Mission-Oriented Autonomy for Intelligent, Adaptive, and Multi-Agent Remote Sensing of Ice Sheets using Unmanned Aerial Systems

Aaron Blevins

Submitted to the graduate degree program in Aerospace Engineering and the Graduate Faculty of the University of Kansas in partial fulfillment of the requirements for the degree of Doctor of Philosophy.

Dr. Shawn Keshmiri, Chairperson

Committee members

Dr. Richard Hale

Dr. Carl Leuschen

Dr. Mark Ewing

Dr. Emily Arnold

Dr. Leigh Stearns

Date Defended: July 22, 2020

The dissertation committee for Aaron Blevins certifies that this is the approved version of the following dissertation:

Mission-Oriented Autonomy for Intelligent, Adaptive, and Multi-Agent Remote Sensing of Ice Sheets using Unmanned Aerial Systems

Dr. Shawn Keshmiri, Chairperson

Date Approved: July 22, 2020

Abstract

Throughout our history, humanity has been developing and progressing technology in order to help us better understand the world in which we live. As climate change becomes an increasingly urgent global crisis, scientists have been tasked with developing models for better understanding the complex dynamics involved, as well as to more accurately forecast the long term effects on our environment. With respect to sea level rise, both our knowledge of these dynamics and the accuracy of these models can be improved through the routine collection of crucial data concerning glacier ice thickness and bedrock topology. To accomplish this, innovative solutions are being developed by groups of inter-disciplinary research teams, combining fields such as earth-science, radar systems, data science, and aerospace engineering. Through this collaboration, we have the potential to leverage breakthroughs in unmanned systems technology and miniaturized, specialized sensors for comprehensive, precise, and routine data collection of key polar research objectives.

As Unmanned Aerial Systems (UASs) have become more reliable research platforms in recent years, they now have the capability to perform these remote sensing operations at a reduced cost compared to manned operations, while also providing repeatable, precision tracking capabilities along flight lines, enabling the surveying of tightly-spaced grids, and removing human flight crews from hazardous polar environments. However, the payload, range, and wind constraints for these platforms severely restrict their operational sensing footprint. Additionally, UASs generally have a much smaller wingspan compared to manned aircraft typically used in Earth Science missions, which becomes a challenging factor for incorporating efficient directive antennas at the low operating frequencies required for glacial sounding. The aim of this work is to address these issues and to enhance mission efficiency and the overall quality of data collection for these operations through the implementation of onboard mission-oriented autonomy that includes cognitive decision-making for intelligent survey operations, adaptive functionalities, and a scalable, robust framework for multi-agent operations.

As opposed to conventional methods for polar research operations which generally involve single-agent missions, using standard waypoint guidance and fixed-routes planned by human operators, the unique contributions of the developed mission-oriented autonomy in this work include:

1) Automated flight line generation for rapid and reliable mission planning of tightly-spaced flight lines required for cross-track synthetic aperture radar processes and surface clutter suppression, with required spacing based on the operating frequency of the onboard radar system.
2) Implementation of Dubins Path guidance methods into polar research operations for precision end-to-end survey of mission flight lines while taking into account the kinematic constraints of the fixed wing aircraft, as well as for efficiently traversing to and from a home loiter location during mission operations.
3) Cognitive, real-time optimal path planning through mission flight lines utilizing both deterministic and stochastic Traveling Salesman Problem heuristics.
4) Modifications to these Traveling Salesman Problem heuristics for ensuring safe, feasible, and reliable operations in real-time by taking into account aircraft range constraints.
5) Collaborative Multi-Agent survey operations utilizing space partitioning and Hungarian Assignment for distributed task allocation, as well as morphing potential fields for collision avoidance.
6) Modifications for Multi-Agent deployment scheduling to reduce inter-agent interference for sensitive radar systems to improve coherency of the collected data, and to rapidly and efficiently deploy agents into and out of survey areas.
7) Modifications for Heterogeneous flight operations for increasing operational capabilities through cross-platform collaboration.

8) Failsafe features to instill robustness in Multi-Agent operations with respect towards accommodating and adapting to single-agent system failures, by automatically re-planning collaborative survey operations.

