motor_final.ino** - This sketch constantly runs on the arduino half of the processor using an embedded serial port connection. This serial connection is generally called ”ACM0” but in testing would occasionally change without reasonable explanation. This sketch receives the character sent by the python script and sends the physical command to the motor controller. Status is also constantly sent back to python, where numbers 0-3 represent either open, close, opening, or closing. When the project was stopped, functionality had just been added to incorporate feedback from the limit switches. This functionality must be tested and debugged with the hardware before any full system testing could be accomplished.

4.1.1.3 Camera Commanding

Trevor Willson

The software for the camera commanding was built as a customized script that utilizes a python binding for the ZWO C++ library that was produced by Steve Marple[20]. The scripts developed for the project allow for the system to initially connect to the camera module, to determine a gain value for a pre-determined exposure time, and to initiate an image capture sequence. All of these functions are defined in the same ImageCapture.py script. The script can be found at WRAITH/hardware_cmd/camera/zwoasi/.

1. **ImageCapture.py** - This script possesses the functions: imcapture, AutoGain, and CamCalibrate which respectively capture images on command, determine a gain value for a pre-set exposure length, and initiliaze the system-to-camera interface upon the first call to the camera. These functions can be called by either a user or through another script to operate the camera.
2. **ImageTestScript.py** - This script calls the three functions from ImageCapture.py and runs them in the order of CamCalibrate, AutoGain, and imcapture which is the expected order for the system to operate nominally. The purpose of this script was to check the feasibility of the autogain function by monitoring final values with preset exposure and gains values.

4.1.2. Hardware scheduler

Maya West

At the time of halting progress on the software, the team was in the midst of creating two scripts that would mesh the scheduler with the image capturing hardware. The basic functionality of these scripts was to monitor a file location for the creation of a schedule: a sequence of objects and their locations at specific times to be imaged. Upon creation of this schedule these functions would work together to build a job sequence for the hardware to follow. The file location was monitored using a python package, ”watchdog”. The need for constant monitoring as opposed to one initialization comes from the obligation to update the scheduler to search for missing objects. It is important to note that while these functions were written, we were not able to test that they worked properly.

- **SchedHW_Interface.py** - Monitors for the creation of a .cmd file in the ”commands” file inside the scheduler folder that contains the times and positions for imaging certain objects. It also includes the image sequence ID and NORAD ID for each object. The script parses through the file and creates a list of times for each image sequence to send to the following function. These times correspond to the functions listed in Section a. Each of these lists will be sent to the following function a minute before the first time listed in each sequence. This script can be found at WRAITH/scheduler.
- **hw_auto.py** - The various times for each hardware command type are received and lined up in a job. At the time of action for each hardware type the command is sent to the appropriate hardware commanding script. These scripts are defined in the above section. This script can be found at WRAITH/hardware_cmd.

4.2. Hardware

4.2.1. Main Enclosure

Austin Cyrus, Dalton Turpen

The moon roof design was manufactured out of 6061 T6 Aluminum and was predominantly CNC milled. The walls and lid were machined from 1/4” plate and the base was machined from 1/2” plate. They are fastened together with 6-32 cap head socket screws with a 3” spacing along the edges of the plates. The roof is sloped front to back (1/4” of drop per linear foot) and contoured to make sure that precipitation is not allowed to pool. To reduce the mass of the system as much as possible, mass relief cutouts were made to all four walls and the base plate, however not to the roof. These mass relief cuts were made in a way that left the pieces structurally compliant while still a significant amount lighter.

A flange was fabricated to cover the roof sliders, preventing precipitation from entering the sliders vertically. The sliders were COTS, and attached to the main enclosure via a support bar that was screwed into the side walls. The support beam was machined from a stock 60601-T6 bar. The sliders sat on top of these supports and in turn were attached via 6-32 cap screws to roof. The flanges were also contoured so that incident precipitation is forced off of the roof. While the lid is both open and closed, these flanges ensure that the slider mechanisms remain clear of foreign objects and precipitation, greatly reducing the likelihood of actuation failure. The flange was machined from pre-folded, stock 6061-T6 aluminum.

All of the components of this enclosure required only common machining techniques, moved from SolidWorks to SolidCAM to the CNC mills themselves. The manufacturing of this enclosure marked the start of component manufacturing and installation

The entire electrical system of the box was manufactured in-house, with the components of the system being COTS parts. The batteries, voltage conditioner, PDB, processor, limit switches, GPS, antenna, motor, and motor controllers were all COTS. These were all routed with custom-manufactured wire harnesses. These were installed from component to component with larger components requiring 14 AWG wire, while smaller components could tolerate in some cases down to 24 AWG. These harnesses were first prototyped while the enclosure was being completed, were installed as prototypes to test functionality, then redone as final installs. The wires were attached to jump points on boards with lead-free solder, ensuring a safe and stable connection. Every part of the custom wiring harness was fixtured when possible to the frame via zip ties/zip mounts. In addition, the central vector board was customized from a stock part, and was manufactured by shearing it to fit into its enclosure, then soldering the necessary wires into place. Sensitive electrical components were then placed in modified COTS containers that had vents placed in them as cable glands.

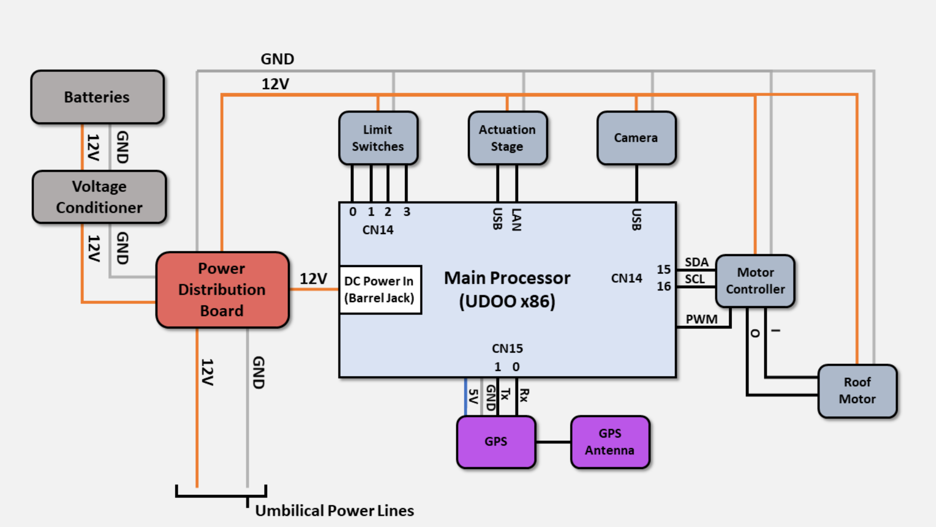


Figure 30. Main Box Wiring Diagram

The wiring was comprised of various gauges of stranded copper wire, design dependent on expected max current load.

4.2.2. Sensor Suite Enclosure

Dalton Turpen

The sensor suite enclosure was custom fabricated to replicate the general design of most commercially available Stevenson Screens. These screens are designed to allow ambient airflow through the enclosure while still providing precipitation protection for the electronics within. This enclosure was built on a 1/4” 6061 Aluminum base plate. The base-plate was hand milled to size. Three threaded bushings were press-fit to allow for leveling feet to be installed. Five 6061 Aluminum posts were CNC milled to allow for polycarbonate slats to be installed as components of the Stevenson Screen. The roof was hand milled from polycarbonate plastic so that it could be fastened to the 5 posts with machine screws, a hole was drilled into the roof to allow for the IR sensor to protrude through the material so there was no attenuation concern.

4.2.3. Sensor Suite Wiring

Jared Bartels

The environment wiring was done as shown in Fig. 19. The only the thing the team had left to add into the design and manufacture was switches for the resistive heaters that would have allowed for the Particle E-Series to turn them on or off based off temperature readings. Soldering directly to sensors as well as using header pins with the E-Series was utilized. The wire used for most of the suite is 20-22 AWG wire. Note that the water protection ports on the side of the passive boxes were left off during manufacturing for ease and because the team decided that water into the passive boxes in the weather sensor suite was not a risk.

4.3. Environmental

Shannon Chott

For the weather detection system, the enclosure for the sensor suite along with the wiring were both manufactured by the team, as described in the previous sections. In addition to these hardware elements, the code for the weather detection system was also manufactured by the team but utilized command libraries from the manufacturer and is described in the following section. All of the sensors along with the microcontroller were purchased from various vendors and integrated into the enclosure by the team.

4.3.1. Weather Detection Code

Trevor Willson, Austin Cyrus

The weather detection code manufactured by the WRAITH team serves the design outlined in the previous section. This was done by utilizing COTS libraries from the manufacturers of the sensors that included various commands for calling data from each sensor. Pre-made libraries were utilized for the lightning detector, the relative humidity/temperature/barometer board, IR thermometer, and the anemometer. These pre-made libraries were taken directly from the components manufacturers, and were intended for use on an Arduino device. These modules were first compiled in an Arduino sketch to ensure each individual sensor operated nominally on its own. From there, a preliminary sketch was created that executed the trigger condition logic for all of the sensors. Once this sketch was completed and verified, the WRAITH team began to port the sketch over to the Particle compiler. Since the WRAITH team opted to utilize a Particle E-Series microcontroller to operate the weather detection suite, the libraries had to be incorporated into the Particle IDE. Both the Particle IDE and Arduino are based on C++ logic, thus only minor edits needed to be completed to the libraries for the commands to function on the Particle board. The final function for the weather detection system was then composed in the Particle interface and based upon both the logic as previously seen in Fig. 17 and the preliminary Arduino sketch. The code was manufactured to first check for cellular inputs that could initiate safe mode, then read the single-trigger shutdown sensors, and then read the multiple-trigger sensors over the course of a 1Hz loop. As the logic read the sensors, the values would be individually compared to the pre-set safe mode conditions and a variable would be stored if the readings were outside the allowed range. Interrupts were introduced into the code to send the close command when any valid combination of the trigger conditions were met, conditions of which were being sampled once a second. In the event the weather readings were outside the allowed range, the system would set a variable to indicate the state and then record the current up-time as the ”last closed” time. This would prohibit the system from prematurely exiting safe mode since the safe mode variable would not reset if 30 minutes had not elapsed since the last time the system entered safe mode. In order to ensure the system did not open in adverse weather conditions, the logic still read each sensor and updated the ”last closed” time during the safe mode phase. Once the 30 minutes had elapsed, the system would then return to the nominal state.

5. Verification and Validation

Matthew Barrett

Much like the schedule, testing was broken down into three primary categories: hardware, software, and environmental. Through the course of the semester, the team managed to complete a portion of tests from each category, with some in progress and others planned when the shutdown occurred. Had all testing been successfully completed, the results would've built into both the full system test and long duration test. Details on how these tests were to be conducted, actual and expected results, as well as any required special equipment is discussed in the following subsections.

5.1. Full System Tests

Matthew Barrett

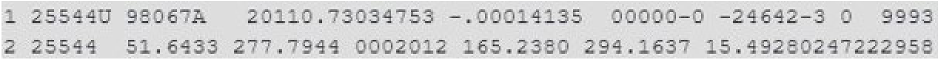
5.1.0.1 Test Design

The purpose of the full system test was twofold, to first build upon the tests described in the rest of this section and to prove the end-to-end capability of the WRAITH system. As such, the test setup looks similar to the CONOPS. The team would first begin by loading two line elements for the ISS and a HEO object from a flash drive onto the WRAITH system, after which WRAITH would be deployed to an open field. Once the scheduler has determined viewing windows for the objects, the actuation stage would be commanded to point the camera at the correct location. From there, the three streak images and a background image will be taken for image processing. Once the endpoints of the streaks are processed, the information would be passed on to the legacy orbit determination code to provide a final updated two line element. Throughout this, the weather detection system will be operating and will close the system if inclement weather is detected. This test provides a means of verifying all functional requirements except FR 2, though the full system test is only meant to last for a few hours. The scheduler successfully predicting visibility windows verifies FR1, if the suite detects poor weather and WRAITH closes then FR3 is successfully accomplished, and by tracking the ISS and a HEO object successfully FR4 is satisfied. Processing images within ten seconds would be verified in the code by recording timing information thus verifying FR5, the final updated orbit estimate being created and saved within five minutes would be verified through timing information again and thus satisfy FR6. Finally, by using a stopwatch to time deployment FR7 would be verified so long as it takes less than half an hour with two people. The long duration test would have been a full 12-hour deployment recording the same data points as the full system test with more objects scheduled, and by successfully operating for 12 hours autonomously the test would verify FR2.

5.1.0.2 Test Results

Not complete

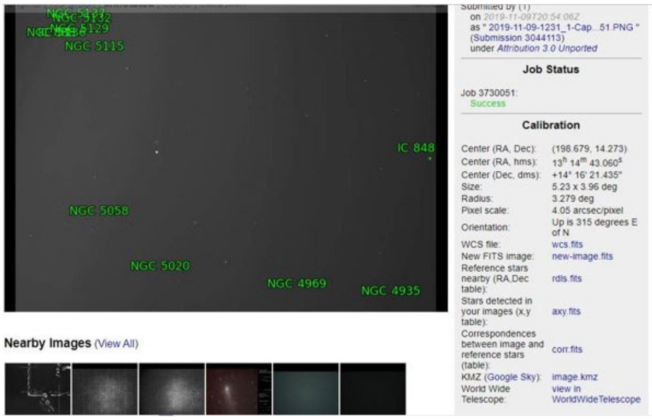
The first result that will come from this test is using a stopwatch to time two team members deploying the system. As long as it takes less than 30 minutes to deploy the system, the requirement is satisfied. Since this test will include a HEO object and the ISS, an example TLE for the ISS can be seen in Fig. 31.



```
1 25544U 98067A   20110.73034753 -.00014135  00000-0 -24642-3 0  9993
2 25544  51.6433 277.7944 0002012 165.2380 294.1637 15.49280247222958
```

Figure 31. Example ISS TLE

Once the system has completed scheduling and taking images, the operators will be able to perform several checks. First, the background image can be used to compare the right ascension and declination to what the scheduler provided. These expected results can be seen in Figs. 32 33. When both results agree with one another, the system has been checked for its capability to accurately schedule viewing windows and to track objects in the various orbits.



Submitted by (1)
on 2019-11-09T20:54:06Z
as " 2019-11-09-1231_1-Cap...51.PNG "
(Submission 3044113)
under *Attribution 3.0 Unported*

Job Status

Job 3730051:
Success

Calibration

Center (RA, Dec):	(198.679, 14.273)
Center (RA, hms):	$13^h\ 14^m\ 43.060^s$
Center (Dec, dms):	+14° 16' 21.435"
Size:	5.23 x 3.96 deg
Radius:	3.279 deg
Pixel scale:	4.05 arcsec/pixel
Orientation:	Up is 315 degrees E of N
WCS file:	wcs.fits
New FITS image:	new-image.fits
Reference stars nearby (RA,Dec table):	rdls.fits
Stars detected in your images (x,y table):	axy.fits
Correspondences between image and reference stars (table):	corr.fits
KMZ (Google Sky):	image.kmz
World Wide Telescope:	view in WorldWideTelescope

Figure 32. Example Background Image

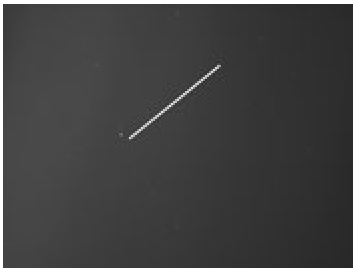
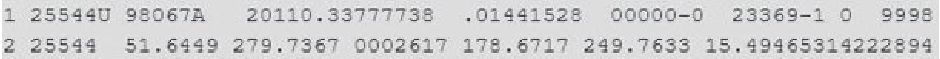


Figure 33. Example Streak Image

With the end points from the three streak images, orbit determination will provide an updated TLE to the operators. The difference between this updated TLE and the original will depend on the object being imaged. However in the case of the ISS, it would be expected to be very similar due to the abundance of tracking information and its easily visible orbit. This similarity is most important in the second row of the TLE, which contains orbit inclination, right ascension of the ascending node and eccentricity among other orbital elements. An example of the updated TLE can be seen in Fig. 34.



```
1 25544U 98067A   20110.33777738  .01441528  00000-0  23369-1 0  9998
2 25544  51.6449 279.7367 0002617 178.6717 249.7633 15.49465314222894
```

Figure 34. Example Updated TLE

Finally, throughout this test the weather detection system will be utilizing its sensor suite to detect if adverse weather is approaching or occurring. In the event of adverse weather, WRAITH will enter safe mode, and the conditions that triggered the close will be recorded. This would include the time closed, wind speed, relative humidity, pressure, lightning status, precipitation status, and IR temperature. This mirrors the full environmental system test which will be discussed in depth later. The major pieces of code will also continue to provide timing information, following the same method as the timing test described later. Finally, this would have successfully verified all of WRAITH's functional requirements except FR2 in one completed system through the successful tracking of target objects, obtaining updated orbit information, and operating autonomously.

5.2. Hardware

Dalton Turpen

In order to to produce a system that meets and exceeds requirements the performance of each component of the system must be verified and validated. The WRAITH system has many hardware components that are critical to the execution of the mission and a test plan was devised for each component that would allow us to characterize and quantify the performance of each individual component of the system as well as the hardware systems as a whole. The role of the hardware systems encompassed both the active and passive environmental protection for the main enclosure and the sensor suite as well as slewing and imaging. In order to have the system work as a whole, there were many batteries, motors, seals, electronic boards, and wires that had to be individually validated to ensure sufficient performance.

5.2.1. Batteries

5.2.1.1 Test Design

In order to verify and validate the nominal operation and requirement satisfaction of the batteries in the WRAITH system, a series of tests was planned to monitor the battery voltage and capacity over time. A battery cycle tester was purchased and used to verify nominal performance from each battery individually, ensuring we got the rated performance from each battery. The battery system we chose consisted of three 21Ah sealed lead acid batteries. To test the worst case scenario for power draw at 55.14 Ah with the batteries in parallel, the test current was derived from the following equation for the predicted 12 hours.

$$I = \frac{55.14}{duration} \tag{1}$$

With a current of 4.61A, the batteries were attached to the load and the results can be seen in the next subsection. The wiring configuration we designed allowed for the use of one, two, or three batteries at any given time. For the short duration and partial system tests, a single battery would be used and the voltage of the battery would be recorded throughout the tests; the data from these tests could be extrapolated to longer deployments with the full three battery system. For the long duration partial and full system tests, the battery system voltage would be recorded.

5.2.1.2 Test Results

Not complete

While testing the worst case power draw situation, the means of charging the batteries at this point in the semester was not the necessary 14.5V but instead 12V. This is why the test cuts off at 9 hours and doesn't reach the full 12. However the rate of the capacity change indicated that the three batteries were going to be capable of providing necessary amounts of power to operate WRAITH.

Time	Voltage [V]	Current [A]	Capacity [Ah]
30:22	12.5	-	2.32
1:00:00	12.1	4.59	4.62
2:01:58	12.1	4.63	9.36
3:16:56	12.1	4.61	15.13
4:03:51	12.1	4.61	18.73
5:00:00	12.1	4.63	23.05
7:04:35	12.1	4.63	32.63
8:00:00	12.1	4.61	36.89
9:00:43	12.1	-	41.55

Table 12. Batteries in Parallel - Test Results

The preliminary test results on each battery individually discharged by the battery cycle analyzer showed that all three of the sealed lead acid batteries were functioning nominally. The expected results for this test would have indicated that the batteries, at the end of the 12 hour mock deployment, still had an estimated 30% of their energy remaining.

5.2.2. Wiring

5.2.2.1 Test Design

For verification and validation of the wiring system, each of the electrical connections were redundantly tested by two different team members throughout the installation process to ensure that the designed voltage was achieved at each point in the wiring harness. Once the wiring harness was in place, every element was both continuity tested and tested to ensure proper voltage.

5.2.2.2 Test Results

During the process of using a multi-meter to verify correct continuity and voltage at each test point, we were able to redundantly verify that each connection was behaving nominally.

5.2.3. Weather Protection

To verify and validate the proper functionality of the active and passive weather protection systems, a series of tests were planned. Each of the tests was to utilize a hydro-sensitive paper that would indicate where water had been present. This paper was to be installed in all of the COTS boxes as well as a single piece to confirm all elements within the WRAITH unit would be water resistant.

5.2.4. Roof Actuation

Alex Kirk

Functional requirement 3 and design requirement 3.1 both pertain to the necessary functionality of the lid. Functional requirement 3 defines the need for the system to be able to autonomously enter and exit safe mode. Design requirement 3.1 defines that the active weather shielding must protect the sensitive components from adverse conditions. The roof is the main aspect of the hardware that protects the system when in safe mode, so its functionality is a crucial component that must be verified and validated to satisfy the given requirements. Other testing focused on the weatherproofing aspect of the system, while the roof actuation testing specifically focused on the mechanical behavior of the lid, as well as the motor controlling software. Testing of the lid had multiple phases. Physically the lid needed to smoothly actuate on the sliders, while having a complete range of motion necessary to cover the system when closed, and not obstruct field of view when open. This was functionality was tested by hand. Initially there was some resistance, but after adding lubrication and some slight screw adjustments the lid smoothly actuated. The next phase was in combination with preliminary roof control software. The team self-imposed a design point of 10s as the maximum time the lid could take to open/close and be considered successful. This testing was done using a stepper motor borrowed from the aerospace electronics lab to determine if the rack and pinion would operate the lid. This motor was able to reliably actuate the roof at 1 in/s, but struggled when sliders needed to telescope. The team formulated the following model to determine if the motor required to meet the 10s requirements.

5.2.4.1 Roof Motor Model

Due to the motion of the lid, there are 3 phases of motion whether opening or closing. These are constant positive acceleration, constant velocity, and constant negative acceleration. This is shown in Figure(36).

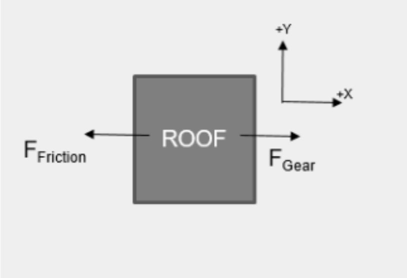


Figure 35. Roof Free Body Diagram

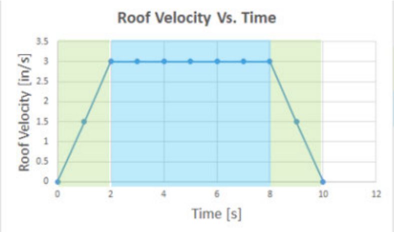


Figure 36. Roof Actuation vs Time

Using the free body diagram shown in Figure (35), the following set of equations was used to determine the necessary torque output to move the lid.

$$\Sigma F_x = m * a_x \tag{2}$$

$$F_{friction} = N * \mu_{static} \tag{3}$$

$$F_{gear} = N * \mu \tag{4}$$

$$a = \alpha * R \tag{5}$$

$$T = I * \alpha \tag{6}$$

$$\boxed{T_{required}} = 146 oz - in \tag{7}$$

The final aspect of the model is the physical relation between the gear and linear motion. The gearing used for this system has 24 teeth per revolution corresponding to 5 teeth per inch. The roof must actuate 24 inches when opening or closing. Using this information, the necessary RPM needed to close in 10s was found to be:

$$\boxed{10s\,Close : 30RPM} \tag{8}$$

5.2.4.2 Test Design

Now that the necessary design point was known to be 146 oz-in at 30 rpm the team bought a motor that would exceed these specifications. The motor design plot is shown in Figure (37). The green dot represents the design point necessary, while the three lines represent possible motor configurations. This design point was verified by using a similar procedure as with the borrowed motor. The speed of the motor was continuously ramped up to determine the maximum speed with which it could close the lid without any failures, or other mechanical issues. If this test was successful the next test would be a long duration test to ensure system reliability.

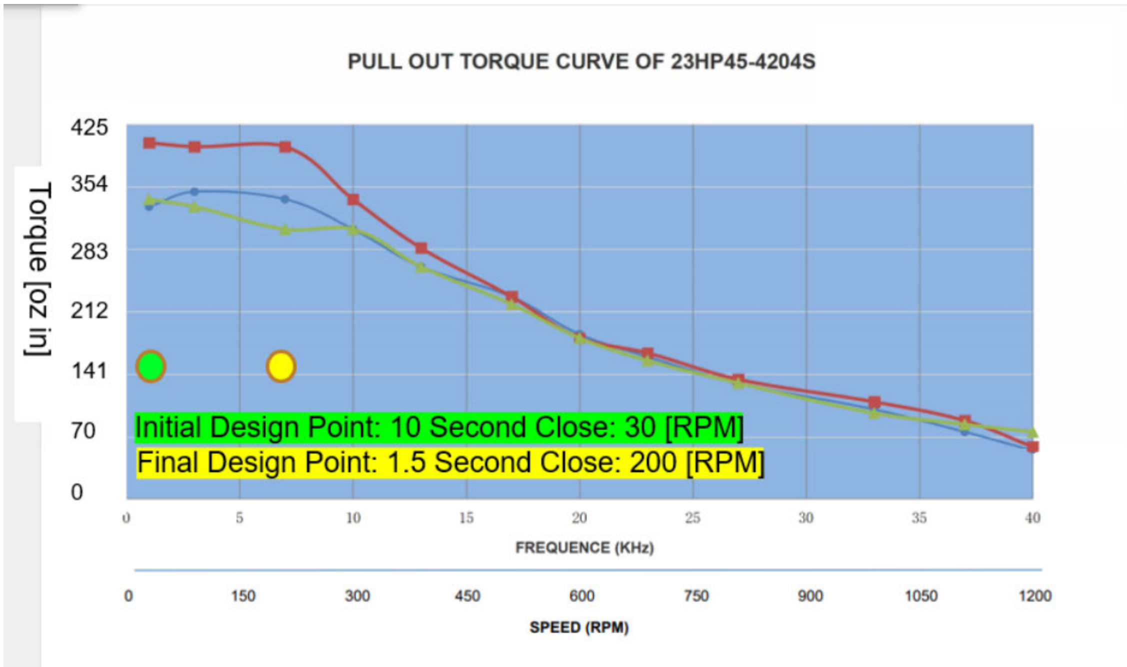


Figure 37. Stepper Motor Configurations and Design Points

5.2.4.3 Test Results

Using stop-watches and video to get accurate timing info the team was able to successfully cycle the lid open and closed in 3s, much faster than the initial design point of 10s. This speed led to smooth mechanical motion and no jamming, or other undesired behavior. The green dot on Figure (37) represents the final design point for the motor. With this timing and motor verified the long duration test was conducted to verify long term reliability. Over 20 minutes the lid was cycled 150 times at the 3s speed. This many repetitions is much more than would ever be expected on the most strenuous nights of observation, even with inclement weather. This test was videoed and can be found in the project archive. This test was extremely successful. No failures or incomplete actuation occurred during the test, and no elements were worn or hot to the touch. Physically the roof and motor were verified to be functional and reliable. Further testing would have been done to remove all bugs from code, but was not completed due to the early end of the project.

5.2.5. Slewing

Cal Van Doren

While there are no requirements that directly pertain to the performance of the stage itself, there are a few capabilities that it must have to support the verification of a variety of design requirements. First and foremost, the stage must slew predictably and in a consistent amount of time such that relative time sequences with the roof could be executed with reduced risk of a collision. Another key capability that the stage must have is consistency in pointing to the stowed position; given that there is only a 1/2 inch clearance between the camera and the roof in the stowed position, we need to be confident in the stage's ability to always stow in a correct orientation such that the roof wouldn't clip it while closing. With these two derived requirements being key to the success of the autonomy of the project, a test was designed to verify the models associated with the stage.

5.2.5.1 Stage Models

The model for the time which it will take the stage to slew based on the distance from the target on each the azimuth and elevation axes was derived from a single timed angular rate along each axis. The elevation axis was approximated to slew about 33% slower than the the azimuth axis which contributes to the fact that the elevation axis is more often expected to be the limiting factor in slewing time. The predicted slew time model is shown in Figure 38 where it can be seen that the expected maximum slew home time would be approximately 33 seconds which corresponds to the case where the camera is 180°off in the elevation axis. Furthermore, the predicted limiting axis for each location is mapped in Figure 39.

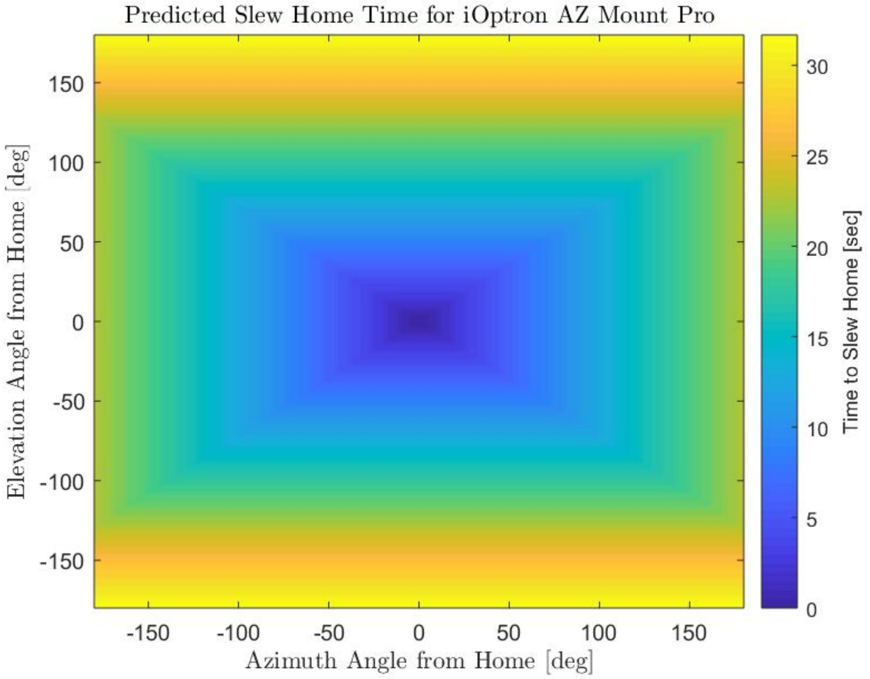


Figure 38. Model for Stage Slewing Times

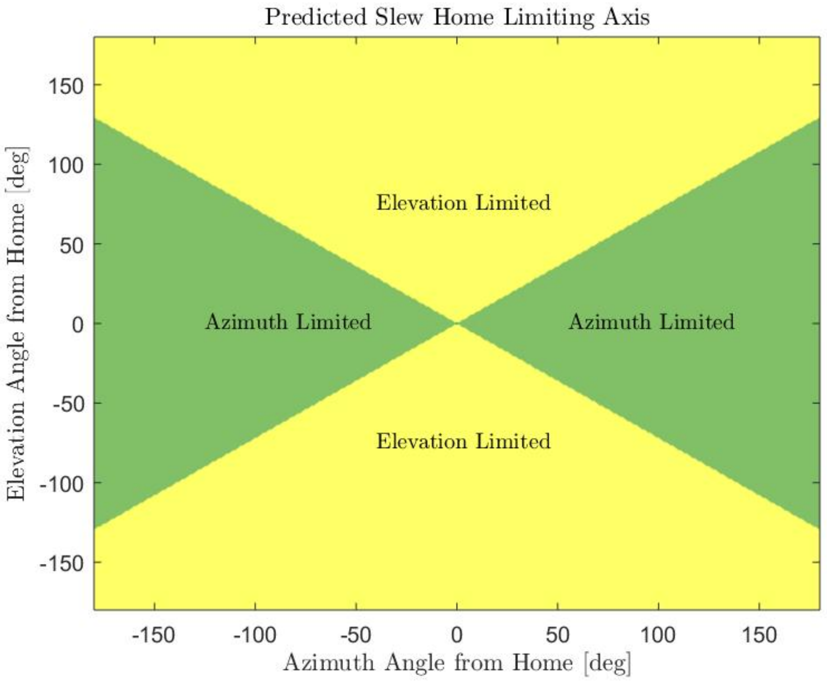


Figure 39. Expected Limiting Axis Throughout Model Space

5.2.5.2 Test Design

In order to verify the model shown above in Figure 38 a test was designed in which a series of design points were selected across the space covered by the model. These selected points are shown in Figure 40. The stage will be commanded to slew from each of these test points to the home position and the time that it takes to do so will be taken with a stopwatch. These data points will then be fed into MATLAB and a surface will be fit to the data; this surface could then be compared to the predicted model and differences could be reconciled.

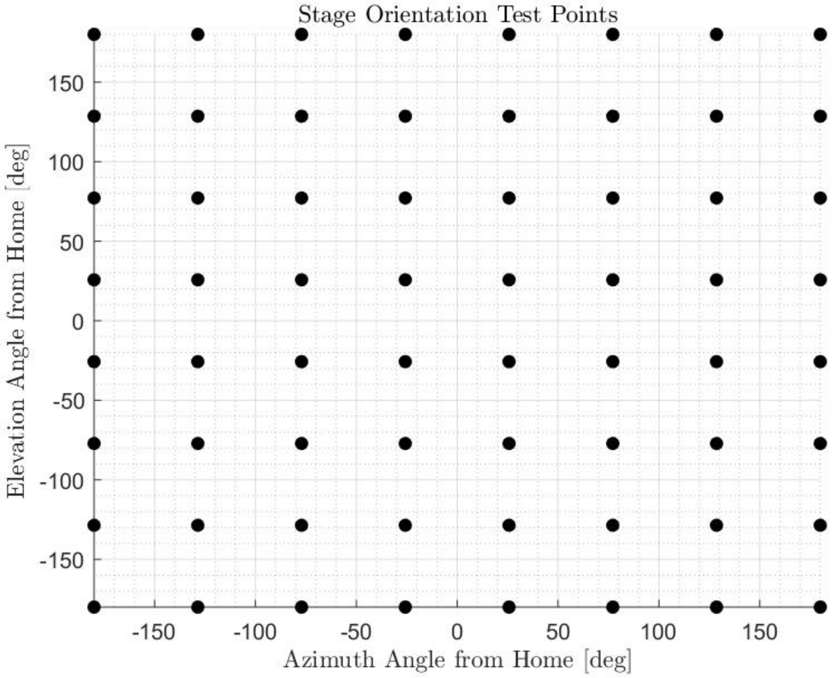


Figure 40. Test Points for Stage Slewing Verification Test

Furthermore, in order to better understand the accuracy in the home pointing, when each of these data points is tested, the accuracy of its pointing will be quantified by mounting a laser to the stage arm and marking its pointing on a white board 10 feet away. After the test is completed, the pointing each time the stage slewed home could then be categorized by one, two, and three sigma error levels which would give the team confidence that the home pointing is reliable.

5.2.5.3 Test Results

Unfortunately, this test was not run before the project had to be halted due to the COVID-19 outbreak. The slew time model is expected to match the actual results very closely within the error bounds of human reaction time due to the stopwatch method. The laser home pointing method would be expected to yield almost no visible difference in the pointing from any of the iterations which would show that the accuracy of the home pointing is not a concern.

5.3. Software

5.3.1. Partial System Test

Noah Crisler

The partial system test is aimed to validate three software subsystems, the live pass scheduler, the hardware scheduler, and the hardware commanding module. The test was key to verify WRAITH's ability to fulfill fundamental requirement 1, 4, and 6. This test will allow for much of WRAITH's crucial automation to be tested end to end for the first time. Due to the nature of the automation, it is very difficult to test all components without having all pieces present and being in a the actual deployment environment.

5.3.1.1 Test Design

The test will require all components of the WRAITH system except the environment detection suite, that is the UDOO processor, the active weather protection suite, the camera, and the stage actuation system. The test begins with the live pass scheduler creating a ".cmds" file which includes, among other information, the exact time and position at which the system needs to capture an object. The hardware scheduler then takes the information and uses a created algorithm to schedule and issue hardware commands to:

1. Open the roof
2. Slew to azimuth and elevation of object
3. Begin sidereal slewing
4. Auto-gain the camera
5. Take a dark image
6. Capture a streak of the object
7. Stop sidereal slewing
8. Close the roof

The above sequence is repeated multiple times for various objects and time stamps for each completed task and command sent are recorded to a log file.

5.3.1.2 Test Results

Not complete
At the time of quarantine the WRAITH team had completed the first version of the software and was days away from completing the first partial system test. One of the key components of WRAITH system is its complete autonomy. As with any autonomous system a great amount of tuning is needed to ensure the system can function without error for a long duration of time. If the test had been completed, the results of the log file combined with the images captured would have been used to tune the algorithm creating command times. This would ensure that each step outlined above, would be completed at the precise time needed to allow for an accurate capture of objects each time. In addition to the the tuning of the algorithm, the partial system test would allow for the verification of the systems ability to meet the desired target of six objects captured per hour.

5.3.2. Automated Calibration

Noah Crisler
Autonomous calibration is a key component of WRAITH to ensure it can meet its 12 hour deployment goal(FR 4). To allow operators to set up WRAITH in the daytime and then leave the system, WRAITH would have to be able to autonomously calibrate once the sun went down. The stage WRAITH uses to point from one location to the next was designed to be calibrated using a three star calibration. This manufacture calibration requires an operator to be present on scene and manually correct the error in pointing. If done correctly the calibration ensure the pointing of the stage has less than a 0.237°error in both azimuth and elevation. The autonomous calibration test would employ the autonomy designed by the WRAITH team to allow the system to match the manufacturer calibration without any operator inputs. Ensuring the system is calibrated accurately is key to the success of the rest of the WRAITH autonomy as an assumption is made that there is no error in pointing. Thus if an object is missed its assumed to be due to propagation error and not pointing error. The result of a successful vs. unsuccessful calibration is depicted in fig. 41.

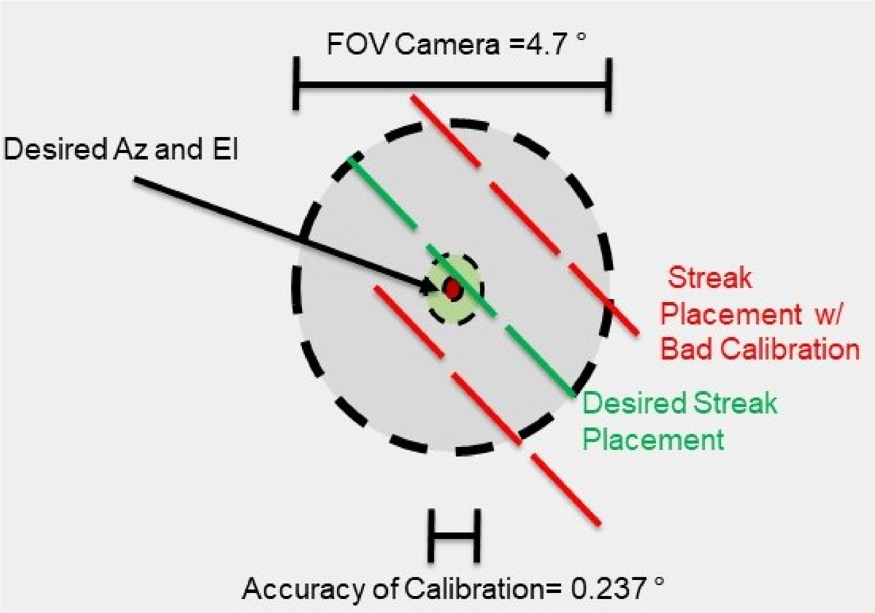


Figure 41. Model of Auto-Calibration Desired Result

5.3.2.1 Test Design

The design of the auto-calibration test allows for the number of calibration points needed to match the manufacturer calibration to be found. Each calibration point follows the following steps:

1. In an un-calibrated state WRAITH slews from a its horizontal stow position to vertical.
2. A picture of the night sky is then captured
3. This picture is passed into the image processing software, where a star map is used to determine the exact elevation and azimuth of the center of the camera Field of View (FOV).
4. The found azimuth and elevation are then compared with where the stage reports its current azimuth and elevation to calculate a error delta for each.

The error delta calculated is applied to each successive point taken. The entire processed is then repeated numerous times and recorded to a log file.

5.3.2.2 Test Results

Not complete
As with other planned tests, the auto-calibration test was not completed due to the covid-19 quarantine. If the testing had been completed the team would have expected the data to be similar to the data shown in fig. 42. Its important to note the data in this figure is not real and is just a mock up of how the system was designed to run.

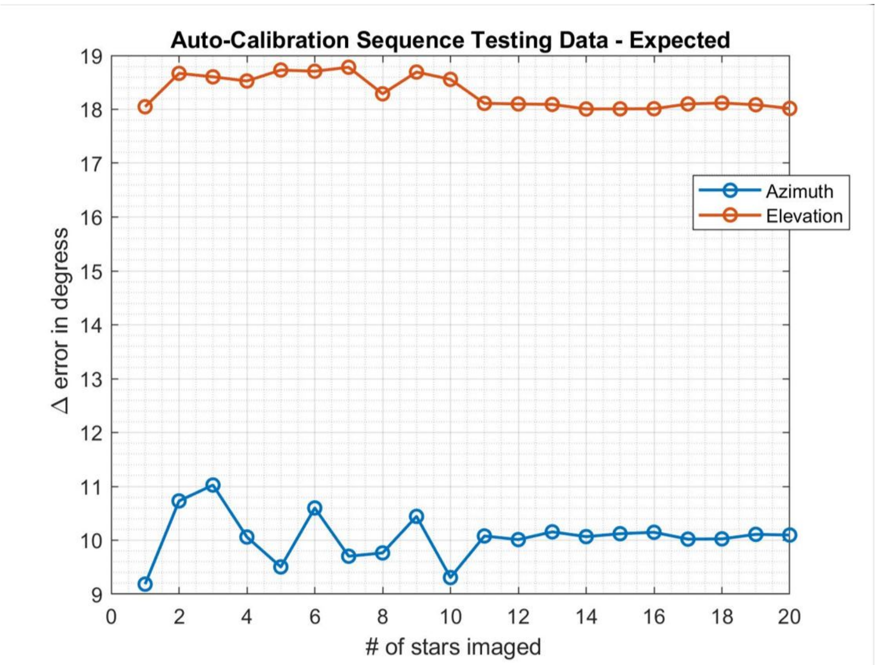


Figure 42. Mock data of the auto-calibration test

The test data would have been used to determine both how the many calibration points were needed and the estimated duration the script would need to complete. A successful test would allow for the script to take all the calibration points, average them, and find a constant offset that would allow for accurate pointing throughout the entire deployment.

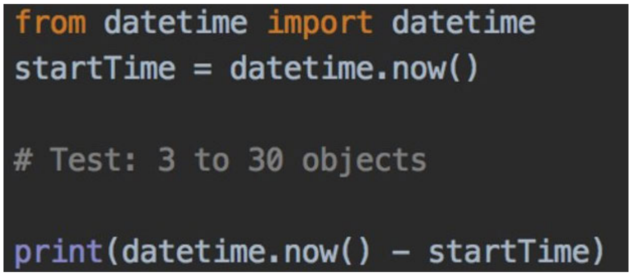
5.3.3. Timing Test

Matthew Barrett

In order to move towards verifying Functional Requirement 6, the team intended to add code to each major system to record the time each action would take. This would also provide the team with more information on general system function as well as ensuring a maximum of 6 objects could be tracked in an hour. Additionally, accurate timing was crucial for building the imaging sequence, as the hardware should be commanded at times relative to each other.

5.3.3.1 Test Design

Each major function of WRAITH would have included code such as that in Fig. 43. The *datetime* function is native to Python and records the time to the microsecond, if desired. This would have provided time information for the scheduler, hardware commanding, imaging, image processing, and orbit determination. The test would have been conducted just prior to the full system test, and wouldn't require any special equipment. WRAITH would be deployed with certain targets to view and the timing information would supplement the normal operation of the system. In order to ensure accurate timing information, the test would be conducted for multiple objects, ranging from 3 to 30, over multiple nights. The goal is to eliminate any outlier times that would arise from a cloudy picture, extremely long or short slew distances, differences that could arise if more or less pictures are taken of the same object, and other unknown interferences.