In this work, the motivation for the creation of this mission-oriented autonomy is discussed, along with the methodology of each of the autonomy features, and the framework for implementation onto UAS platforms. Case studies are conducted for past and future polar research deployments using unmanned systems to assess the potential improvements in operational capabilities and data collection for the developed autonomy compared to conventional methods. Finally, the developed autonomy is implemented onto an embedded system for preliminary flight testing and validation, as well as used for intelligent mission planning for a manned operation.

Acknowledgements

First and foremost, I'd like to thank my incredible advisor Dr. Shawn Keshmiri for all of his support, dedication, and just overall relentless enthusiasm for research. Over these last five years working under him at the KU Flight Research Lab, I've had the opportunity to work on numerous research projects across a wide array of flight dynamics, unmanned systems, sensor integration, swarming, and polar research projects, and have gained truly invaluable experience from field testing that has prepared me for a successful career in unmanned systems. However, I'm most grateful for his example of how through hard work and dedication, research goals that may seem impossible at the time can ultimately be realized.

I'd also like to thank Dr. Hale and Dr. Leuschen for their dedication and commitment to the University of Kansas and the CReSIS program during their careers, and for their support for both me and my research endeavors. I'd like to thank Dr. Ewing for his leadership at the KU Flight Research Lab, as well as for his optimistic outlook throughout field testing, for all the laughs, and especially for his expansive wisdom, much of which I will carry with me as I navigate through life. I'd like to thank Dr. Arnold for teaching her aircraft antenna fundamentals course, and for our research collaboration over the last few years. Finally, I'd like to thank Dr. Stearns for her support as an expert in the field of glaciology, and for providing her perspective on mission objectives in glacial sounding operations.

I'd like to thank the University of Kansas for helping me develop as an academic and as a human, and for all the opportunities that I've been afforded over the last nine years. This university has allowed this small town Kansas kid to explore the world, to meet and work alongside a diverse group of talented individuals, and to gain a wide array of technical skills that have opened up many potential career opportunities. I'd like to thank the CReSIS program for introducing me to the field of unmanned remote sensing applications, something that I've developed a true passion for and hope to continue doing throughout in my career. And I especially want to thank the National Science Foundation and the Paul G. Allen Family Foundation for their support of the 2016 Russell Glacier deployment, which may truly be one of the most cherished two months of my life, which included being blessed enough to see the northern lights as a 23 year old college kid.

I'd like to thank my fellow graduate students at the KU Flight Research Lab, and for the endless hours that you put into our research projects. Dr. Bella Kim and Dr. Daksh Shukla, thank you for the countless early mornings and late evenings, for the many hours of troubleshooting, and for all of the computer science skills that you patiently taught me. I'd like to thank Dustin Hauptman for his work on developing our ground station and telemetry networks, Aaron McKinnis for his help with flight testing operations, and Hady Benyamen for the dynamic models that he created for our aircraft. I'd like to thank pilots Matt Tener, Grant Godfrey, Alex Zugazagoitia, and Darrel Cordle for their help bringing our research to life during flight testing.

I'd like to thank all of my friends and family for their support and encouragement throughout my academic career, and especially to my mother for always believing in me. I'd like to thank my high school teachers and coaches for preparing me both academically and mentally for my college career, and to my college pole vault coach Tom Hays for bringing me to the University of Kansas. Finally, I'd like to thank my amazing girlfriend, Dr. Melanie Snyder for all of her love and support during these final months of my graduate experience.

Contents

List of Figures

List of Tables

Chapter 1: Introduction

In order to understand the unique requirements involved in developing autonomy for polar research missions, a thorough background of earth-science objectives will first be discussed, followed by the technological developments enabling the integration of this work for survey operations. Finally, unique operational challenges and considerations will be addressed in order to define the need for the various features of the autonomy developed in this work.

1.1 Remote Sensing of Ice Sheets

The Earth's ice sheets are one of the key components in the modeling process for climate change, and two crucial factors for updating these models are sea ice volume measurements and outlet glacier basal conditions [1]. However, obtaining these measurements have proven to be very challenging due to the remoteness of polar regions, the technological challenges of developing high performance ice-penetrating radar systems, the integration of these systems onto aircraft, and the sheer scale of the polar surveying needed to create accurate climate models. Additionally, routine measurements of these areas are desirable for predictive modeling and for better understanding the complex dynamics of these systems.