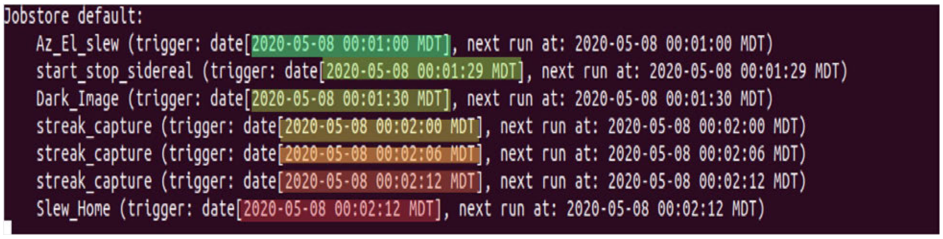
```
from datetime import datetime
startTime = datetime.now()

# Test: 3 to 30 objects

print(datetime.now() - startTime)
```

Figure 43. Timing Test - Example Code

5.3.3.2 Test Results

Not complete

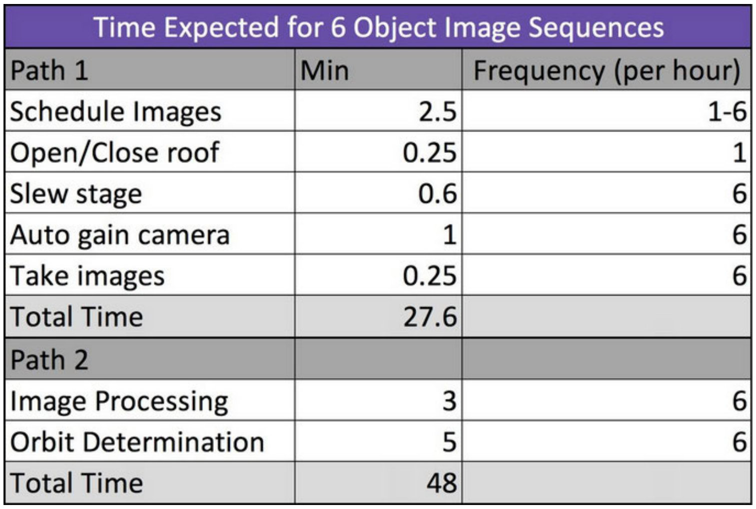
One goal of this test is to gather timing information in order to building the imaging sequence commands, among other sequenced commands. An example of the imaging sequence is shown below. The test would allow the team to better define the highlighted times in order to ensure both that the commands do not overlap and that as much time as possible is conserved between them.



```
Jobstore default:
    Az_El_slew (trigger: date[2020-05-08 00:01:00 MDT], next run at: 2020-05-08 00:01:00 MDT)
    start_stop_sidereal (trigger: date[2020-05-08 00:01:29 MDT], next run at: 2020-05-08 00:01:29 MDT)
    Dark_Image (trigger: date[2020-05-08 00:01:30 MDT], next run at: 2020-05-08 00:01:30 MDT)
    streak_capture (trigger: date[2020-05-08 00:02:00 MDT], next run at: 2020-05-08 00:02:00 MDT)
    streak_capture (trigger: date[2020-05-08 00:02:06 MDT], next run at: 2020-05-08 00:02:06 MDT)
    streak_capture (trigger: date[2020-05-08 00:02:12 MDT], next run at: 2020-05-08 00:02:12 MDT)
    Slew_Home (trigger: date[2020-05-08 00:02:12 MDT], next run at: 2020-05-08 00:02:12 MDT)
```

Figure 44. Timing Model

As this test wasn't completed, a rough estimate of expected results would have been numbers following the model created as seen in Fig. 45. In practice, the team would have expected better results than provided by the model, as the model is assuming worst case scenarios for how long actions will take. If a particular action took far longer than anticipated, this would have indicated a potential issue and troubleshooting would then occur.



Time Expected for 6 Object Image Sequences		
Path 1	Min	Frequency (per hour)
Schedule Images	2.5	1-6
Open/Close roof	0.25	1
Slew stage	0.6	6
Auto gain camera	1	6
Take images	0.25	6
Total Time	27.6	
Path 2		
Image Processing	3	6
Orbit Determination	5	6
Total Time	48	

Figure 45. Timing Model

5.4. Environmental

Shannon Chott

5.4.1. Individual Sensor Characterization Tests

Before the sensors were integrated together, the team needed to ensure their accuracy by comparing the sensor output to known values, such as those provided by the National Weather Service. To do this, each sensor was tested in a variety of ways and their outputs compared to whatever known values were available. In addition to characterizing the sensor behavior, these tests also serve to validate **Function Requirement 7** which states that WRAITH must be deployed in under 30 minutes. These tests will show that the sensors are capable of starting up and outputting accurate data in less than 30 minutes, as per the requirement. These characterization tests were carried out using an Arduino Uno with open-source code, and PuTTY was utilized to log the data as *.txt* files. These files were then loaded into Matlab so the results could be plotted and analyzed. These lower level tests were fully completed before the shut-down due to Covid-19 and they are outlined in detail in the following sections.

5.4.1.1 Anemometer

Test Design:
To ensure the accuracy of the cup anemometer, it was tested in two ways. The first was using a hairdryer along with a handheld anemometer from the Electronics Lab. The handheld anemometer was held at roughly 2 feet from the hairdryer with the hairdryer set to low speed. The wind speed outputted by the handheld anemometer was recorded. Next, the handheld anemometer was switched with the cup anemometer at the same distance and with the hairdryer at the same speed. The anemometer was powered by a single battery from the main WRAITH system as well as being connected to an Arduino Uno and a laptop running code and Putty to log the data. The wind speeds outputted by the cup anemometer were then compared to the wind speeds outputted by the handheld anemometer.

The second test was performed outside the Aerospace Building on a windy day. It was performed in a similar manner to the hairdryer test, with the handheld anemometer being held in a certain location and then switched with the cup anemometer in the same location. The wind speeds were recorded from both anemometers and then compared. For this test, since the winds were gusty and quite variable, the trends were compared rather than the actual wind speeds. If the cup anemometer showed increasing and decreasing wind speeds at the same times that the handheld anemometer showed the same trends, the result was considered positive.

Model:
The model to compare the test results to for the anemometer tests was the output from the handheld anemometer. These values were taken to be the ”known” values to determine if the cup anemometer from the weather detection system was accurate.

Test Results: For the hairdryer test, the recorded wind speeds can be seen in Table 13 below. The wind speeds were similar from the cup anemometer to the handheld anemometer, showing that the cup anemometer was reading fairly accurate wind speeds. It is important to note that these wind speeds were quite variable over the course of the test which was most likely due to the small stream of air outputting by the hairdryer. In natural conditions, the winds are moving in more of a vector field as opposed to a single vector, as is simulated here. Because of this, only one side of the cup anemometer was being impacted by the simulated wind instead of both sides like in a natural scenario. This could potentially affect the outputted wind speeds from the cup anemometer so a conservative trigger condition was chosen to account for any inaccuracies held by the cup anemometer.

Table 13. Anemometer Hairdryer Test Results

Cup Anemometer Wind Speed [m/s]:	4.2 ± 0.2
Handheld Anemometer Wind Speed [m/s]:	4.0 ± 0.2

For the outside wind test, specific values were not compared because the wind speeds were constantly shifting, making it difficult to accurately take data points from both anemometers and compare the values. Instead of this, the trends were compared between the two. It was found that when a gust of wind came through, the cup anemometer increased the outputted wind speed at about the same rate as the handheld anemometer. The same is true for decreasing winds down to no winds, as both anemometers reached 0 at roughly the same time. This test confirmed that there was no significant lag in the cup anemometer as it tracked the shifting winds similar to the handheld anemometer.

5.4.1.2 Relative Humidity Sensor

Test Design:
There were two tests performed to characterize the RH sensor. The first was a series of binary checks to determine if the RH sensor was reading the same values as the NWS weather stations. To perform this test, the RH sensor was connected to an Arduino Uno and the Arduino was connected to a laptop that was running open-source code and Putty to log the data. The sensor was taken outside of the Aerospace building on CU Boulder's East Campus and allowed to run until it reached steady state. This steady state value was recorded and then immediately compared with the NWS value from the closest weather station at Boulder Municipal Airport. This binary check was repeated a few times to ensure that the sensor maintained accurate readings. These binary checks also validated **Functional Requirement 7** since they prove that the RH sensor can reach steady state in under 30 minutes.

The second test was a maximum test to determine if the sensor could track sharp increases in RH as well as read high RH values since this is common during adverse weather. For this test, a hot shower was turned on in an enclosed bathroom and ran until there was visible steam in the air, indicating 100% RH. The sensor (in the same test configuration as the binary checks) was allowed to reach steady state in a room away from the shower, and then was abruptly brought into the steam-filled room and once again allowed to reach steady state.

Model:
Since both of these tests are basic checks to ensure that the sensor is reading accurate values, the models are simply the expected or known values that the sensor output is being compared to, so the NWS data for the first series of tests and 100% for the maximum RH test.

Test Results:
The binary check results can be seen in Table 14 below. It's important to note that these checks were all performed at night since the sensor is sensitive to direct sunlight which causes erroneous measurements. All of these binary checks from the sensor were within 1-2% of the reported NWS values. Since relative humidity alone is not harmful to the system, these results were deemed accurate enough to fulfill the weather detection system's purpose. In addition, the trigger condition for RH was set as low as possible at 70% to prevent unnecessary closures while also giving about a 10% margin of error between the average RH that rainstorms generally occur at, which is closer to 80%. These results confirm that the RH sensor is capable of accurately measuring the relative humidity compared with the NWS.

Table 14. Relative Humidity Binary Check Test Results

	Sensor RH %:	**NWS RH %:**
Trial 1:	39	38
Trial 2:	47	49
Trial 3:	23	24

Fig. 46 below shows the results from the maximum RH test. The RH in the room away from the steam-filled room was about 29%, as seen on the plot. Around 400 seconds, the sensor was brought into the steam-filled room and it only took about 4 minutes (roughly 250 seconds) to reach steady-state around the expected value of 93%. This expected value was determined by taking the maximum value our sensor can read according to the data sheet, 90%, and then adding in the positive error, 3%. While the RH in the room is actually 100%, the sensor is not capable of reading that high so it was expected that it would output it's maximum value instead, as shown on the plot. This test confirmed a few things: the sensor can reach equilibrium in under 30 minutes per FR 7, the sensor is capable of tracking drastic changes in RH, and the sensor is capable of reading it's highest possible RH value when expected.

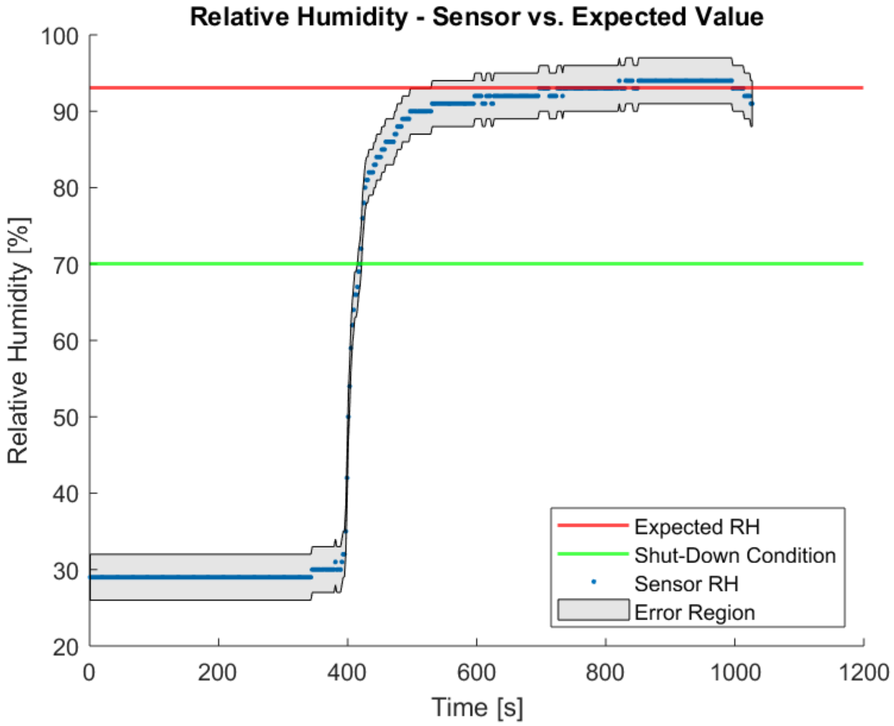


Figure 46. Maximum RH Test Results

5.4.1.3 Barometer

Test Design:
There were two tests performed on the barometer as well to determine if the output was accurate. The first test was a series of binary checks run at the same time as the RH binary checks since these two sensors are housed on the same breakout board. These tests were performed in the same way as the RH tests and compared with pressure values from the NWS.

The second test was a trend test where the sensor (in the same test configuration as with the binary checks) was driven from Boulder, Colorado to the top of Berthoud Pass and back down into Winter Park. The intent of this test was to ensure that the sensor could track various changes in pressure as well as read the low pressures seen at the top of the pass. These values were checked along the way to determine consistency and the value at the top of Berthoud Pass was checked against the NWS station values from the same location.

Model:
For the binary checks, the model is once again the values from the NWS. For the trend test, the model was the altitude of the route. Since pressure decreases with altitude, there is a inverse relationship between pressure and altitude. As such, along the route to Berthoud Pass, if the altitude was increasing, the pressure ws expected to decrease. As for the top of Berthoud Pass, there is an NWS station at the top of the peak. Since there is roughly 1000ft of elevation difference between the road and the peak, the NWS reported value was mathematically corrected to compare to the sensor value. This correction is discussed below.

The NWS pressure recording from the station at the top of Berthoud Pass was roughly **629 mb**. The sensor output from the road on top of Berthoud Pass was roughly **659 mb**. The altitude of the NWS station is 12490ft while the altitude of the road at it's highest point is 11307ft, so there is a total altitude difference of **1183ft**. In atmospheric science calculations, for every 26.25ft increase in altitude, the pressure will decrease by 1mb.

$$\frac{1183ft}{26.25ft} = 45.07mb \tag{9}$$

Therefore, the pressure at the NWS station should be about 45mb less than the pressure at the road. According to the NWS value and the sensor output, there was a **30mb** decrease from the road to the NWS station. This means that the sensor should have been outputting a pressure that was about **15mb less** than what was actually outputted, so about 613.93mb. This could mean that the pressure sensor either does not accurately read lower pressures, or it was not given enough time to fully equalize as the the test was not paused at the highest altitude. Some potential sources of error for this correction include an inaccurate altimeter reading from the breakout board, which would cause an inaccurate pressure measurement, or a lack of time to reach steady state. In addition, the closest pressure reading from the NWS was reported about an hour earlier than the test was conducted, so it's possible that the pressure changed in that hour although unlikely that it changed by 15mb. Lastly and most importantly, the standard pressure change per increase in altitude is generally used at lower altitudes, such as those at 3000ft and below. This means that this correction is most likely not as accurate as it could be for the high altitudes the test was conducted at, 11000 and 12000ft. Since pressure decreases exponentially with altitude, it is extremely likely that the linear approximation is introducing a source of error for this correction.

Test Results:

For the binary checks, the results can be seen in Table 15 below. For each trial, the pressures were within about 1-2mb of each other, exactly as expected since the barometer has an error of ± 1mb. As with the RH sensor, pressure itself is not harmful to the system so these results were deemed accurate enough for the purpose of the system. Additionally, the trigger condition is a pressure increase of 3mb, which is conservative given that pressure changes preceding storms generally occur over the course of days and not 12 hours. This conservative trigger condition combined with these results confirm that the barometer is capable of detecting pressure accurately.

Table 15. Pressure Binary Check Test Results

	Sensor Pressure [mb]:	**NWS Pressure [mb]:**
Trial 1:	838.0	836.3
Trial 2:	838.7	836.6
Trial 3:	843.9	842.1
Trial 4:	844.2	842.4

Fig. 47 shows the pressure trend from Boulder, CO to the top of Berthoud Pass, and then back down the pass into Winter Park, CO. As can be seen on the plot, the pressure is generally decreasing as it journeys up the pass until roughly 1900 seconds when it reaches it's lowest pressure at the summit of Berthoud Pass. It then begins increasing again as it drops in altitude down the other side of the pass. The trip from Boulder into the mountains has some hills, accounting for the increasing and decreasing bumps seen early on in the test. Once the beginning of the pass is reached, the altitude steadily increases, accounting for the steadily decreasing pressure, just as expected. These results confirm that the sensor is capable of tracking sensitive pressure changes over a long duration, as well as detecting significant drops in pressure which is critical for detecting incoming storms and low pressure systems.

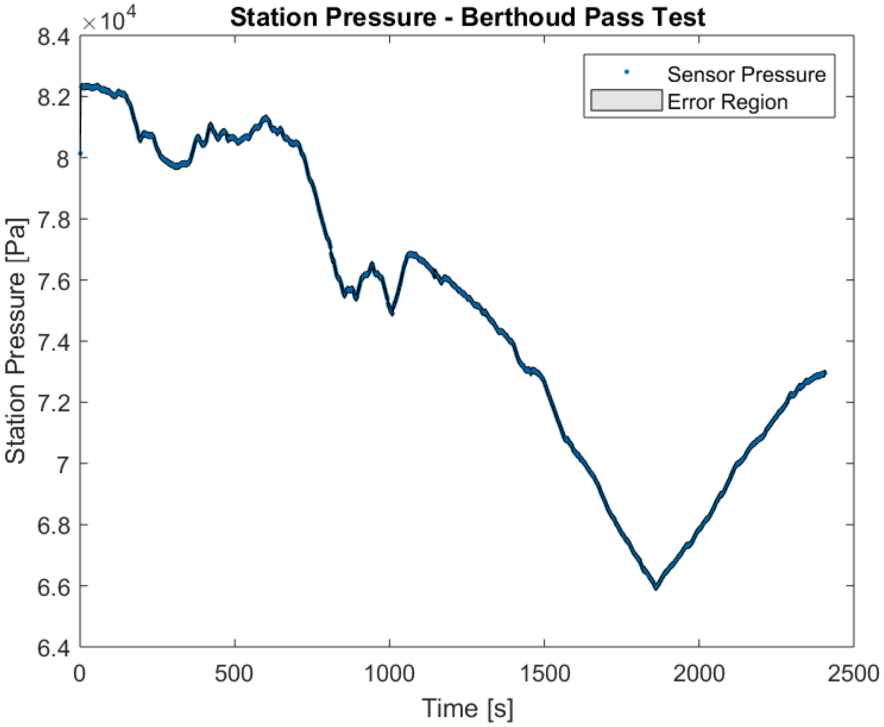


Figure 47. Berthoud Pass Pressure Trend Test Results

5.4.1.4 Infrared Thermometer

Test Design:
To test the IR thermometer, a series of binary checks were performed much like with the RH and pressure sensors. The sensor was connected to an Arduino Uno and a laptop that was runnning code and PuTTY to log the data. The sensor was taken outside to a location with a clear view of the sky and no obstructions in the field of view of the sensor. It was then pointed straight at the sky and allowed to come to equilibrium. While the sensor was logging data, the operator of the test would record observations about the environmental conditions that could affect the sky temperature. These conditions included: cloud cover in oktas, relative humidity (NWS), ambient temperature (NWS), visibility (NWS), precipitation events, and any additional observations such as haze, smoke, or fog. It is important to note that IR thermometers are incredibly sensitive to direct sunlight, so all of these tests were performed at night. The binary checks were performed multiple times under multiple sky coverage conditions such as fully overcast, fully clear, and various conditions in between like broken or scattered clouds. These binary checks were used to determine what temperature should be used as the trigger condition for this sensor to indicate cloud cover.

In addition to characterizing the temperature of the sky under various conditions, the sensor also needed to be tested through a variety of roof materials to see what materials were transparent to IR in the wavelength range that this sensor sees in. These materials included polycarbonate, acrylic, and a specific type of optical glass. To perform these tests, the sensor was placed inside, pointed at the ceiling, to record the temperature of the open air and ceiling. Next, the operator's hand was placed over the sensor, roughly 6 inches above it. This temperature was also recorded. The chosen material was then placed over the sensor so that the entire field of view of the sensor was through the material. The temperature of the material was recorded. Lastly, the operator placed their hand above the sensor and the material (again about 6 inches above it) and that temperature was recorded. If the sensor could detect the increase in IR temperature from the operator's hand to be the same as the temperature of the hand from before the material was placed in the way, then the material was deemed fully transparent for the sensor. This test was critical because the sensor was going to be placed inside the weather detection system enclosure so the team needed to determine what the roof material should be to keep the sensor safe but also infringe on it's ability to detect IR temperatures.

Test Results:
Table 16 shows the characterizations of the sky temperature for a variety of sky conditions. These results clearly

show a distinct difference between a clear sky and a cloudy sky, with clear skies being negative and generally around 20°F - 30°F and cloudy skies being positive and in the teens. Broken clouds from 7 oktas of cloud cover are on the edge of precipitating clouds, as scattered cloud cover will generally not produce significant precipitation. Because of this and since clear sky temperatures do not approach positive temperatures in the slightest, the trigger condition was conservatively chosen as 10°F. This condition will not produce false-positives as clear skies are much colder, but it is high enough to determine when the sky is mostly covered by clouds. This is important because there will not be precipitation without clouds (excluding the rare event of sun showers), and the WRAITH system cannot image satellites under cloudy conditions anyway, so when there is significant cloud cover such as 6 oktas or higher, WRAITH should be in safe mode.

Table 16. IR Thermometer Sky Temperature Characterization

Sky Condition:	Sky Temperature [°F]:
Clear sky, 0 oktas of cloud cover, slight haze on the horizon	-23
Overcast, 8 oktas of cloud cover, light snow	18
Broken clouds, 7 oktas of cloud cover	14
Clear sky, 0 oktas of cloud cover	-30
Clear sky, 0 oktas of cloud cover	-23

These characterizations were critical to determine the approximate temperature of a cloudy sky, however the sensor still must be able to see through whatever material it is housed under. These results can be seen in Table 17 below. As can be seen by the results in the table, the IR thermometer could not see through the polycarbonate or the acrylic at all because when the operator's hand was moved over the sensor and the material, there was no change in the outputted object temperature. This means that the sensor was measuring the temperature of the material and could not see through it. When the optical glass was tested, the object temperature increased very slightly, but not as high as it should have. This indicates that some wavelengths were transparent for this material but not all of them. The team attempted to change the emissivity of the sensor in the code as advised by the data sheet to see if it was possible to fully see through the optical glass, however this yielded no better results than the initial test. As such, it was decided that the best option was simply to expose the tip of the sensor through the enclosure roof material to ensure that the sensor could obtain accurate readings. Before this could be done, the team needed to ensure that the part of the sensor that was exposed was waterproof and would not break in the event of precipitation. The sensor was placed through the roof material (polycarbonate) and sealing using caulking. Once it was dry, a generous amount of water was poured over the sensor. The sensor did not sustain any damage and the outputted values did not change with the addition of the water, therefore the sensor was deemed waterproof and exposed through the roof instead of fully inside the enclosure.