As heightened levels of greenhouses gases in the atmosphere cause more thermal energy from solar radiation to be stored, the majority of this heat is transferred to the Earth's oceans. In doing so, the oceans effectively act as a buffer against global temperature increases, while leading to other environmental concerns [2]. In addition to decimating marine ecosystems and increasing the frequency and strength of tropical storms, warmer oceans further contribute to the issue of sea level rise due to the thermal expansion of the ocean water. For instance, an increase in global ocean temperatures by a single degree Celsius would result in a 2.3 meter increase in sea level [3]. Rising sea levels are projected to have a devastating impact on coastal populations in future decades [4]. Rising ocean temperatures also have another problematic effect: hindering the production of sea ice.

Sea ice is naturally formed through the seasonal freezing of polar ocean water, and is very crucial for maintaining the planet's global temperatures due to its higher relative albedo, or surface reflectivity, over surrounding water. Due to their respective albedo factors, an area of ocean water absorbs approximately 10 times more solar radiation than would the same area of sea ice. A positive feedback cycle is formed by this relationship, where rising ocean temperatures reduce the production of sea ice, leading to increased absorption of solar radiation, further increasing ocean temperatures [5].

The total volume of a portion of sea ice can be approximated by measuring the volume of the ice above the surface of the water using altimetry data obtained by satellites and using the density ratio of the ice to the surrounding sea water to approximate how much ice lies below. However, these approximations can be greatly improved by precisely measuring and accounting for the amount of snow cover that is accumulated on the surface of the sea ice.

Ice sheets, which are defined as continental glaciers over 50,000 km^2 in size, contain nearly 99% of the planet's supply of fresh water [6]. For context, the Greenland ice sheet contains enough water to raise global sea levels by 6 meters, while the Antarctic ice sheets could contribute up to 60 meters. The size of these glaciers directly depends upon their mass balance, which is the difference in accumulation and ablation rates [7]. As snowfall accumulation builds along the surface of the glacier, ice within the glacier naturally flows from its summit to its exterior, where it is deposited into the oceans via channels known as outlet glaciers. As outlet glaciers are the primary ablation mechanism for the ice sheet, they are of particular interest to the scientific community [8]. The flow rates of these outlet glaciers can be highly dependent on the bedrock topology and basal conditions, where subglacial water can accumulate and dramatically decrease the friction between the glacier and the bedrock, leading to increased glacial flow rates in a condition known as basal sliding [9].

NASA's Operation IceBridge was founded in 2009 in order to bridge ice altimetry data collections between the polar observation satellite deployments of ICESat (2003-2009) and ICESat-2 (2018-present). Operation IceBridge was a program that leveraged a fleet of highly specialized research aircraft carrying an assortment of innovative science instruments, including lidar, accumulation radars, infrared cameras, multichannel coherent radar depth sounders (MCoRDS), and many others in order to collect comprehensive ice sheet data [10]. Most of these deployments have utilized the P-3B and DC-8 aircraft as airborne laboratories that have been flown extensively over polar regions to routinely collect an assortment of observational data for ice sheets.

The Center for Remote Sensing of Ice Sheets (CReSIS) was founded in 2005 in order to develop new and specialized technologies for polar research and to improve and update climate model predictions [11]. The University of Kansas serves as the lead institute for CReSIS, with six collaborating partner institutes. The work conducted at CReSIS has contributed to vast innovations in the field of ice and snow radars, as well as the structural integration of these specialized sensors onto aircraft, and the operational collection of data during seasonal IceBridge deployments.

1.2 Unmanned Aerial Systems

As Unmanned Aerial Systems (UASs) become increasingly more capable, reliable, and cost effective, the advantages of leveraging these systems grow more apparent. Transitioning from manned to unmanned operations can enable precision tracking capabilities along tight grids, which is of particular interest for remote sensing operations collecting 3D spatial measurements using airborne sensors. Deploying unmanned systems into hazardous environments can remove human operators from unnecessary risk, while operational costs can be reduced due to decreased airframe, fuel, runway, and piloting costs.

Additionally, the automation of unmanned operations can drastically increase efficiency, as well as feasibly enable scaled operations for utilizing collaborating swarms of aerial platforms.

The increased popularity of UASs can be attributed to a variety of factors enabling their success. Advancements in propulsion system performance and efficiency have increased available aircraft thrust and decreased energy consumption, leading to increases in available payload and extending aircraft range and endurance. Recent breakthroughs in battery technologies have increased the capacitance and energy density for onboard batteries, directly increasing flight times and power outputs for the aircraft. The capabilities for UASs has dramatically been expanded due to the miniaturization of airborne sensors, allowing for their integration onto increasingly smaller platforms. Finally, the exponential advancements in microprocessor technology has produced lightweight microprocessors of increasingly small form factors, which are capable of running complex, real-time processors aboard UASs, enabling the implementation of various levels of autonomous features into these systems.

UASs have recently been utilized across a wide variety of civil, commercial, and military applications [12-20]. Infrastructure inspection has greatly benefitted by the introduction of UASs, particularly for obtaining precise and comprehensive images of tall and dangerous structures such as bridges, power lines, and wind turbines. UASs have aided the agricultural sector through precision crop monitoring and crop dusting, while rapid response teams have used UASs to quickly deploy life-saving medicines and healthcare devices to individuals experiencing medical emergencies. Stunning aerial views obtained by UASs have been used extensively in the real estate sector, in the film making industry, and for media coverage of events. Swarms of UASs have recently been used for dazzling aerial light shows, capable of rapidly forming and morphing 3D structures. Scientists from a myriad of diverse fields have put UASs to use for important data collection, including studying and tracking wildlife, collecting information from hazardous areas such as volcanoes, and even for archaeological surveying. The quick response capabilities of UASs have been leveraged in response to humanitarian crises, such as providing real-time monitoring of areas affected by natural disasters such as hurricanes or earthquakes, as well as monitoring and combating the progression of wildfires. UASs have successfully aided in search and rescue efforts to save the lives of individuals lost in mountains, adrift at sea, and trapped in warzones.

In 2007, CReSIS began the construction process for a purpose-driven UAS for polar research [21]. Named the “Meridian”, the aircraft was designed, manufactured, and flight tested by students and faculty at the University of Kansas. With a wingspan of over 26 feet and a takeoff weight of 1,100 lbs, the aircraft was the largest civilian-operated UAS in the world. The aircraft was capable of carrying 120 lbs of payload (suitable for the radar system of the time) over a range of 1,750 km, along with a 300 km reserve. Removable, modular wings were designed for shipping feasibility, rapid aircraft assembly in the field, and to incorporate various array configurations needed for ice-penetrating radars. Due its size, extensive flight testing was needed to properly characterize its stability and control characteristics, while precautions were taken to ensure the safety of the aircraft and the operators [22]. The aircraft and the flight research team from the University of Kansas voyaged on several polar deployments to test the aircraft in its intended conditions, including two trips to McMurdo Station, Antarctica, and one trip to Neem, Greenland, where the aircraft was successfully flown autonomously overhead and preliminary radar testing was conducted. Prior to Meridian flights, a 33% scale YAK-54 model aircraft was used to train the RC pilots for takeoff and landing on the runway conditions, as well as for difficulties that arise in maintaining visual reference of the aircraft in the harsh polar environment.

However, following the miniaturization of the radar system into a 5 lb unit, it was then possible to incorporate the system into the model aircraft previously used for pilot training. Custom glass-composite wings were designed for a 40% scale YAK-54 aircraft, named the “G1X”, and dual frequency HF/VHF antennas was designed to be incorporated into the wing platform and airframe [23]. With a 17-foot wingspan and an 85-lb takeoff weight, the G1X was capable of carrying the radar system for a total range of 100 km. While this range is significantly lower than the Meridian, the operational costs and risks involved were greatly reduced. Additionally, the G1X requires significantly less runway area to operate from and can fly at lower speeds, enabling increased maneuverability for closely spaced grids. Using invaluable experience gained from the Meridian program, the G1X was deployed to Antarctica in 2014, where it performed the first successful sounding of glacial ice using a UAS-based radar [24].

Lowering the platform, transport, and operational costs directly led to the feasibility of increasing the overall mission reliability by having redundant platforms available at the field. Following design improvements, a newer version for the G1X platform was developed with a wingspan of 14.5 feet and a takeoff weight of 71 lbs, increasing the aircraft range to 150 km. In the spring of 2016, the KU team deployed to Kangerlussuaq, Greenland, with three operational aircraft, two of which were fully outfitted with radar and antenna systems, and the third was strictly used for preliminary flight testing and as a reserve airframe. During this deployment, KU team successfully collected ice thickness and bedrock topology measurements of the Russell Glacier using a 35 MHz radar, through eight over-the-horizon flights covering 28 distinct survey lines for a total of over 200 miles of glacial surface [25]. Results were validated through comparison with previous measurements of the same area from JPL’s WISE VHF sounder [26-27]. Previously missing bedrock information was revealed through the G1X’s HF sounder due to its lower sensitivity to both surface roughness and temperate ice attenuation. The five CReSIS UAS deployments are shown in Figure 1, while the improved imaging of the Russell Glacier compared to JPL’s WISE Sounder [25] is shown in Figure 2.