Table 17. IR Thermometer Material Transparencies

Material:	Transparent to IR Thermometer?
Polycarbonate	No
Acrylic	No
Optical Glass	Slightly
Exposed (no material)	Yes

5.4.1.5 Lightning Detector

Test Design:

The purpose behind the lightning detector test was to see if a simulated lightning strike would cause the sensor to think there was lightning nearby. The sensor was wired to an Arduino Uno with a laptop running the code and Putty to log the data. The team then used multiple apparatuses to determine which ones would trip the sensor. These included an arc welder, an electronic arc lighter, and an arbitrary motor found in the lab. In addition to verifying that the sensor would detect the necessary energy signatures, it was also tested with a car starting and with WRAITH starting to ensure that common occurrences that may happen around the system would not cause any false-positives.

In an ideal scenario, the team would have tested the sensor in an actual thunderstorm and compared the lightning strike counts and distances with data from an online lightning tracker. However, given that the time of testing was late winter and thunderstorms are sparse during this season, this was unable to be completed.

Test Results:
As can be seen in Table 18 below, the first four apparatuses caused the lightning detector to trip and say a strike was detected. The last two, a car starting and WRAITH powering on, did not cause the sensor to say that a strike was detected. This confirms that the sensor can detect similar energy signatures to lightning strikes without causing false-positives due to non-lightning events in the area.

Table 18. Lightning Detector Test Results

Simulated Lightning Strike:	Lightning Strike Detected by Sensor?
Arc Welder	Yes
Electronic Arc Lighter	Yes
Motor	Yes
Electric Drill	Yes
Car Starting	No
WRAITH Powering On	No

5.4.1.6 Precipitation Sensor

Test Design:
To ensure that the precipitation sensor could detect all types of precipitation that WRAITH could see in the field, the precipitation sensor was tested in as many precipitation events as possible. These events included moderate rain, simulated rain (drops of water from a bottle), light snow, and mist. The sensor was connected to an Arduino Uno with a laptop running the code and Putty to log the data. During the precipitation event, the sensor was brought outside and held out to catch the precipitation. This was repeated in various scenarios to fully characterize what precipitation events the sensor could detect and what events it was not capable of sensing.

Test Results:
The precipitation sensor was tested under the following conditions: **simulated rain consisting of water droplets from a bottle, moderate rain, light snow, and mist**. A dry board reads a value of 4095, and the lower that outputted value drops the more moisture is detected on the board. All of the tested conditions caused the outputted value to drop significantly with the exception of mist. The mist droplets on the precipitation board were not large enough to short out the board, which is how it senses the moisture. Because of this, the sensor thought the board was completely dry when it was actually damp. This was deemed acceptable purely because the other environmental factors that cause mist, such as overcast skies and high relative humidity, would have caused the other sensors on the weather detection system to send WRAITH into safe mode, so even if the precipitation sensor does not detect the moisture WRAITH would still be protected.

It is also important to note that the light snow caused only a slight drop in the outputted value until the snow had the chance to melt on the room-temperature board. The more snowflakes melted on the board, the lower the value dropped. To ensure that snowflakes or any sort of frozen precipitation melted on the board, resistance heaters were added to each precipitation board.

5.4.2. Environmental System Test

To test the completed weather detection system, a full environmental system test was planned to validate **Functional Requirement 3** which states that WRAITH must autonomously protect itself from adverse weather. To protect itself from adverse weather, the system must be able to accurately detect the weather conditions over the course of a 12 hour deployment. The purpose of this test is to determine how much confidence the team could hold in this system to ensure safe deployments. Before this test could take place, all the individual components needed to be completed and tested on their own. These components include the environmental code, the characterization of the individual sensors, the enclosure, and all the wiring within the enclosure. All of these components were completed right as the project was stopped, however the test itself was unfortunately never completed due to Covid-19. The environmental system test is explained in the following sections along with what the team expected the results to look like had the test had been completed.

5.4.2.1 Model

Atmospheric science is incredibly dynamic and as such, while many aspects of it are rooted in equations and models, there is not one mathematical model that clearly predicts any type of adverse weather that WRAITH could encounter over a 12-hour deployment. Since this test is tracking various weather conditions, from light rain to thundersnow, these mathematical principles do not provide a solid model to compare test results. As such, the model for this test is a qualitative model comprised of the data compiled by the operator of the test. While the weather detection system is deployed for this test, an operator will also be present for the duration of the test and will be periodically recording observations and data of their own. Ideally, the operator will have a working knowledge of atmospheric science so they can make informed and detailed observations. These observations will include details such as oktas of cloud cover, the thickness and type of the clouds (such as cumulonimbus for thunderstorm-producing clouds), the apparent wind speed, prevailing visibility, and type of precipitation (such as snow, freezing rain, or mist). In addition to these qualitative observations, the operator will also use NWS data from a nearby weather station to record the relative humidity and pressure level. These observations and pulled data points make up the model that the sensor output will be compared to. While not a traditional model that can be seen in equations and plots, this model will provide solid values and observations to ensure that the weather detection system is functioning properly.

5.4.2.2 Test Design

The set-up for this test involves deploying the completed weather detection system outside at a location where it has a relatively clear view of the sky, simulating a typical WRAITH deployment. The completed weather detection system is shown in Fig. 48. The system would be powered from one of the three batteries used to power the whole system to negate the need for wall power and a long USB cable will connect the microcontroller to a laptop. The system will be deployed for a significant number of hours depending on the forecasted weather. While the system is deployed, it will be continuously collecting data and logging it in a *.txt* file using PuTTY. While the system is deployed, an operator, such as the Environmental Systems Lead, will periodically go outside at the deployment location and record qualitative observations regarding the weather conditions. These observations will be as specific as possible, as discussed in the Model section, and will include the time of the observation. Using these observations, the operator will conclude whether they deem the conditions safe enough for WRAITH to operate or if it should be in safe mode at that time.

Once the test has been completed, the weather detection system data will be pulled from the logged file and run through the trigger conditions for entering safe mode. This will show if the system thought WRAITH should be in safe mode or not based on the sensor readings. This data will then be compared to what the operator thought at that time. If both the operator and the weather detection system came to the same conclusion about safe mode, then that result is considered positive. This test will be completed as many times as is necessary to reach 100 observation times and ideally, 95 of those 100 observations will be positive. This would indicate a 95% success rate for the weather detection system and would validate the model as well as Functional Requirement 3. It is important to note that a 95% success rate reaches the first level of success. To fully reach our goal at the third level of success, the system would need to have greater than 99% success rate which would involve additional test iterations to fully determine. As such, it was determined that a 95% success rate provides enough confidence in the system to deploy WRAITH fully without extreme risk of environmental damage.

Figure 48. Completed Weather Detection System

5.4.2.3 Test Results

Not complete

As this test was never completed, this section will outline what the team expected the results to look like had the test been completed. The results shown here along with the test scenario are fabricated to show what the output of the test would have looked like and how it would have been used to validate Functional Requirement 3. Fig. 49 shows an example of what the operator would see during an iteration of this test. The operator would record observations at a specific time such as those seen in the list below. Based on these observations, the operator would conclude that WRAITH should be in safe mode at this time.

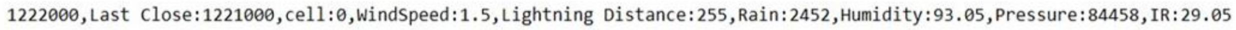
- Moderate to heavy rain
- Low level cumulonimbus clouds (thunderstorm) indicating possible lightning in the area, though none was observed
- 100% relative humidity from NWS Boulder Municipal Airport
- Light, breezy winds
- Safe mode should be enabled

Figure 49. Example Weather Conditions for Environmental Test

At the same time as the operator's observations, the data logged by the weather detection system would output as a line in a text file, such as the one seen in Fig. 50. This line of data shows the last time of close for the lid, indicating that safe mode was entered. The output from the individual sensors is also outputted in this line, giving the following information:

- Wind Speed: $1.5\frac{m}{s}$
- Lightning Distance: 255 indicating no lightning nearby
- **Rain: 2452 indicating a significant amount of moisture on the board due to the value being much less than 4095**
- **Humidity: 93.05 indicating that the RH sensor is reading it's maximum value**
- Pressure: 84458 indicating a pressure level of 844.58mb which is around nominal for Boulder, Colorado
- **IR: 29.05 showing an IR temperature of 29.05°F which indicates overcast sky coverage**

The bold points in this list indicate which conditions would have triggered the safe mode command from the weather detection system to the main WRAITH system. In this case, there are two cases that would have sent the command: the precipitation detected as well as the combination of high relative humidity and high IR temperature. Having multiple sensors indicate the need to enter safe mode in the case of a typical thunderstorm was intentional to mitigate the risk of single-point failures. The precipitation sensor acts as a fail-safe in the case that the multiple sensor triggers fail since water is the most detrimental to the system. There are also two precipitation boards implemented in the weather detection system so if one fails there is a backup.



```
1222000,Last Close:1221000,cell:0,WindSpeed:1.5,Lightning Distance:255,Rain:2452,Humidity:93.05,Pressure:84458,IR:29.05
```

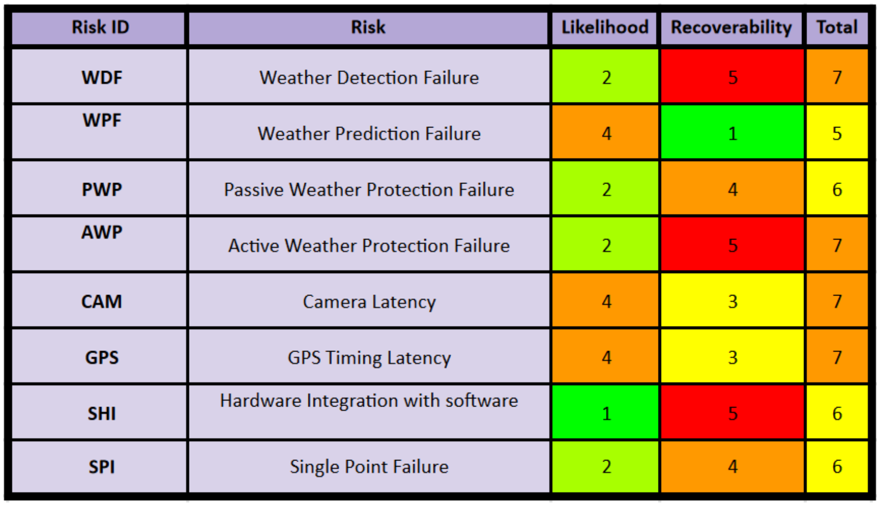
Figure 50. Example Data Output from Weather Detection System

Since the weather detection system would have commanded WRAITH to enter safe mode and the operator also thought WRAITH should be in safe mode, this example scenario indicates a match and would yield a positive result for this observation time. The next step is to iterate this test at a different observation time and repeat the process until a minimum of 100 observations were completed.

6. Risk Assessment and Mitigation

Julian Jurkoic

Risks for the project were initially identified by the team at an early stage, and subsequently assigned scores based on their likelihood and the projects ability to recover from a given risk event occurring. In both categories, a higher score is considered riskier Below one can find a list of the risks identified as highest concern during the critical design portion of this project. A comprehensive list of all risks can be found in Appendix E along with a legend.



Risk ID	Risk	Likelihood	Recoverability	Total
WDF	Weather Detection Failure	2	5	7
WPF	Weather Prediction Failure	4	1	5
PWP	Passive Weather Protection Failure	2	4	6
AWP	Active Weather Protection Failure	2	5	7
CAM	Camera Latency	4	3	7
GPS	GPS Timing Latency	4	3	7
SHI	Hardware Integration with software	1	5	6
SPI	Single Point Failure	2	4	6

Table 19. Key Risks

During the project, risks were monitored and tracked on the team's risk spreadsheet (see Appendix E). All risks of score 7 out of 10 or higher were discussed at weekly meetings to either mitigate their potential, or ensure that they do not worsen if they have already been mitigated.

A discussion of every single risk, its effect on the project, and mitigation strategies for the risk is beyond the scope of this report. However, there are some key risks which had major effects on this project which this report would not be complete without a discussion of.

One of the key project concerns which was repeatedly voiced by the project advisory board during WRAITH's design development stage was that of a failure of the weather protection system in the event of adverse weather conditions. As far as mitigation for this risk goes, the best strategy the team was able to adopt was choosing a more simple, robust design as detailed above in the Design Process and Outcome section. This decision came to fruition when the protection system was tested to statistical confidence in the closing mechanism.

As can be seen above in figure 19, one of our highest concern risks at the critical design phase of this project and throughout the rest has been that of single point failure. This is the idea that if, for whatever reason, a team member was unable to continue working, the lack of their knowledge wouldn't be a major hindrance to project work. Two main strategies were put in place to mitigate this risk. The first being that project work was scheduled in a linear fashion so that as many team members get experience with each work package as possible. The second of these strategies was called "The rule of 2's", the main idea being that no team member should work on any project work without the aid of another. The second of these two rules ended up being to simple and strict for project work, and as such was not always followed. This resulted in a different type of single point failure in which one team member was responsible for ensuring the power system was correctly wired. This caused the loss of a circuit board on the iOptron actuation stage, which was promptly replaced, but caused an unexpected increase in budget for the project.

Of course the more comprehensive list of risks in the appendix is far from exhaustive, and the risk event which eventually put an end to this project was not even considered at the outset. Of course, this is in reference to the COVID-19 pandemic which put an end to all aerospace capstone projects in the spring 2020. This risk wasn't tracked until the threat it posed was imminent, so mitigation strategies were limited. The team rushed the finish of the hardware component of the project in anticipation of the shut down of machine shop resources, and though successful, proved to be futile as all project work was ordered to stop.

7. Project Planning

Julian Jurkoic

Throughout this section there are a few key considerations that were taken into account in the planning of this project. First, as you will see in the next section, the entire WRAITH team is organized into three sub teams: Software, Hardware, and Environmental. This organization is maintained in the work breakdown structure as well as the work plan section. Second, many steps were being taken to ensure the smooth integration of various subsystems and between these teams. This was often given priority over further developing the project as it is more important that a full system is delivered to the customer.

7.1. Organizational Chart

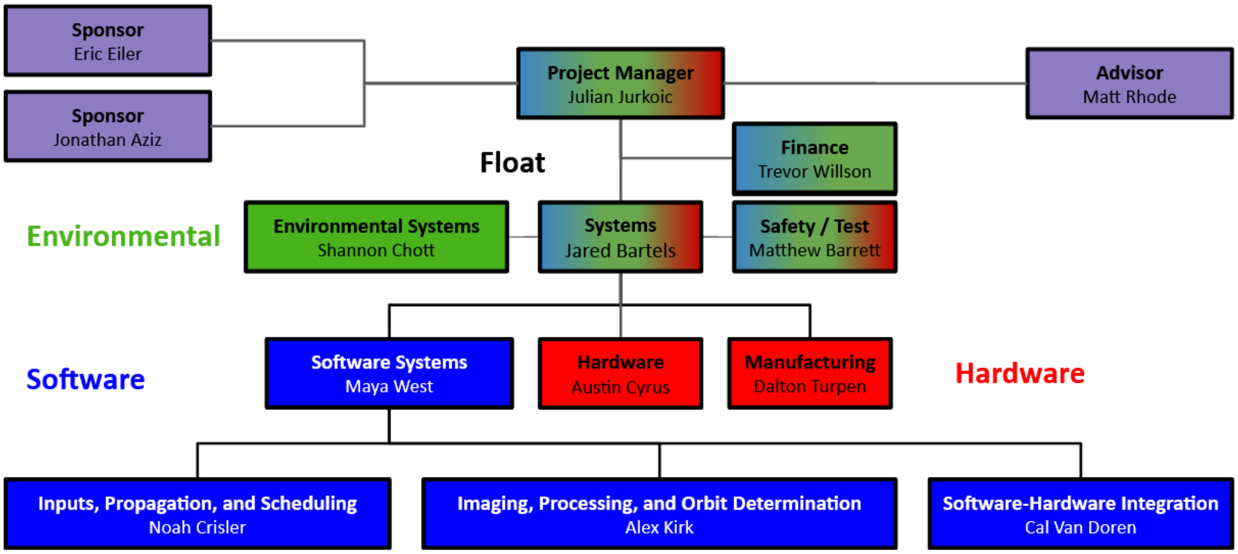


Figure 51. Organizational Chart

As you can see in the organizational chart above, each team member has a leadership position on the team. Additionally, one can see that the team has been further organized into three sub teams. This allows for groups of people to become intimately familiar with a large functional part of the project. Furthermore, four of the team members worked with multiple of these teams, thus aiding in the integration between them.

7.2. Work Breakdown Structure

The following page shows the work breakdown structure for the project on a high level for the sake of brevity. A complete work breakdown structure can be found in Appendix F. This abbreviated work breakdown structure is color coded to show which work packages were completed at the time of project shut down, which were in progress, and which had yet to be started.

🟩 Complete
🟨 Started
🟥 Not Started

- 🟨 1. Software
  - 🟨 1.1. Software Hardware Integration
    - 🟨 1.1.1. Hardware Commanding
    - 🟨 1.1.2. Safety Override
    - 🟥 1.1.3. GPS
    - 🟩 1.1.4. Roof Controller
  - 🟩 1.2. Software Integration
    - 🟩 1.2.1. Scheduler to Control interface
    - 🟩 1.2.2. Camera Output
    - 🟩 1.2.3. Memory Path
    - 🟩 1.2.4. Image Processing to orbit determination
  - 🟥 1.3. Scheduler Redesign
    - 🟥 1.3.1. Rerun Function
    - 🟥 1.3.2. Shutdown and return home for environmental system
    - 🟥 1.3.3. Integrate cable unwrap
  - 🟩 1.4. Sensor Suite
    - 🟩 1.4.1. Determine Sensor Functionality
    - 🟩 1.4.2. Characterize uninstalled sensors
    - 🟨 1.4.3. Characterize installed sensors as a system
    - 🟩 1.4.4. Shutdown logic coding
    - 🟨 1.4.5. Communications
- 🟩 2. Hardware
  - 🟩 2.1. Main Box
    - 🟩 2.1.1. Prototype
    - 🟩 2.1.2. Actual
    - 🟩 2.1.3. Lid Assembly
    - 🟩 2.1.4. COTS Box Modifications
    - 🟩 2.1.5. Internal Component Installation
    - 🟩 2.1.6. Wiring and Electrical Install
  - 🟩 2.2. Sensor Suite Box
    - 🟩 2.2.1. Build Box
    - 🟩 2.2.2. Install Components
    - 🟩 2.2.3. Install Wiring