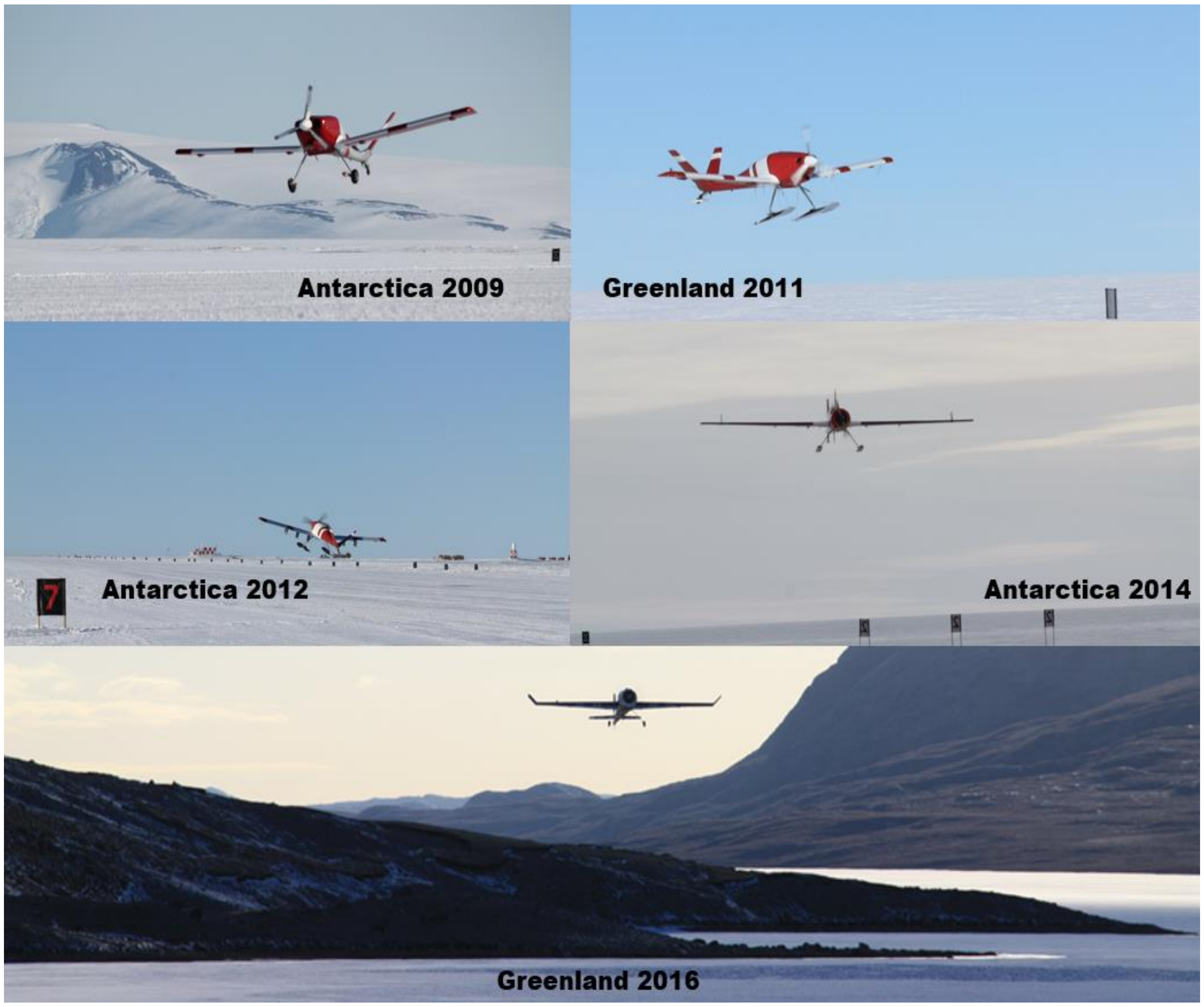


Figure 1: CReSIS UAS Deployments (Photo Credit: KU Flight Research Lab)