7.3. Work Plan

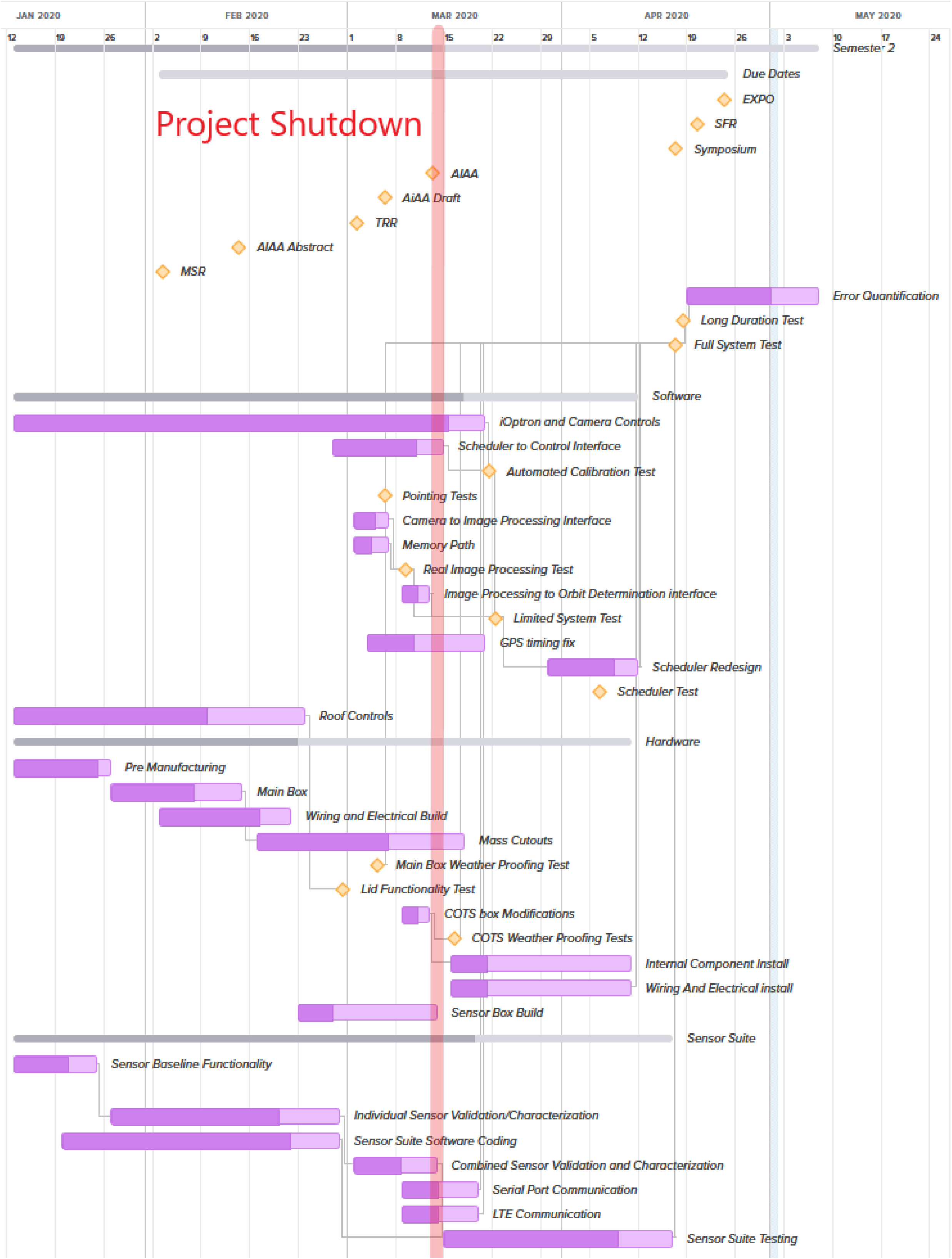


Figure 52. Gantt Chart

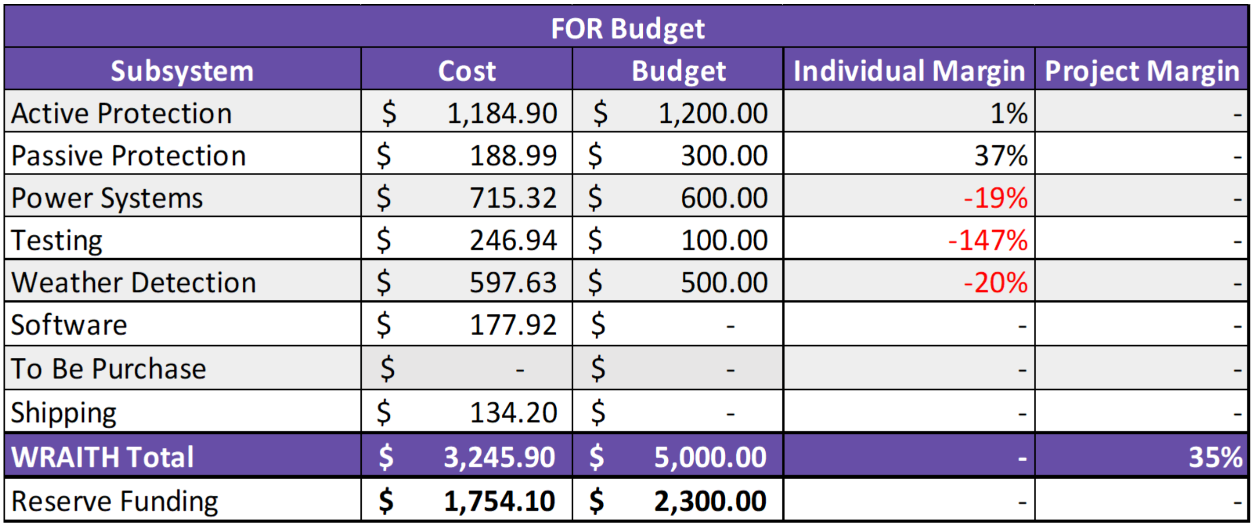
The above Gantt chart is once again broken into the three major teams. It is of note, that within each team, a parallel path approach was avoided. This served the goal of making sure the project was well integrated and was done on the recommendation of old GHOST members. The idea being that if all members of a certain team help deliver each work product in sprints, each member of the sub team will have a better knowledge of each piece of the project for when integration issues come up. This additionally allowed for better tracking of progress as well as helps mitigate the risk of single point failures (having one person responsible for a whole subsystem). Each team therefore has it's own critical path, with the software team's critical path being the critical path for the whole project.

This is the schedule as it stood at the time of project shutdown. It can be compared with the schedule as it stood during the critical design portion of the project in Appendix G. The project schedule evolved as the duration of certain work packages became more clear when they were being completed, particularly in regard to integrating the software systems with the hardware systems.

7.4. Cost Plan

Given WRAITH is a continuation of the GHOST project, the team acquired various hardware components from the previous year. Due to this, many fundamental components have already been acquired and do not need to be purchased this year. As explored in previous sections, the main components the WRAITH team inherited were the Ioptron actuation stage, the ZWO camera, and the UDOO processor. These components would have drastically increased the budget had the team needed to purchase these in addition to the other pieces. The overall budget of the WRAITH project is broken down into several different subsystems and presented in Table 20. Notably, the Active Weather Protection and Power Systems encompass the bulk of the anticipated costs, however, the Weather Detection subsystem for the sensors and the hardware covers a reasonable portion as well. As seen in Table 20, the total expenses for the project was $3245.90 of the total $5000 budget. This results in a project margin of about 35%. The final expenditures for the project differs from the estimated expenses from the Critical Design Review and the Fall Final Report since the team received ample feedback suggesting the team buys down risk and difficulty with the excess budget. This resulted in an increase in Active Weather Protection expenses since more expensive components were purchased to increase the longevity and minimize the risk to the project from weather conditions. In addition, the entire Software category was added in the spring semester as a means to expedite the software manufacturing and testing process. This category comprised of the purchase of a chrome-book to operate as a Linux machine in order to proved the team with another resource in developing and testing code.

Table 20. Overall Budget



FOR Budget				
Subsystem	**Cost**	**Budget**	**Individual Margin**	**Project Margin**
Active Protection	$ 1,184.90	$ 1,200.00	1%	-
Passive Protection	$ 188.99	$ 300.00	37%	-
Power Systems	$ 715.32	$ 600.00	-19%	-
Testing	$ 246.94	$ 100.00	-147%	-
Weather Detection	$ 597.63	$ 500.00	-20%	-
Software	$ 177.92	$ -	-	-
To Be Purchase	$ -	$ -	-	-
Shipping	$ 134.20	$ -	-	-
WRAITH Total	**$ 3,245.90**	**$ 5,000.00**	**-**	**35%**
Reserve Funding	**$ 1,754.10**	**$ 2,300.00**	-	-

7.5. Test Plan

Matthew Barrett

The test schedule for the WRAITH system has gone through several iterations, both as knowledge of the system increased as well as updating to reflect problems that proved to be more of a challenge than anticipated. The final test schedule can be seen in Fig. 53.

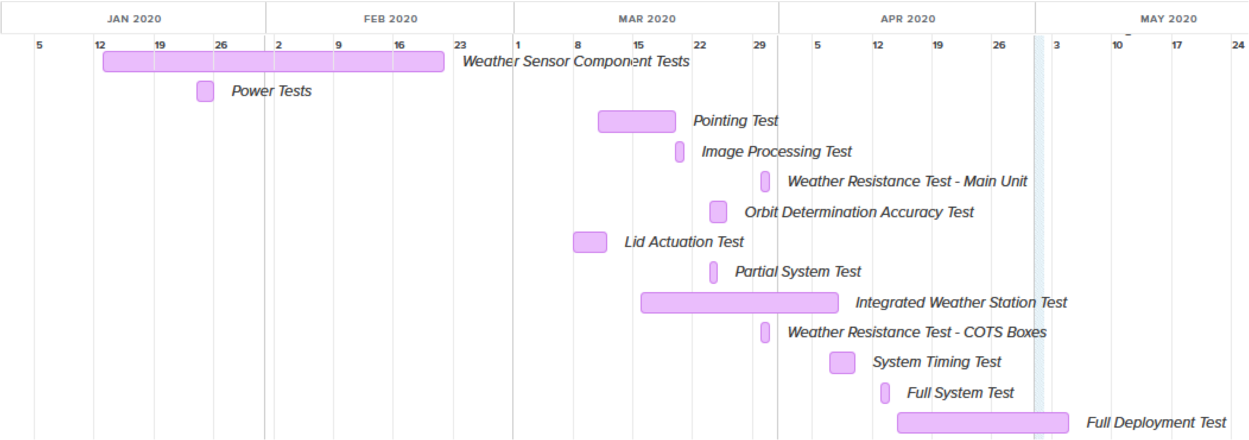


Figure 53. Test Schedule

Prior to shutdown, the individual weather sensor tests, lid actuation, and preliminary battery testing had been completed with full environmental testing being mere days away from starting. Note at the time of project shutdown, the start of the week of March 15th, 2020, the team was in the process of completing the pointing tests. This included slew speed testing and calibration testing. Completing this test was proving to take longer than anticipated, though just prior to shutdown several members of the WRAITH team made a breakthrough and were confident in proceeding with the test. Once the pointing tests had been completed, it would have been a quick transition to image processing followed by the partial system test done in conjunction with the orbit determination accuracy test. Outside of the partial system test, the other major tests were the integrated weather system tests and full system test. The only outside equipment required for testing was necessary at the beginning of the semester with the use of the handheld anemometer to compare to WRAITH's cup anemometer and the purchase of an electronic load to test the batteries and verify their ability to provide necessary amounts of power.

8. Lessons Learned

Julian Jurkoic

The lessons learned during the course of this project by the members of the WRAITH team are both invaluable and innumerable. Arguably, the most important lessons we learned revolved around working as a team in a professional manner and solving complex problems while understanding how to keep the PAB, our customer, and ourselves happy so as to deliver a project that we were proud of. That is to say these most important lessons revolved around how to be good engineers which is the sort of lesson one can really only learn by actually going through the motions of working as a team, solving complex problems, and understanding how to keep stakeholders happy. The object of this section is not to teach those most important lessons as they must be learned by doing. What we can offer is some advice that we feel will make the next generation of project teams' lives easier.

Starting at the very beginning of the project, one of the first tasks is to define the project's requirements. When defining these requirements, there is a line that is walked. On one side one can define requirements in a manner that is less regimented and more ”implementation free” allowing for more freedom in design choices, on the other one can define requirements in a more ”constricted” manner thus limiting design choices. As a team, we often found ourselves too far on the side of implementation free, and didn't always have requirements that should have been included. We can't tell you exactly how much more constrained the requirements should have been, but it is worth **serious** consideration at the beginning of the project. One strategy which we would implement in the future would be to get important **performance based** requirements from discussion with the customer, and build other requirements working backwards from that.

With regard to legacy projects, the work load should be very heavy on the front end. A legacy project team should strive to learn everything there is to learn about the previous project as soon as possible. This should include the good and, more importantly, the bad. The WRAITH team was still learning new things about GHOST as late as project shutdown, and it was not the successes which the old team were eager to share that hurt us.

One of the risks the PAB is most concerned with is single point failure, and rightfully so. To mitigate this we made a rule in which no project member should work on the project without getting a second set of eyes on it. This is a great idea, most of the time. There are, however, many instances when it's not needed, especially during testing. We had no plan for such situations, and because of that we lost hardware and had our schedule delayed. It's important to have such a plan to mitigate this risk. One idea the WRAITH team had is to have checklists to make sure the operator uses the hardware correctly.

Two of our more successful strategies were starting with integration on the schedule and making sure to avoid parallel-pathing on the schedule to ensure many team members were familiar with each system. These came from a large integration failure of the previous team which we strived to avoid. Even so, we really struggled with integration, it warrants a lot of thought in the planning phase. In this effort to have a cohesive system, we often missed the performance based objectives the PAB sought. Now we understand it is important to have both.

Finally, enjoy it while it lasts, if we learned anything this year is that things can change very quickly. As mentioned earlier, the lessons learned in this project are innumerable, but even so, here is a list of some of the more important ones that will hopefully make the next teams' lives easier.

- Start testing as early as possible because unforeseen things will go wrong and it takes time to figure out the issues or possibly purchase more components
- Good documentation, organization, and version control will save tons of time when it comes to report writing and presentation creation.
- Analyzing images is harder than you think.

9. Individual Report Contributions

9.1. Julian Jurkoic

Design Contributions: Project Planning, Project Risks, Customer Communication, PAB communication, Progress tracking, Meeting running, Project Integration, Weather Detection Trade Study, Power system wiring, Power system construction
Report Contributions: Project Purpose, Design outcome, Project Planning through work plan, Lessons Learned.

9.2. Jared Bartels

Design Contributions: Functional and design requirements, system functional block diagrams, main box power and sensor suite wiring design and manufacturing, UART communication
Report Contributions: Functional Requirements in Project Objectives and Functional Requirements, Requirements Flowdown, Sensor Suite Wiring

9.3. Alex Kirk

Design Contributions: Motor control software, image processing, arduino to UDOO interface, active weather protection trade studies
Report Contributions: Legacy Software, Roof motor commanding, Roof actuation testing.

9.4. Shannon Chott

Design Contributions: Design of weather detection system and trade study, design of safe mode trigger conditions, design of individual sensor characterization tests, design of weather detection system wiring diagram, design of final environmental system confidence model and test
Report Contributions: Design Process and Outcome - Weather Detection, Manufacturing - Environmental, and Verification and Validation - Environmental

9.5. Noah Crisler

Design Contributions: Auto-Calibration Script, scheduler to hardware interface scripts, python wrapping of hardware shell scripts for commanding, missing objects algorithm, passive weather protection design, active weather protection

trade studies
Report Contributions: Software Design, Software Verification and Validation, Passive and Active weather Protection trade studies.

9.6. Matthew Barrett

Design Contributions: Radio trade study, sensor suite trade study, helped with creating initial functional requirements, test plans
Report Contributions: Final system tests, timing test, test plan

9.7. Maya West

Design Contributions: Missing Search Algorithm, Calibration Trade Studies, Scheduler to Hardware Interface, Rerun Scheduler Design
Report Contributions: Hardware Scheduling Manufacturing, Timing Test Design and Results

9.8. Austin Cyrus

Design Contributions: Power System, Active Weather Protection, Passive Weather Protection, Weather Sensor Suite Code
Report Contributions: Passive Weather Protection, Manufacturing

9.9. Trevor Willson

Design Contributions: Weather sensor research and assistance to Weather Detection trade study, radio trade study, Camera commanding script, design of shutdown conditions, weather detection logic design
Report Contributions: Weather Detection Design Process and Outcome, Camera Commanding, Weather Detection Code, Cost Plan, Weather Detection Trade Study, Radio Trade Study

9.10. Cal Van Doren

Design Contributions: Design of the hardware-software interface and the commanding logic for each of the three main pieces of hardware.
Report Contributions: Conceptual and detailed design sections for the software-hardware interface

9.11. Dalton Turpen

Design Contributions: Main Enclosure Trade Study, Main Enclosure Design, Weather Sensor Enclosure Trade Study, Weather Sensor Enclosure Design, Lid Actuation Design, Umbilical Design
Report Contributions: Hardware Design, Hardware Manufacturing,

References

[1] Hidore, J. J., Oliver, J. E., Snow, M. (2010). Climatology: An Atmospheric Science (3rd ed.). New York, NY: Prentice-Hall.

[2] US Department of Commerce, NOAA. (2016, September 2). Beaufort Wind Scale. Retrieved March 5, 2020, from https://www.weather.gov/mfl/beaufort

[3] US Department of Commerce, NOAA. (n.d.). National Weather Service. Retrieved March 6, 2020, from https://forecast.weather.gov/MapClick.php?CityName=Boulderstate=COsite=BOUlat=40.0269lon=-105.251

[4] Pultarova, T., ”Meet the Space Custodians: Debris Cleanup Plans Emerge,” Space.com, April 2017. https://www.space.com/36602-space-junk-cleanup-concepts.html

[5] Space Situational Awareness, ”Space Surveillance and Tracking - SST Segment,” European Space Agency, November 2017. http://www.esa.int/Our_Activities/Operations/Space_ Situational_Awareness/Space_Surveillance_and_Tracking_-_SST_Segment

[6] Space Track, spacetrack.org

[7] Astrometry, astrometry.net

[8] Scott Kittelman, Professional Lab Assistant, Department of Atmospheric and Oceanic Sciences, University of Colorado Boulder,

[9] “Environmental Sensor Data Sheets.” Sparkfun Electronics, *Sparkfun.com*.

[10] “Adafruit Cellular Breakout.” Adafruit Industries, www.adafruit.com.

[11] Hernández, María. “Top 6 Affordable Cellular IoT Hardware Devices.” Ubidots Blog, Ubidots Blog, 6 Mar. 2019, ubidots.com/blog/cellular-iot-devices-comparison.

[12] “3G Network Comparison.” Cellular Maps.com, www.cellularmaps.com/3g_compare.shtml.

[13] ”Particle Device Firmware”, Particle Industries, docs.particle.io/reference/device-os/firmware/electron/

[14] ”E-Series Module Datasheet.” Particle Industries, docs.particle.io/datasheets/cellular/e-series-datasheet/.

[15] ”Anemometer Wind Speed Sensor with Analog Voltage Output”, Adafruit, https://www.adafruit.com/product/1733

[16] ”SparkFun Atmospheric Sensor Breakout - BME280”, SparkFun Electronics, https://www.sparkfun.com/products/13676

[17] ”SparkFun Lightning Detector - AS3935” Sparkfun Electronics, learn.sparkfun.com/products.

[18] ”XLX Rain Drop Sensor”, Amazon, amazon.com/XLX-Detection-Monitor-Large-Sensitive-Humidity-High/

[19] ”SparkFun IR Thermometer Evaluation Board - MLX90614”, SparkFun Electronics, https://www.sparkfun.com/products/10740

[20] Marple, Steve (2017). Python Binding for the ZWO ASI Library [Software]. Available from https://github.com/stevemarple/python-zwoasi

[21] ”Measuring the Temperature of the Sky and Clouds”, My NASA Data, 4 Dec. 2018. https://mynasadata.larc.nasa.gov/

[22] ”Beaufort Wind Scale”, National Weather Service, https://www.weather.gov/mfl/beaufort

Appendices

A. Trade Studies

A.1. Sensor Suite

Shannon Chott, Trevor Willson

The ability to detect weather is vital to WRAITH operations because a false-positive in adverse weather detection would impact observation windows while failure to detect adverse conditions would jeopardize the integrity of the system. To ensure that the weather detection system is as accurate as possible while remaining within the scope of the project, multiple design ideas were considered. A trade study was performed with these three design methods to choose the best method for this project. The three methods considered for this system were: an On-Board Sensor Suite, Radar Influenced Commands, and a combination of the aforementioned methods.

A.1.1. On-Board Sensor Suite

The On-Board Sensor Suite is the implementation and integration of multiple sensors onto the WRAITH system. The entire sensor suite would consist of a barometer, humidity and temperature sensor, anemometer, lightning detector, and a precipitation detector. Fig. 54[9] provides an example of each sensor that would be included in the sensor suite. The sensors would work in unison, such that the readings from multiple sensors would be used to assess the risk of adverse weather. For example, a drastic decrease in pressure alone might suggest the arrival of a low pressure system; however, a rapid rise in relative humidity along with the drop in pressure would provide more confidence in the arrival of a precipitation system. In addition to a multi-sensor state determination method, the system would have fail-safes in place to combat rapid environmental condition changes that could not be predicted beforehand. One example of this is the precipitation sensor which strictly measures precipitation in the moment. This allows the system to identify adverse conditions with no lead time as a last resort. Additionally, the suite will include an IR temperature sensor to measure the sky temperature, since higher temperature readings indicate cloud cover. Overall, the sensor suite would provide some weather condition predictability while also providing immediate last-resort features to decrease environmental risks.

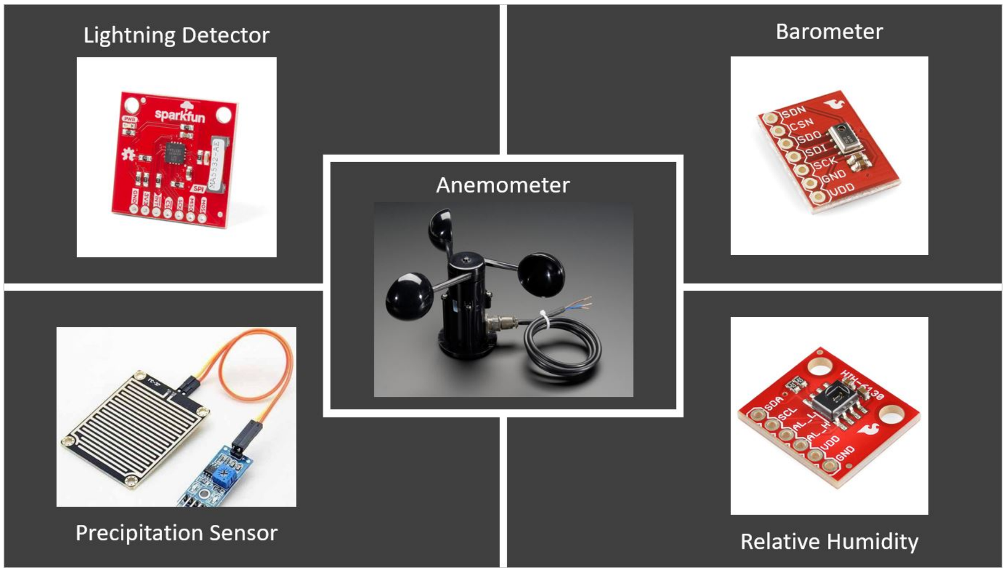


Figure 54. On-Board Sensor Suite Examples

A.1.2. Radar-Based Weather Communication by Radio

The Radar-Based Weather Communication by Radio method employs the use of Doppler Radar through web applications to determine situations in which the WRAITH system should enter safe mode. The determination software will be run at an off-site ground station where internet connectivity will be present. This software will import radar overlays from online sources and will determine if the storm system poses a threat to WRAITH based upon WRAITH's GPS location. If the environmental conditions are deemed dangerous for operation, the ground station will relay a safe mode command to the WRAITH unit via radio communications. This method benefits from increased lead time for the safe mode feature, however, this does not provide any on-site feedback for the Weather Determination system.

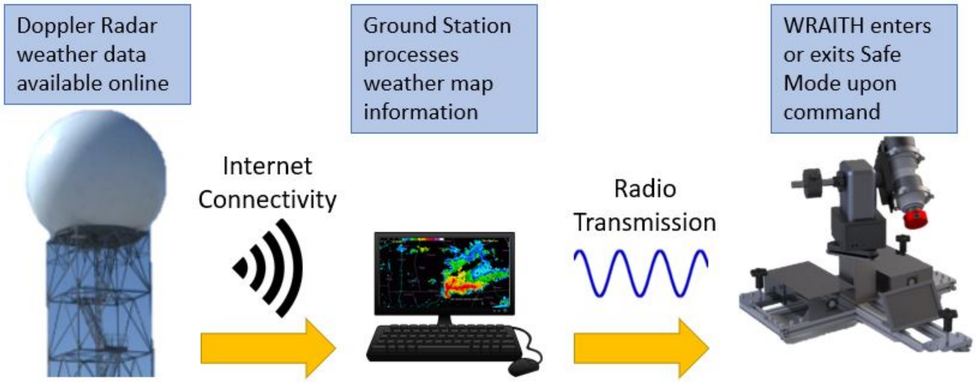


Figure 55. Radar Method Schematic

A.1.3. Radar and On-Board Sensor Suite Combination

The Radar and On-Board Sensor Suite Combination method incorporates the two aforementioned methods. This method would employ the use of the sensors outlined in Fig. 54 with the radar predictability method from Fig. 55. As previously discussed, the sensor suite would allow the system to have an immediate fail-safe while also utilizing sensors to predict and anticipate adverse conditions. On its own, the radar method allows the system to access commands derived from up-to-date forecast models which would increase the lead time for the weather systems. Therefore, this method benefits from the predictability of the radar method and the risk decreasing fail-safes of the sensor suite method.

A.1.4. Weather Detection Method Summary

The following table, Table 21, provides the higher level pros and cons of each proposed method. These pros and cons, as well as many other factors, were considered when conducting the trade study in the following section.

Table 21. Weather Detection Methods Pros and Cons

Option	Pros	Cons
On-Board Sensor Suite	• Precipitation Sensor Fail-Safe • Multiple sensors decrease false-positives • Provides conditions for exact location	• Limited capabilities to predict conditions • No ability to communicate from ground station • Prediction model based solely on conditions
Radar-Based Weather Communication by Radio	• Increased lead time • Off-Site ability to send Safe Mode commands	• No Fail-Safe features in place • Reliant upon steady radio communication • Generalized conditions for GPS location • No on-site feedback for WRAITH system
Radar and Sensor Suite Combination	• Precipitation Sensor Fail-Safe • Multiple Sensors decrease false-positives • Increased lead time • Off-Site	• More Complex to integrate multiple methods • Higher Costs • More space consuming than other methods

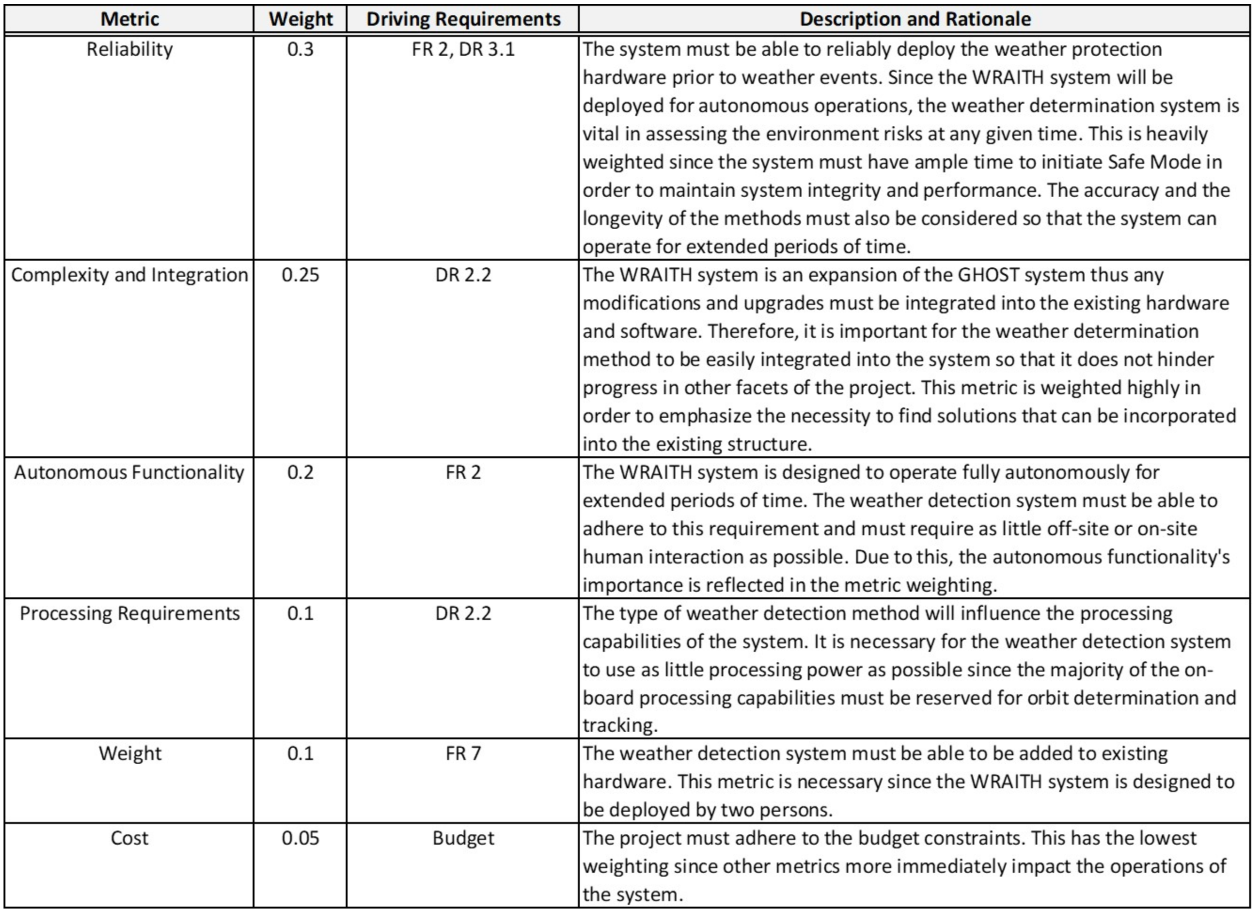
A.1.5. Trade Study

A trade study was performed using the metrics below in order to choose the necessary Weather Detection Method for the WRAITH system. The metrics in this case were Reliability, Integration Complexity, Autonomous Functionality, Processing Requirements, Weight, and the Costs of the method. The weights and explanations for these metrics are given in Table 22.

The weights for these metrics were decided based on the most critical aspects of the system. These numerical weights then went into the algorithm for the overall trade study. The methodology behind assigning these weightings is described below:

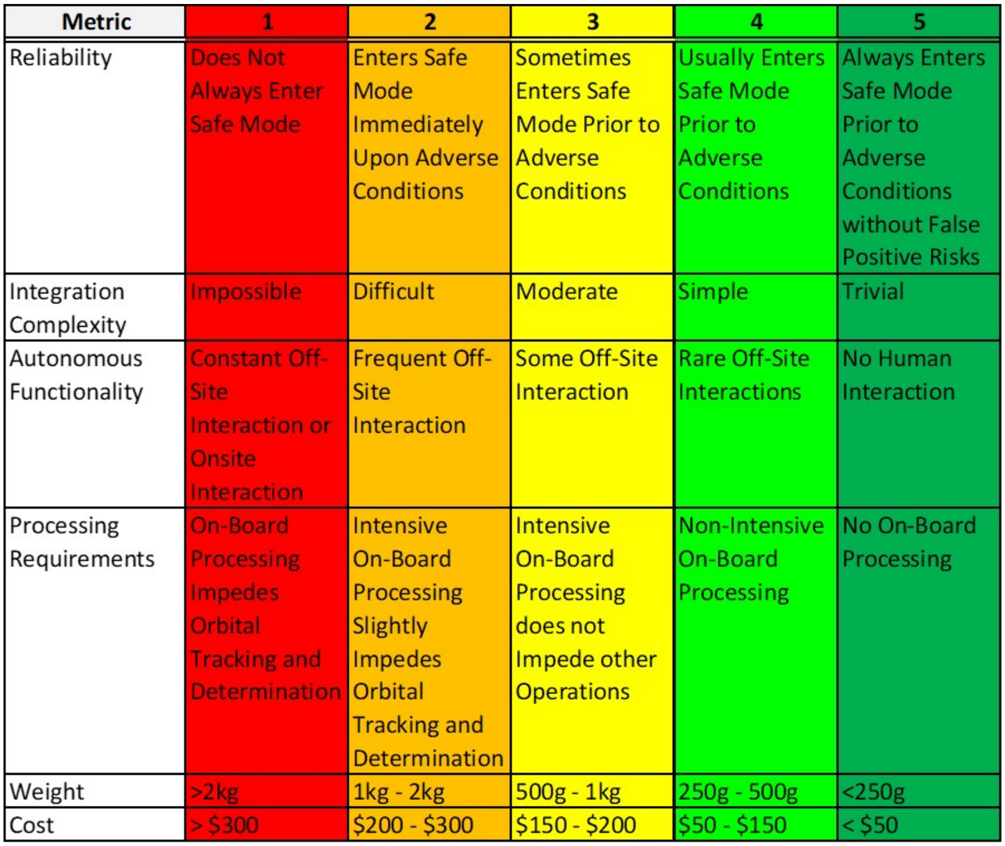
- Reliability: If the weather detection system fails, the active weather protection system will be incapacitated and thus the safety of the system will be compromised greatly. Since the safety of the hardware is an important aspect of the system, reliability was given the highest weight of 0.3.
- Complexity and Integration: Since the weather detection system is going to be implemented into the previous GHOST system, it's critical that the complexity is as low as possible and the integration is as simple as possible. This was deemed critical to the success of the system and thus was given a numerical weight of 0.25.
- Autonomous Functionality: WRAITH must be fully autonomous for a 12-hour deployment, so the weather detection system must require as little human interaction as possible. Since this is a direct functional requirement for WRAITH, it was assigned a numerical weight of 0.2.
- Processing Requirements: The weather detection software must not interfere with the on-board software in terms of processing power, however the weather detection software is expected to be simple so this was given a numerical weight of 0.1.
- Weight: WRAITH must be deployed by two operators so the system cannot be too heavy, however since most sensors are lightweight, a low numerical weighting of 0.1 was given to the weight of the weather detection system.
- Cost: Due to the team's predicted low costs for other aspects of the system and the higher importance of the other metrics to the operations of the system, the cost was given the lowest weight of 0.05.

Table 22. Weather Determination Metric Rationale and Descriptions



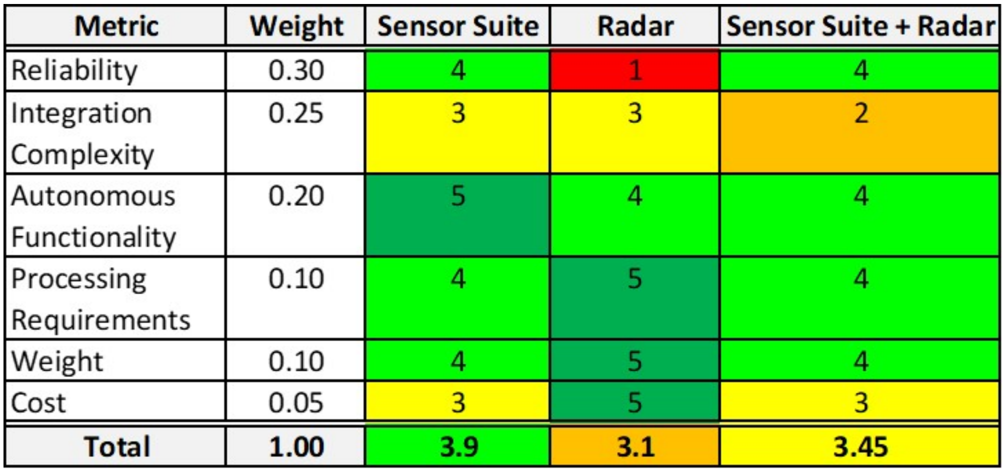
Metric	Weight	Driving Requirements	Description and Rationale
Reliability	0.3	FR 2, DR 3.1	The system must be able to reliably deploy the weather protection hardware prior to weather events. Since the WRAITH system will be deployed for autonomous operations, the weather determination system is vital in assessing the environment risks at any given time. This is heavily weighted since the system must have ample time to initiate Safe Mode in order to maintain system integrity and performance. The accuracy and the longevity of the methods must also be considered so that the system can operate for extended periods of time.
Complexity and Integration	0.25	DR 2.2	The WRAITH system is an expansion of the GHOST system thus any modifications and upgrades must be integrated into the existing hardware and software. Therefore, it is important for the weather determination method to be easily integrated into the system so that it does not hinder progress in other facets of the project. This metric is weighted highly in order to emphasize the necessity to find solutions that can be incorporated into the existing structure.
Autonomous Functionality	0.2	FR 2	The WRAITH system is designed to operate fully autonomously for extended periods of time. The weather detection system must be able to adhere to this requirement and must require as little off-site or on-site human interaction as possible. Due to this, the autonomous functionality's importance is reflected in the metric weighting.
Processing Requirements	0.1	DR 2.2	The type of weather detection method will influence the processing capabilities of the system. It is necessary for the weather detection system to use as little processing power as possible since the majority of the on-board processing capabilities must be reserved for orbit determination and tracking.
Weight	0.1	FR 7	The weather detection system must be able to be added to existing hardware. This metric is necessary since the WRAITH system is designed to be deployed by two persons.
Cost	0.05	Budget	The project must adhere to the budget constraints. This has the lowest weighting since other metrics more immediately impact the operations of the system.

Table 23. Weather Detection Metric Values



Metric	1	2	3	4	5
Reliability	Does Not Always Enter Safe Mode	Enters Safe Mode Immediately Upon Adverse Conditions	Sometimes Enters Safe Mode Prior to Adverse Conditions	Usually Enters Safe Mode Prior to Adverse Conditions	Always Enters Safe Mode Prior to Adverse Conditions without False Positive Risks
Integration Complexity	Impossible	Difficult	Moderate	Simple	Trivial
Autonomous Functionality	Constant Off-Site Interaction or Onsite Interaction	Frequent Off-Site Interaction	Some Off-Site Interaction	Rare Off-Site Interactions	No Human Interaction
Processing Requirements	On-Board Processing Impedes Orbital Tracking and Determination	Intensive On-Board Processing Slightly Impedes Orbital Tracking and Determination	Intensive On-Board Processing does not Impede other Operations	Non-Intensive On-Board Processing	No On-Board Processing
Weight	>2kg	1kg - 2kg	500g - 1kg	250g - 500g	<250g
Cost	> $300	$200 - $300	$150 - $200	$50 - $150	< $50

Table 24. Weather Detection Trade Study Results



Metric	Weight	Sensor Suite	Radar	Sensor Suite + Radar
Reliability	0.30	4	1	4
Integration Complexity	0.25	3	3	2
Autonomous Functionality	0.20	5	4	4
Processing Requirements	0.10	4	5	4
Weight	0.10	4	5	4
Cost	0.05	3	5	3
Total	**1.00**	**3.9**	**3.1**	**3.45**

For this trade study, three options were considered: radar detection only, sensor suite detection only, and a combination of radar and sensor suite detection. To assign values to each option in this trade study, a table of metric values was constructed using 1 as the worst case and 5 as the most ideal case. These metrics can be seen in Table 23, which was used to assign values to each option. Using these metric values and the weights assigned to each metric, the highest scoring option can be determined. The final trade study results can be seen in Table 24 and the sections below outline why each metric received its score for each design method considered.

As can be seen in Table 24, the *Total* row shows how well each method performed in the trade study. The Radar Only option scored the lowest with 3.1 while the Sensor Suite Only option scored the highest with 3.9. The Sensor Suite Only method was deemed the best option for the weather detection system as it is the easiest to implement autonomously, as well as being reliable and simple for the software behind it. It is important to note that the sensor suite option provides the best autonomous functionality and reliability at the cost of slightly lower processing requirement, weight, and cost values. For the weather detection system, the team traded a lower weight, lower cost, and better processing requirements for increased autonomous functionality and reliability, which were deemed more critical to the success of WRAITH.

A.1.6. Sensor Suite Only Metric Assignments

- Reliability - 4: The sensors will usually be able to detect adverse weather before it reaches the deployment location due to the variety of sensors on-board. However, it will not always be able to enter safe mode before these conditions arrive and without any false positives[8] that may result from having multiple sensors.
- Integration Complexity - 3: Integrating sensors with the hardware is not expected to be difficult, however integrating them with the software is expected to be moderately challenging but doable.
- Autonomous Functionality - 5: The sensors should be free of any human interaction after deployment, so they should be able to detect adverse conditions and enter safe mode independently of the operator.
- Processing Requirements - 4: There will be some minimal on-board processing with the sensors as some of the researched sensors do calculations on-board, however nothing that is expected to hinder the other software operations.
- Weight - 4: The sensors are all lightweight and any additional integration materials such as wiring are expected to be minimal in weight as well, leaving an expected total weight between 250 and 500 grams.
- Cost - 3: All of the sensors combined with any estimated integration materials are expected to cost between $150 and $200.

A.1.7. Radar Only Metric Assignments

- Reliability - 1: Only having radar means that if it's inaccurate[8] (there are many reasons for inaccurate radar) or there is a communication issue between the ground station and WRAITH, there could be adverse weather at the deployment location and WRAITH would never enter safe mode, potentially resulting in damaged hardware.
- Integration Complexity - 3: Integrating this with the hardware is not expected to be difficult since the radio is a small component, however integrating with the software and having it communicate with a ground station is expected to be moderately difficult.
- Autonomous Functionality - 4: The radar should not need any off-site human interactions except in the case of an extreme weather override, during which the operator will remotely force WRAITH to enter safe mode.
- Processing Requirements - 5: The radar is not expected to have any on-board processing since the ground station would be deciding whether or not safe mode needed to be entered.
- Weight - 5: The radio that will be on-board is small and the ground station will not factor into the weight of the system since it's off-site, leading to a total weight of less than 250 grams.
- Cost - 5: The on-board radio is fairly inexpensive, as well as any estimated integration materials. The ground station will consist of an existing computer, so the total cost is expected to be less than $50.

A.1.8. Sensor Suite and Radar Combination Metric Assignments

- Reliability - 4: Since the radar would act as a fail-safe[8] rather than adding additional lead time to detect adverse weather, the combination has the same metric value as the sensor suite alone.
- Integration Complexity - 2: Integration for both the sensors and the radar becomes more difficult in this option because each sensor and the radio all need to be integrated with the existing hardware and software, as well as with each other, which is expected to be a more demanding task.

- Autonomous Functionality - 4: The combination option received the metric value that was assigned to the radar only option because the sensors are expected to operate without human interaction, so the addition of the radio would drop that to the case where there may be occasional human interaction in the emergency shut down situation.
- Processing Requirements - 4: The combination option received the metric value that was assigned to the sensor suite only option because while the radar is expected to have no on-board processing, the sensors are expected to have some minimal on-board processing, resulting in a cumulative metric value of 4.
- Weight - 4: Neither the sensors nor the on-board radio weigh much, so the expected weight for both of them combined is expected to be between 250 and 500 grams.
- Cost - 3: The sensor suite alone is expected to cost around $150, with the radio roughly costing an additional $20 depending on the specific type, so the total cost is expected to fall somewhere between $150 to $200.

A.2. System-to-Operator Communication

Shannon Chott, Noah Crisler, Trevor Willson

As identified in Design Requirement 3.2, the WRAITH system must be capable of receiving operational start-up and shutdown commands from on off-site operator. This criteria made it necessary that the overall system incorporated a means of this communication. The succeeding sections explore various options and perform the proper trade studies to determine the most favorable option that meets the purpose of this project. These options include HAM radio, satellite radio, and utilizing the cellular network.

A.2.1. HAM Radio

The HAM radio communication system would allow for two way line-of-sight communication over a range of 1-2 miles. Both ends of the system would need a waterproof HAM radio transceiver. Each transceiver would connect to a TNC-X modem which demodulates the incoming/outgoing data signal and routes it to/from an open source software such as Winlink. The open source software is used to encode the text files and other telemetry into the desired format of the transceiver. A rough estimate of the combined system cost is $800. The HAM radio transceivers are additionally 100% self-supported, meaning they would not add any load to the battery.

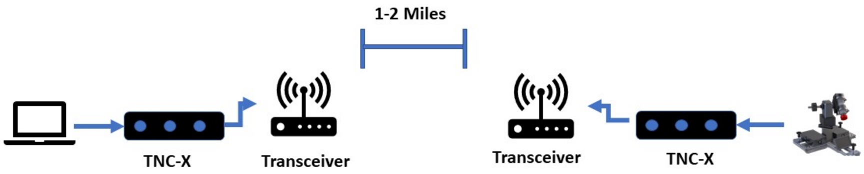


Figure 56. HAM Radio Concept

A.2.2. Satellite Radio

Using satellite radio would enable the WRAITH system to pursue two-way communication. This system would also provide coverage for all of the continental United States, which would allow the WRAITH system to be deployed anywhere in this zone. Several different satellite networks exist that could provide this service, namely Globalstar and the Iridium network. Both systems require a subscription service ranging from $50 to $100 per month depending on the desired number of minutes. These satellite phones would interface with WRAITH via an RS232 connection, Bluetooth, or micro-USB. Additionally, satellite phones are sold off-the-shelf as being water-resistant, adding to its passive protection.

A.2.3. Cellular Network

The Cellular Network Communication method would allow for two-way communication to be incorporated into the WRAITH system. This communication method would utilize 4G US cellular phone infrastructure to send and receive signals anywhere in the US that is covered by cell service. Fig. 57[12] provides an example of the coverage map for 4G communication in Colorado. The integration of this system is relatively straightforward but will not be trivial since the cellular hardware must be integrated into the existing hardware through cellular breakout boards. An example of the cellular hardware[9] is given in Fig. 58. In addition, this system must be able to relay commands for the environmental protection system and must be able to transmit TLE data to the system, thus the software integration is moderately difficult. A downfall of this method is that a monthly data subscription must be purchased for less than $10 a month since this method sends and receives data over 4G networks.[10] [11]

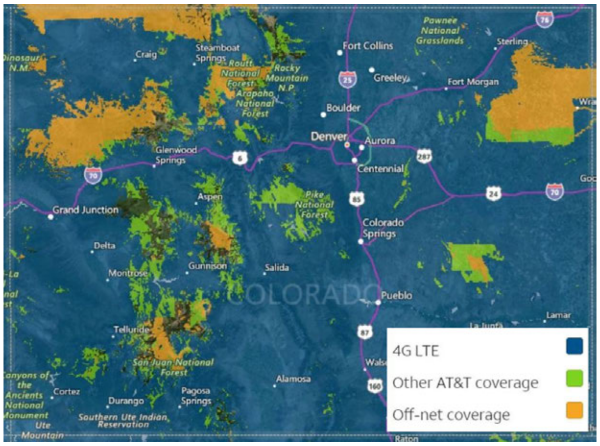


Figure 57. Example 4G Cellular Service Map

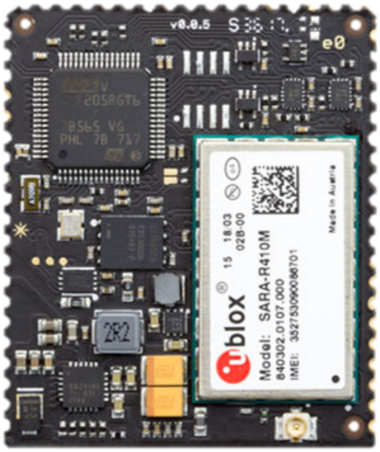


Figure 58. Example Cellular Communication Hardware

A.2.4. Radio Methods Summary

Table 25 provides the pros and cons of each radio communication method. These factors were then considered in the analysis of the trade study for the Radio Communication system.

Table 25. Radio Communication Method Pros and Cons

Option	Pros	Cons
2G/3G Cellular	• Cheap • Network Coverage • Compact System	• Software Intergration • Non-waterproof • Monthly Subscription
Satellite	• Covers All of Continental US • Water Resistent • Moderate Integration Complexity	• Extremely Costly • Monthly Subscription
Pocket/HAM Radio	• No Monthly Subscription • Water Resistent	• Costly • Complex Integration • Line of Sight Communication

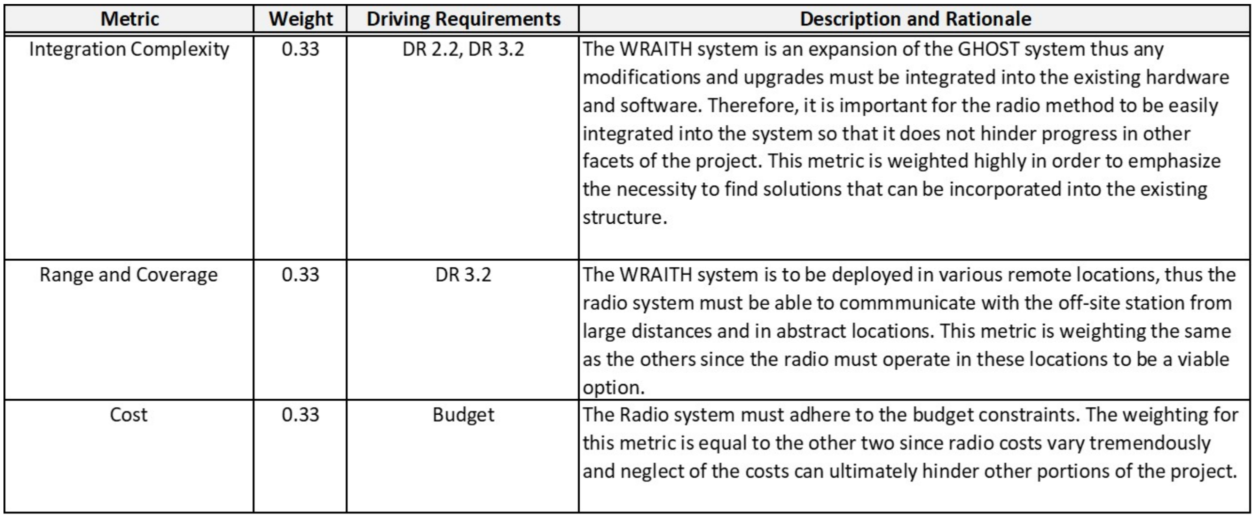
A.2.5. Trade Study

A trade study was performed using the metrics below in order to choose the necessary radio system for WRAITH. The metrics in this case are integration complexity, range and coverage, and cost of the method. The weights and explanations for these metrics are given in Table 26.

The weights for these metrics were decided based on the most critical aspects of the system. These numerical weights then went into the algorithm for the overall trade study. The methodology behind assigning these weightings is described below:

- Integration Complexity: The communication method must be incorporated into the existing GHOST system, thus the integration complexity should be as simple as possible. This characteristic was deemed critical for the success of the system, therefore it was assigned a weight of 0.33.
- Range and Coverage: Since the WRAITH system must be placed far from major light sources the range and coverage of the communication method is vital for maintaining communication with the off-site operator. Similar to the other metrics, this metric was given a weight of 0.33.
- Cost: Although the radio communication is vital to fulfill the requirements of the system, the cost cannot be neglected since component costs can vary a large amount and exceed the budget for a single component. Therefore, the cost was given equal weight as the other metrics at 0.33.

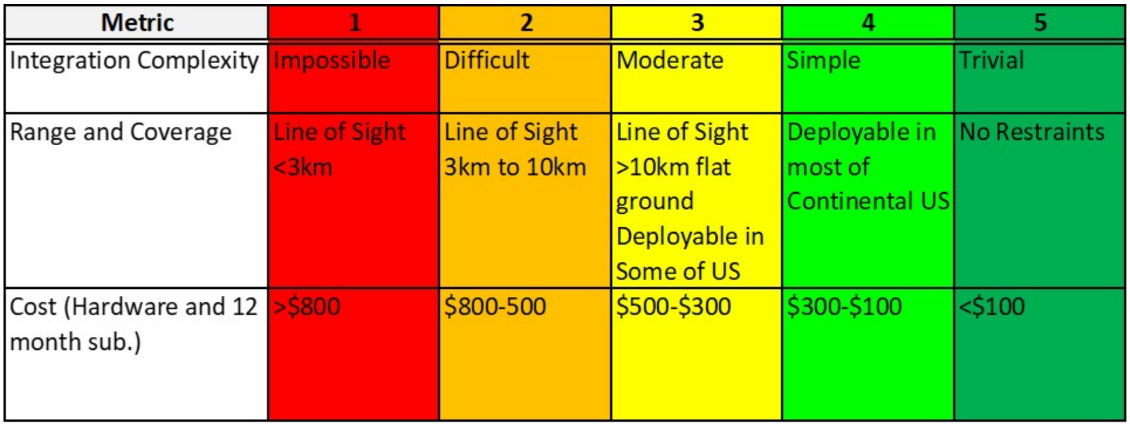
Table 26. Radio Communication Metric Rationale and Descriptions



Metric	Weight	Driving Requirements	Description and Rationale
Integration Complexity	0.33	DR 2.2, DR 3.2	The WRAITH system is an expansion of the GHOST system thus any modifications and upgrades must be integrated into the existing hardware and software. Therefore, it is important for the radio method to be easily integrated into the system so that it does not hinder progress in other facets of the project. This metric is weighted highly in order to emphasize the necessity to find solutions that can be incorporated into the existing structure.
Range and Coverage	0.33	DR 3.2	The WRAITH system is to be deployed in various remote locations, thus the radio system must be able to commmunicate with the off-site station from large distances and in abstract locations. This metric is weighting the same as the others since the radio must operate in these locations to be a viable option.
Cost	0.33	Budget	The Radio system must adhere to the budget constraints. The weighting for this metric is equal to the other two since radio costs vary tremendously and neglect of the costs can ultimately hinder other portions of the project.

For this trade study, three options were considered: pocket/ham radio, satellite, and 4G cellular. To assign values to each option in this trade study, a table of metric values was constructed using 1 as the worst case and 5 as the most ideal case. These metrics can be seen in Table 27, which was used to assign values to each option. Using these metric values and the weights assigned to each metric, the highest scoring option will be the most favorable. The metric assignments for each individual option are further explained in the following sections. The final trade study results can be seen in Table 28.

Table 27. Radio Communication Metric Values



Metric	1	2	3	4	5
Integration Complexity	Impossible	Difficult	Moderate	Simple	Trivial
Range and Coverage	Line of Sight <3km	Line of Sight 3km to 10km	Line of Sight >10km flat ground Deployable in Some of US	Deployable in most of Continental US	No Restraints
Cost (Hardware and 12 month sub.)	>$800	$800-500	$500-$300	$300-$100	<$100

Table 28. Radio Communication Trade Study

Metric	Weight	Pocket/Ham Radio	Satellite	Cellular
Intergration Complexity	0.33	2	2	3
Range and Coverage	0.33	3	5	4
Cost	0.33	2	1	4
Total	**1.00**	**2.33**	**2.66**	**3.67**

As previously mentioned, Table 28 provides the evaluation of the communication options for the system. The table breaks down each option into individual characteristics which were assigned a numerical value to represent the value that option provides the team. As seen in the table, the cellular communication option is the most advantageous for the purposes of this project. This particular option provides an exceptional cost value since the expected cost of the hardware and data subscription is far less than the expected costs of the other two options. In addition, the integration cellular communication option is less complex than that of both the Pocket/HAM Radio and the Satellite communication method. It is important to note that the larger range and coverage of the satellite communication was traded for the simpler integration and lower costs of the cellular method.

A.2.6. Pocket/HAM Radio Metric Assignments

- Integration Complexity - 2: The HAM/Pocket radio system requires getting multiple software programs, a TNC-X modem, and a transceiver to all work together. Once the individual system was working, it would then need to be integrated further into the existing code structure. The large number of individual parts that this design calls for causes it to have a low score.
- Range and Coverage - 3: HAM/Pocket radio is limited to only line-of-sight communications (assuming the system does not have further signal repeaters).
- Cost - 2 : With a rough cost estimate of $800, the need for two of each part causes the system to score very low in the cost category.

A.2.7. Satellite Communication Metric Assignments

- Integration Complexity - 2: The satellite system would require a software system to be developed to take texts from the satellite phone and convert those into commands. WRAITH would also require some form of physical connection to the satellite phone via an RS232 connection, Bluetooth, or micro-USB. As such, the complexity was rated as a 2.
- Range and Coverage - 5: Given that both the Iridium and Globalstar network completely cover the continental United States, range/coverage was rated as a 5.
- Cost - 1: The satellite system would cost a minimum of $850 with limited minutes on the subscription and the hardware. As this value approaches nearly 20% of the budget, it was rated as a 1.

A.2.8. Cellular Network Communication Metric Assignments

- Integration Complexity - 3: The cellular network communication requires a separate breakout board to be integrated into the system. In addition, the method would require software integration with the existing system in order to communicate the received data files. This integration is expected to be moderately difficult.
- Range and Coverage - 4: The cellular method utilizes the already in-place 4G infrastructure (see Fig. 57) to send and receive information.
- Cost - 4: The individual hardware components range from $50 to $100, while the monthly data subscriptions range from $3 to $10.

B. Standard Operational Conditions

University of Colorado
Department of Aerospace Engineering Sciences
WRAITH

Standard Operational Conditions

The purpose of this document is to define the Standard Environmental and Deployment Conditions for the WRAITH system. The definitions will be split between Environmental and Topographical characteristics. The Environmental characteristics will focus solely on the changing weather conditions. The Topographical characteristics will focus on deployment area terrain and climate.

Environmental

1. Temperature Range shall be between -7°C and 32°C.
2. Wind Speed shall be less than $8\frac{m}{s}$.
3. Lightning shall be at least 15km from system.
4. Relative Humidity shall be below 80%

Topographical

1. Average Relative Humidity shall be below 50%.
2. Maximum Ground Grade of 10°.
3. Minimum Sky Quality Meter (SQM) value of $20\,\frac{mag}{arcsec^2}$.

C. Severe Weather Conditions

WRAITH Severe Weather Conditions

The weather conditions will be referenced both from the **National Weather Service** and **Clear Dark Sky** websites for the specific deployment location in the days and hours leading up to the proposed deployment time. The bounds below will be used as a guideline for deciding whether WRAITH will deploy or not. In addition to these bounds, general observation will also be utilized by the operators to make the final decision. If the weather conditions are forecasted to be dangerous to the WRAITH hardware or if the operators feel that WRAITH should not deploy, the deployment will be rescheduled to another night. The operators can make this decision at any point leading up to the deployment or during set up, regardless of what the weather forecasts are reporting. This is important because the safety of the hardware and the operators themselves is more important than obtaining images.

Severe Weather Limits

1. Chance of precipitation greater than or equal to **40%**
2. Greater than **50%** cloud cover at the start of the deployment or greater than **40%** cloud cover predicted for 6 or more hours of the 12-hour deployment
3. Sustained winds predicted greater than or equal to **17 mph (~7.5m/s)**
4. Low temperatures of **5°F** or less

Reasoning

1. The chance of precipitation that forecasters use is a combination of factors. The equation is as follows:

   $$PoP = C\ A$$

   This equation states that the “Probability of Precipitation” is equal to C times A. C is the probability that precipitation will occur somewhere in the area while A is the percent of area that will receive measurable precipitation if it occurs at all. These two factors combine into the percent chance of precipitation that is commonly seen in weather forecasts. While WRAITH has an autonomous weather detection and protection system, it is still important to ensure its safety as much as possible. As such, WRAITH will not be deployed if there is a 40% chance of precipitation or higher.
2. Clouds are an incredibly dynamic component of atmospheric science, that is, they are constantly in motion and changing in thickness and height. As such, if there is too much cloud cover at the start of the deployment, the iOptron (the stage) will have severe difficulty with the calibration, which is critical to the success of the deployment. Along those same lines, if there is too much cloud cover during the deployment, then WRAITH should be in safe mode anyway and it will not be able to image object passes.
3. WRAITH contains a lens that is sensitive to dust and other particulate matter and a camera that will not be able to obtain quality images of the object passes if WRAITH is shaking due to excessive wind. As such, if there are predicted sustained winds at about

D. Stage ASCII Commanding Language

iOptron® Mount RS-232 Command Language 2014

Version 2.0

August 8th, 2014

Abbreviations used:

YYMMDD	YY: last two digits of the year, year are assumed to be 21st Century MM: month, DD: day of the month
HHMMSS	HH: hour, MM: minute, SS: second
s	+ or – sign, “+” cannot be replaced by any other characters
MMM	minutes
SSSSSS	arc-seconds
TTTTTTTT(T)	0.01 arc-seconds
XXXXX(XXX)	milliseconds
nnn or longer	digits

All digits above including leading zeroes to match the format of the each command.

Currently the document applies to the following products:
CEM60 (Firmware 140807 and later)
CEM60-EC (Firmware 140807 and later)
iEQ45 Pro (Firmware 140807 and later)
iEQ45 Pro AA (Firmware 140807 and later)

General Information

Command: “:GAS#”
Response: “nnnnnn#”
The 1st digit stands for GPS status: 0 means GPS off, 1 means GPS on, 2 means GPS data extracted correctly.
The 2nd digit stands for system status: 0 means stopped (not at zero position), 1 means tracking with PEC disabled, 2 means slewing, 3 means guiding, 4 means meridian flipping, 5 means tracking with PEC enabled (only for non-encoder edition), 6 means parked, 7 means stopped at zero position (home position).
The 3rd digit stands for tracking rates: 0 means sidereal rate, 1 means lunar rate, 2 means solar rate, 3 means King rate, 4 means custom rate.
The 4th digit stands for moving speed by arrow button or moving command: 1 means 1x sidereal tracking rate, 2 means 2x, 3 means 8x, 4 means 16x, 5 means 64x, 6 means 128x, 7 means 256x, 8 means 512x, 9 means maximum speed. Currently, the maximum speed of CEM60 (-EC) is 900x, the maximum speed of iEQ45 Pro (/AA) is 1400x.
The 5th digit stands for time source: 1 means RS-232 port, 2 means hand controller, 3 means GPS.
The 6th digit stands for hemisphere: 0 means Southern Hemisphere, 1 means Northern Hemisphere.

Command: “:GLT#”
Response: “sMMMYYMMDDHHMMSS#”
This command include a sign and 16 digits, and gets time related data.
The sign and first 3 digits stands for the minutes of UTC offset (time zone). Note: The Daylight Saving Time does not affect this offset.
The 4th digit stands for the Daylight Saving Time, 0 for Daylight Saving Time not observed, 1 for Daylight Saving Time observed.
The 5th to 10th digits stands for local Date.
The 11th to 16th digits stands for local Time in 24 hours format.

Command: “:GEC#”
Response: “sTTTTTTTTXXXXXXXX#”
This command include a sign and 16 digits, and gets right ascension and declination.
The sign and first 8 digits stands for current declination.
The last 8 digits stands for current right ascension.

Command: “:GAC#”
Response: “sTTTTTTTTTTTTTTTTT#”
This command include a sign and 17 digits, and gets altitude and azimuth.
The sign and first 8 digits stands for current altitude.
The last 9 digits stands for current azimuth.

Command: “:Gg#”
Response: “sSSSSSS#”
This command gets the current longitude. Note: East is positive.

Command: “:Gt#”
Response: “sSSSSSS#”
This command gets the current latitude. Note: North is positive.

Command: “:AG#”
Response: “nnn#”
This command returns the guiding rate n.nn * sidereal rate. The range of n.nn is [0.10, 0.90].

Data entered with the following 8 commands will be “remembered” through a power cycle and automatically re-applied on the next power up.

Command: “:SGsMMM#”
Response: “1”
Sets the time zone offset from UTC (Daylight Saving Time will not affect this value). The offset can only be entered in the range of [-720, +780] minutes.

Command: “:SDS0#” “:SDS1#”
Response: “1”
Sets the status of Daylight Saving Time. “:SDS1#” means Daylight Saving Time observed, “:SDS0#” means Daylight Saving Time not observed.

Command: “:SCYYMMDD#”
Response: “1”
Sets the current Local Date.

Command: “:SLHHMMSS#”
Response: “1”
Sets the current Local Time. The time can only be entered in the range of 00:00:00 to 23:59:59.

Command: “:SgsSSSSSS#”
Response: “1”
Sets the current longitude. The longitude can only be entered in the range of [-180, 180] degree. Note: East is positive.

Command: “:StsSSSSSS#”
Response: “1”
Sets the current latitude. The latitude can only be entered in the range of [-90, 90] degree. Note: North is positive.

Command: “:RGnnn#”
Response: “1”
Sets the guiding rate n.nn * sidereal rate. nnn can only be entered in the range of [10, 90].

Command: “:SHE0#” “:SHE1#”
Response: “1”
Sets the hemisphere. 0 means Southern Hemisphere, 1 means Northern Hemisphere.

Mount Motion

Command: “:MS#”
Response: “1” if command accepted,
“0” The desired object is below 0 degrees altitude.
Slew to the most recently defined right ascension and declination coordinates or most recently defined altitude and azimuth coordinates (only works with Alt-Azi Mount). If the object is below the horizon, this will be stated, and no slewing will occur.

Command: “:Q#”
Response: “1”
This command will stop slewing only. Tracking and moving by arrow keys will not be affected.

Command: “:MnXXXXX#” “:MeXXXXX#” “:MsXXXXX#” “:MwXXXXX#”
Response: (none)
Command motion for XXXXX milliseconds in the direction specified at the currently selected guide rate. The range of XXXXX is [0, 99999].

Command: “:ST0#” “:ST1#”
Response: “1”
These command sets tracking state. “:ST0#” indicates stop tracking, “:ST1#” indicates start tracking.

Command: “:RT0#” “:RT1#” “:RT2#” “:RT3#” “:RT4#”
Response: “1”
This command selects the tracking rate. It selects sidereal (:RT0#), lunar (:RT1#), solar (:RT2#), King (:RT3#), or custom (:RT4#). The sidereal rate is assumed as a default by the next power up. This command has no effect on the use of the N-S-E-W buttons.

Command: “:MP1#”
Response: “1” if command accepted,
“0” The desired object is below 0 degrees altitude.
Park to the most recently defined right ascension and declination coordinates or most recently defined altitude and azimuth coordinates (only works with Alt-Azi Mount). If the target is below the horizon, this command will have no effect. In parked mode, the mount cannot slew, track, guide or perform any movement unless a un-park command is issued. If you parked the mount and powered it off, at the beginning of the next power up, the mount will un-park automatically.

Command: “:MP0#”
Response: “1”
This command un-parks the mount. If the mount is already un-parked, the command will have no effect.

Command: “:MH#”
Response: “1”
This command will slew to the zero position (home position) immediately.

Command: “:MSH#”
Response: “1”
This command will auto search the real zero position (home position) immediately. This command ignores any current zero position, and designed to be a safe operation in any cases. Currently, only CEM60 and CEM60-EC support this feature.

Command: “:SRn#”
Response: “1”
Sets the moving rate used for the N-S-E-W buttons. For n, specify an integer from 1 to 9. 1 stands for 1x sidereal tracking rate, 2 stands for 2x, 3 stands for 8x, 4 stands for 16x, 5 stands for 64x, 6 stands for 128x, 7 stands for 256x, 8 stands for 512x, 9 stands for maximum speed. Currently, the maximum speed of CEM60 (-EC) is 900x, the maximum speed of iEQ45 Pro (/AA) is 1400x. 64x is assumed as a default by the next power up.

Command: “:RRsnn.nnnn#”
Response: “1”
“Custom Tracking Rate” (“:RT4#”) must be selected before this command to take effect.
The command selects the tracking rate in the RA axis to nn.nnnn. The rate added to the standard sidereal rate and can be positive or negative. Currently the value is limited to the range of [-0.0100, +0.0100]. Data entered with this command will be remembered through a power cycle and automatically re-applied on the next power up.

Command: “:RDsnn.nnnn#”
Response: “1”
“Custom Tracking Rate” (“:RT4#”) must be selected before this command to take effect.
The command selects the tracking rate in the Dec axis to nn.nnnn. Currently this command has not been implemented.

Command: “:mn#” “:me#” “:ms#” “:mw#”
Response: (none)
These commands have identical function as arrow key pressed. They will move mounts to N-E-S-W direction at specified speed (may change by “:SRn#”). The mount will keep moving until a “:qR#”, “:qD#”, and/or “:q#” sent.

Command: “:q#”
Response: “1”
This command will stop moving by arrow keys or “:mn#”, “:me#”, “:ms#”, “:mw#” command. Slewing and tracking will not be affected.

Command: “:qR#”
Response: “1”
This command will stop moving by left and right arrow keys or “:me#”, “:mw#” command. Slewing and tracking will not be affected.

Command: “:qD#”
Response: “1”
This commands will stop moving by up and down arrow keys or “:mn#”, “:ms#” command. Slewing and tracking will not be affected.

Position

Command: “:CM#”
Response: “1”
Calibrate mount (Sync). In equatorial mounts, the most recently defined right ascension and declination become the commanded right ascension and declination respectively. In Alt-Azi mounts, the most recently defined altitude and azimuth become the commanded altitude and azimuth. This command assumes that the mount has been manually positioned on the proper pier side for the calibration object. This command is ignored if slewing is in progress. This command should be used for initial calibration. It should not be used after the mount has been tracking unless it is known that it has not tracked across the meridian.

Command: “:SrXXXXXXXX#”
Response: “1”
Defines the commanded right ascension, RA. Slew, calibrate and park commands operate on the most recently defined right ascension.

Command: “:SdsTTTTTTTT#”
Response: “1”
Defines the commanded declination, Dec. Slew, calibrate and park commands operate on the most recently defined declination.

Command: “:SasTTTTTTTT#”
Response: “1”
This command only works with Alt-Azi Mounts. Defines the commanded altitude, Alt. Move, calibrate and park commands operate on the most recently defined altitude.

Command: “:SzTTTTTTTTT#”
Response: “1”
This command only works with Alt-Azi Mounts. Defines the commanded azimuth, Azi. Move, calibrate and park commands operate on the most recently defined azimuth.

Command: “:SZP#”
Response: “1”
This command will set current position as zero position.

Miscellaneous

Command: “:FW1#”
Response: “YYMMDDYYMMDD#”
Gets the date of the mainboard’s and the hand controller’s firmware. The first “YYMMDD” indicates the date of the mainboard’s firmware, the second “YYMMDD” indicates the date of the hand controller’s firmware.

Command: “:FW2#”
Response: “YYMMDDYYMMDD#”
Gets the date of the RA motor board’s and the Dec motor board’s firmware. The first “YYMMDD” indicates the date of the RA motor board’s firmware, the second “YYMMDD” indicates the date of the Dec motor board’s firmware.

Command: “:V#”
Response: “V1.00#”
This command is the first initialization command of iOptron® series products.

Command: “:MountInfo#”
Response: “0060” “0061” “0045” “0046”
This command gets the mount type. “0060” means CEM60 without high precision encoder, “0061” means CEM60-EC, “0045” means iEQ45 Pro EQ Mode, “0046” means iEQ45 Pro AA Mode.

Miscellaneous information

The command set is written in ASCII character format and can be used to write your own programs. All commands are case sensitive.

Initialization sequence

In order to properly initialize the mount with your software, you must issue the following commands when you establish your link:
:V#
:MountInfo#

RS-232 Port Settings

Baud Rate:	9600
Parity:	none
Data bits:	8
Flow Control:	none (does not support Xon/Xoff or hardware flow control)
Start Bits:	1
Stop Bits:	1

Version History

1.0 July 4th, 2014
Initial release.

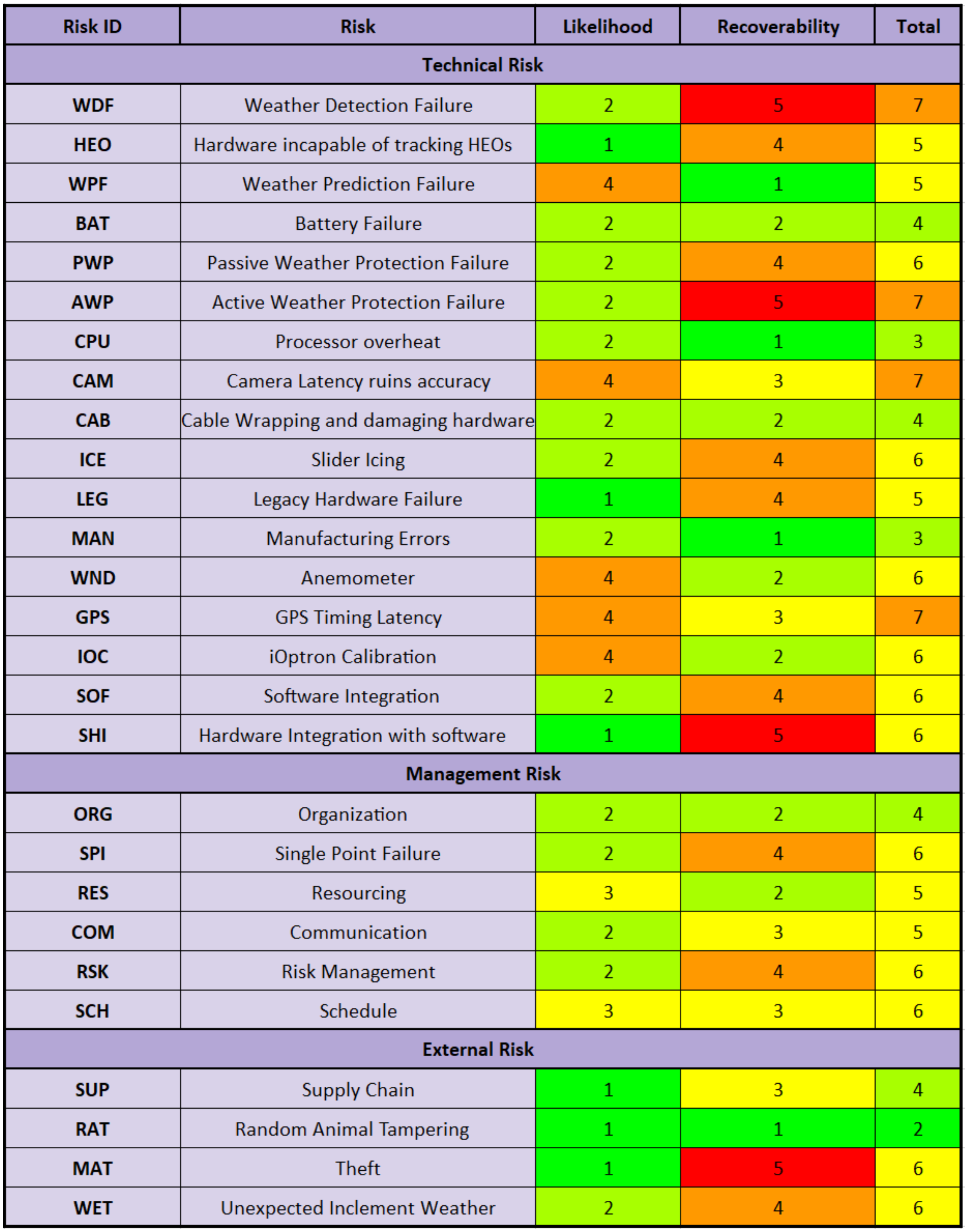
2.0 August 8th, 2014
Commands Corrected:
Corrected the wrong response of “:q#”, “:qR#” and “:qD#”.

Commands Removed:
“:AH#” has been removed.

Commands Changed:
“:GAS#” added the ability to judge if the mount is at zero position.
Guiding rate has been expanded from [0.10, 0.80] to [0.10, 0.90].
Changed response of “:GEC#”, “:GAC#”, “:Gg#”, “:Gt#” and “:AG#”.
Changed sending commands of time zone set, Local Date set, Local Time set, longitude set, latitude set, custom tracking rate set and target RA, Dec, Alt, Azi set.

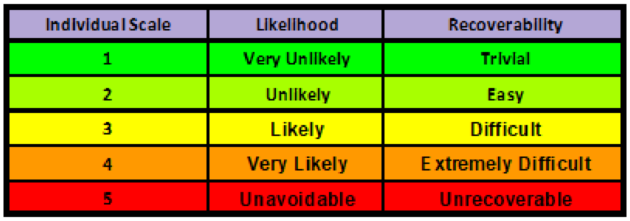
Commands Added:
Added set Hemisphere commands “:SHE0#”, “:SHE1#”.
Added auto search zero position command “:MSH#”.
Added set zero position command “:SZP#”.

E. Risks



Risk ID	Risk	Likelihood	Recoverability	Total
Technical Risk				
WDF	Weather Detection Failure	2	5	7
HEO	Hardware incapable of tracking HEOs	1	4	5
WPF	Weather Prediction Failure	4	1	5
BAT	Battery Failure	2	2	4
PWP	Passive Weather Protection Failure	2	4	6
AWP	Active Weather Protection Failure	2	5	7
CPU	Processor overheat	2	1	3
CAM	Camera Latency ruins accuracy	4	3	7
CAB	Cable Wrapping and damaging hardware	2	2	4
ICE	Slider Icing	2	4	6
LEG	Legacy Hardware Failure	1	4	5
MAN	Manufacturing Errors	2	1	3
WND	Anemometer	4	2	6
GPS	GPS Timing Latency	4	3	7
IOC	iOptron Calibration	4	2	6
SOF	Software Integration	2	4	6
SHI	Hardware Integration with software	1	5	6
Management Risk				
ORG	Organization	2	2	4
SPI	Single Point Failure	2	4	6
RES	Resourcing	3	2	5
COM	Communication	2	3	5
RSK	Risk Management	2	4	6
SCH	Schedule	3	3	6
External Risk				
SUP	Supply Chain	1	3	4
RAT	Random Animal Tampering	1	1	2
MAT	Theft	1	5	6
WET	Unexpected Inclement Weather	2	4	6

Table 29. Risks



Individual Scale	Likelihood	Recoverability
1	Very Unlikely	Trivial
2	Unlikely	Easy
3	Likely	Difficult
4	Very Likely	Extremely Difficult
5	Unavoidable	Unrecoverable

Table 30. Risk Key

F. Full Work Breakdown Structure

1. Administration

1.1. Management

1.1.1. Manage Schedule

1.1.2. Run Meetings

1.1.3. Track Progress

1.2. Financial

1.2.1. Order components

1.2.2. Keep Budget

1.3. Systems

1.3.1. Manage Scope Changes

1.3.2. Track Requirement Satisfaction

2. Software

2.1. Software Hardware Integration

2.1.1. Hardware Commanding

2.1.1.1. Command Translation

2.1.1.2. Commanding interface (camera, iOptron, motor)

2.1.1.3. Calibration

2.1.1.4. Closing protocol

2.1.1.5. Wire wrapping

2.1.2. Safety Override

2.1.2.1. Environmental Suite input

2.1.2.2. Shutdown Protocol

2.1.3. GPS

2.1.3.1. Timing Resolve

2.1.3.2. Data Pull into Scheduler

2.1.4. Roof Controller

2.1.4.1. Calibration

2.1.4.2. Contingency Plan

2.1.4.3. Manual open/close

2.2. Software Integration

2.2.1. Scheduler to Control interface

2.2.2. Camera Output

2.2.2.1. To storage

2.2.2.2. To Image Processing

2.2.3. Memory Path

2.2.3.1. From Camera

2.2.3.2. From Scheduler

2.2.3.3. From OD

2.2.3.4. From environmental

2.2.4. Image Processing to orbit determination

2.3. Scheduler Redesign

2.3.1. Rerun Function

2.3.1.1. Reweight

2.3.1.2. Edit filters

2.3.1.3. Search Algorithm

2.3.1.4. .watchdog

2.3.2. Shutdown and return home for environmental system

2.3.3. Integrate cable unwrap

2.4. Sensor Suite

2.4.1. Determine Sensor Functionality

2.4.1.1. Rain sensor

2.4.1.2. Anemometer

2.4.1.3. RH sensor

2.4.1.4. Pressure Sensor

2.4.1.5. IR thermometer

2.4.1.6. Lightning Sensor

2.4.1.7. Temp Sensor

2.4.2. Characterize uninstalled sensors

2.4.2.1. Rain sensor

2.4.2.2. Anemometer

2.4.2.3. RH sensor

2.4.2.4. Temp Sensor

2.4.2.5. IR thermometer

2.4.2.6. Lightning Sensor

2.4.2.7. Pressure Sensor

2.4.3. Characterize installed sensors as a system

2.4.4. Shutdown logic coding

2.4.4.1. Combinations (multi-sensor shutdown)

2.4.4.2. Individual Cases (single sensor shutdown)

2.4.5. Communications

2.4.5.1. Serial Port Communication

2.4.5.2. LTE Communication

3. Hardware

3.1. Main Box

3.1.1. Prototype

3.1.1.1. Finalize Design
3.1.1.2. Create CNC G-Code
3.1.1.3. Manufacturing
3.1.1.4. Inspection
3.1.1.5. Assembly
3.1.1.6. Function & Fit Check

3.1.2. Actual

3.1.2.1. Update Design
3.1.2.2. Create CNC G-Code
3.1.2.3. Manufacturing
3.1.2.4. Assembly
3.1.2.5. Foot Assembly

3.1.3. Lid Assembly

3.1.3.1. Slider
3.1.3.2. Rack and Pinion
3.1.3.3. Motor
3.1.3.4. Motor Controller
3.1.3.5. Switch
3.1.3.6. Lid

3.1.4. COTS Box Modifications

3.1.4.1. Vents
3.1.4.2. Vent Holes
3.1.4.3. Gaskets

3.1.5. Internal Component Installation

3.1.5.1. Processor Enclosure
3.1.5.2. Processor
3.1.5.3. PDB Enclosure
3.1.5.4. PDB
3.1.5.5. Batteries
3.1.5.6. Battery Harness
3.1.5.7. iOptron
3.1.5.8. iOptron Base
3.1.5.9. Camera

3.1.6. Wiring and Electrical Install

3.1.6.1. Batteries
3.1.6.2. iOptron
3.1.6.3. Camera

- 3.1.6.4. Processor
- 3.1.6.5. PDB
- 3.1.6.6. Motor
- 3.1.6.7. Motor Controller
- 3.1.6.8. Umbilical cord

3.2. Sensor Suite Box

3.3. Build Box

- 3.3.1. Final Design
- 3.3.2. Create CNC G-Code
- 3.3.3. Manufacture
- 3.3.4. Inspect
- 3.3.5. Assemble

3.4. Install Components

- 3.4.1. Rain sensor
- 3.4.2. Anemometer
- 3.4.3. RH/Pressure/Temp sensor
- 3.4.4. IR thermometer
- 3.4.5. Lightning Sensor

3.5. Install Wiring

- 3.5.1. Rain sensor
- 3.5.2. Anemometer
- 3.5.3. RH/Pressure/Temp. Sensor
- 3.5.4. IR Thermometer
- 3.5.5. Lightning Sensor

G. Gantt Chart Comparison

Below is a comparison of the Gantt chart at the critical design phase compared to the chart at project shutdown.

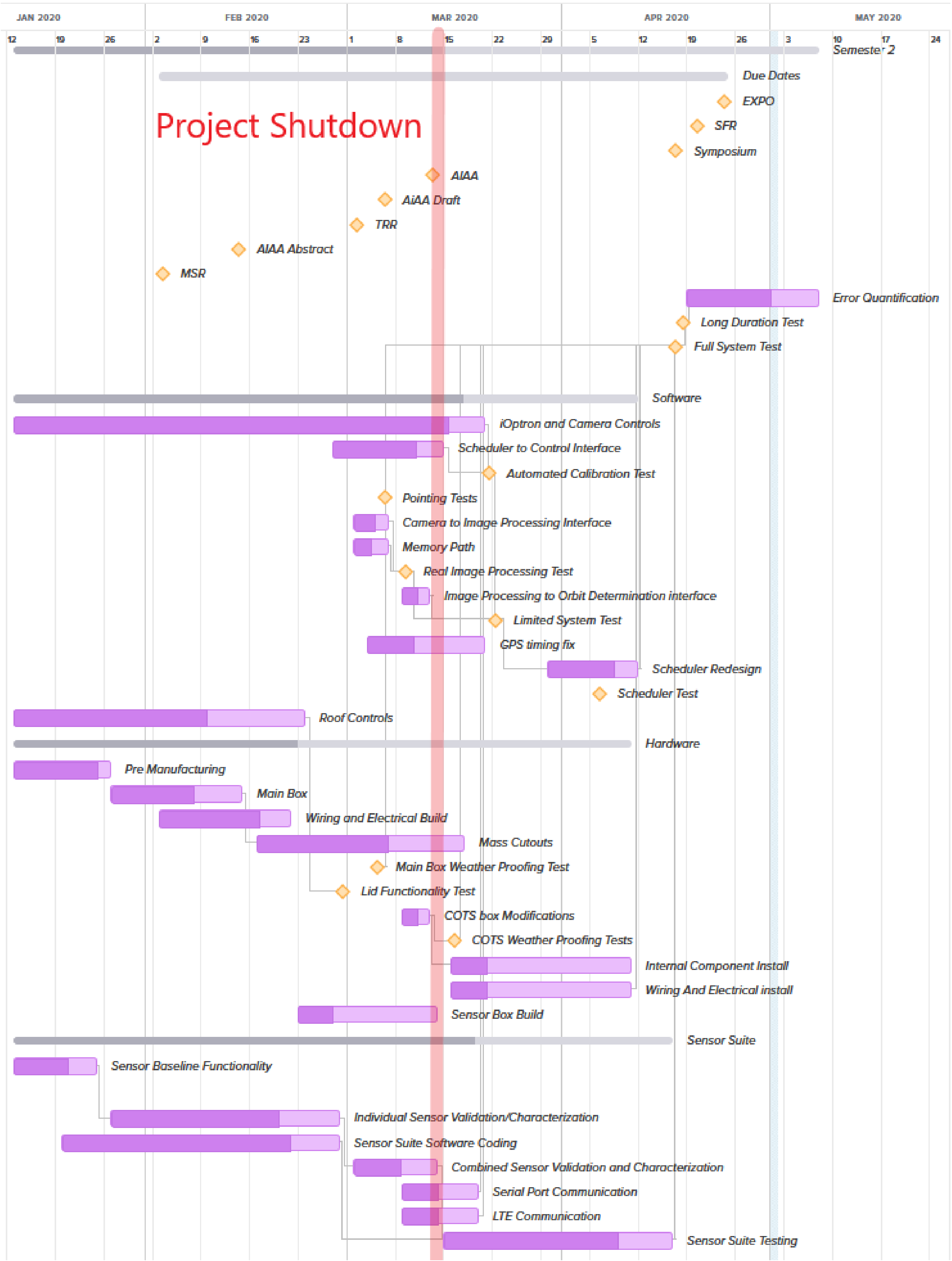


Figure 59. Gantt Chart at Project Shutdown

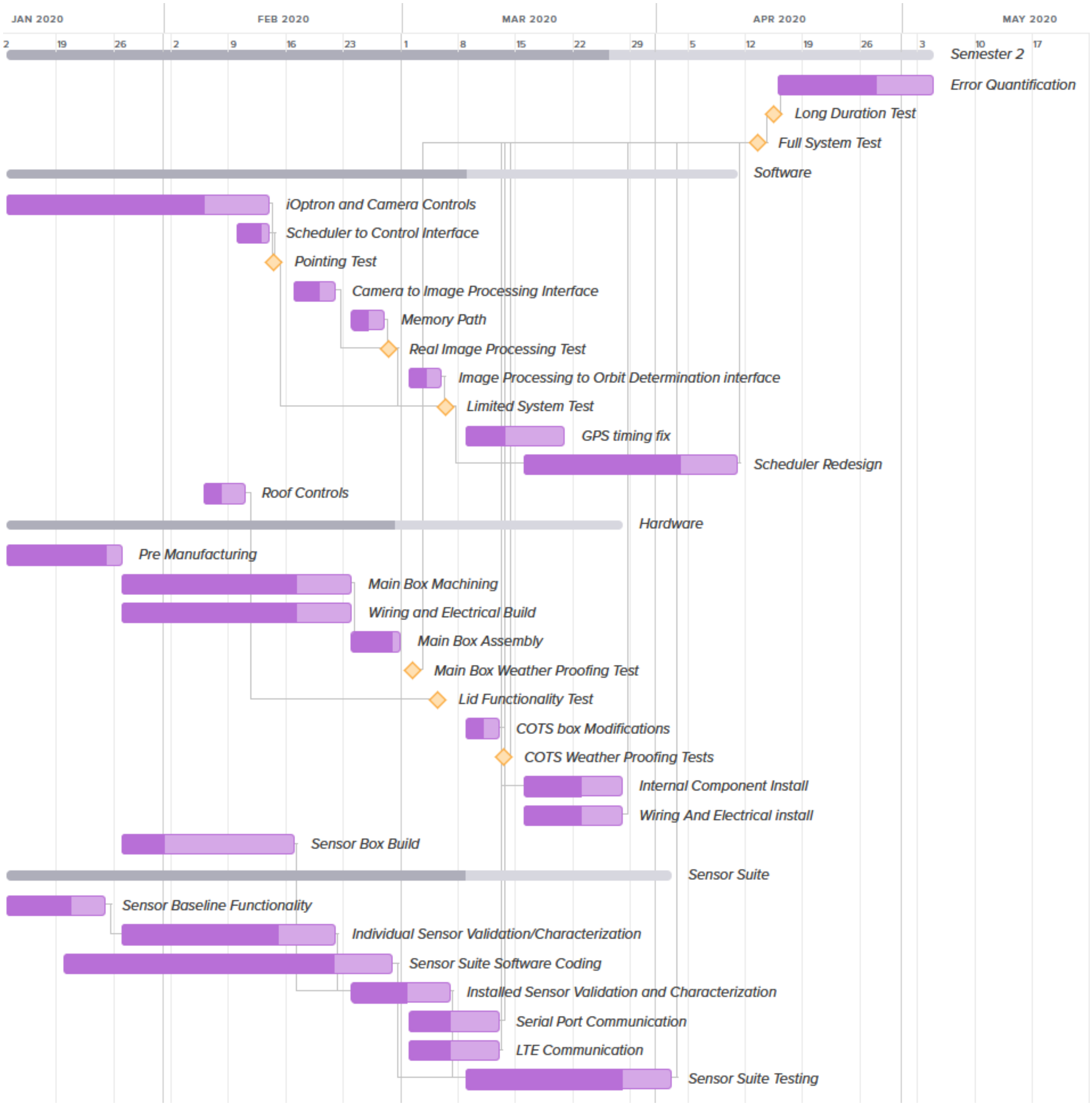


Figure 60. Gantt Chart at Critical Design Phase