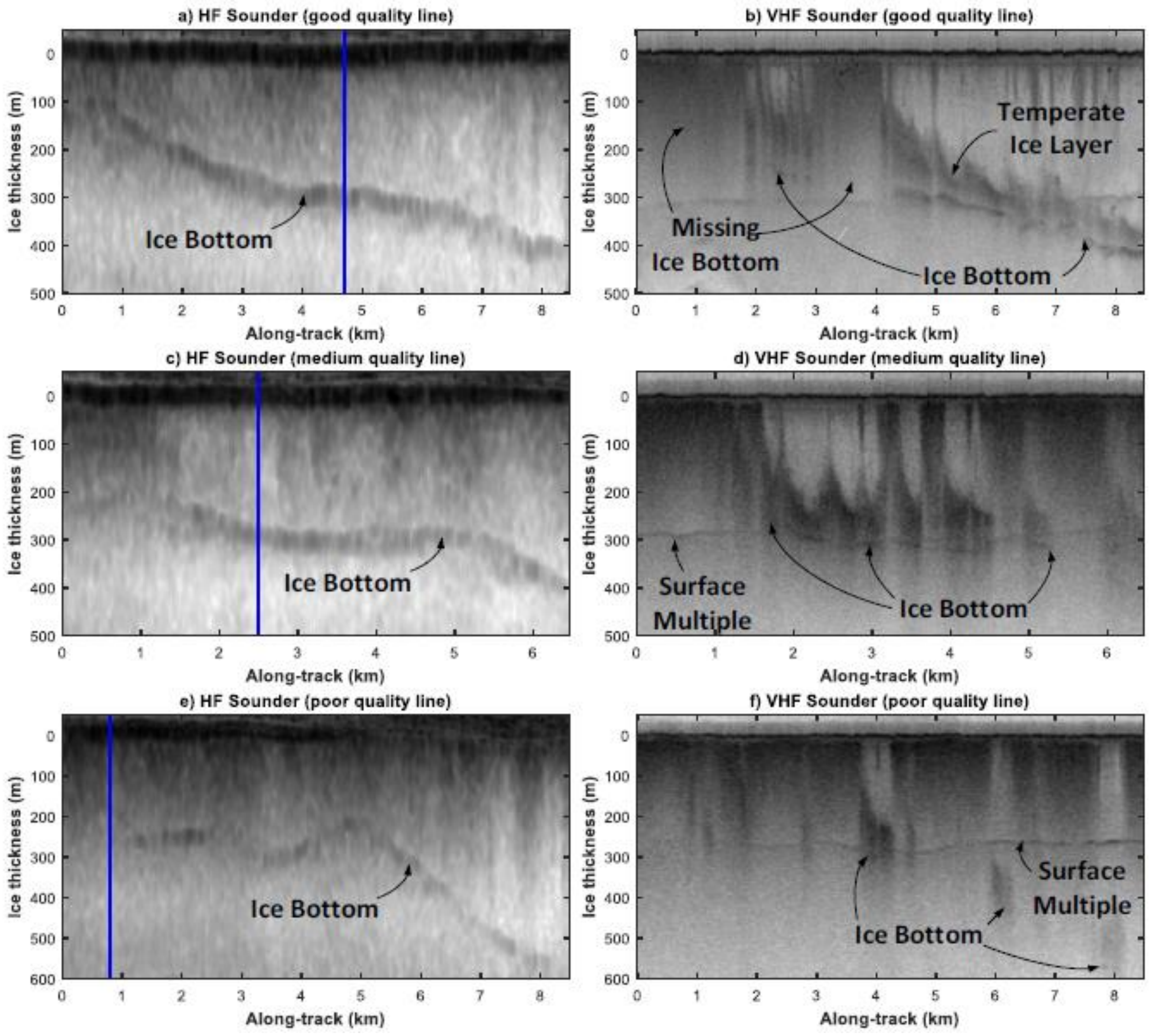


Figure 2: Comparison of imaging along the Russell Glacier from JPL WIFE HF Sounder (Left) and G1X VHF Sounder (Right) [25]

1.3 Autonomy Trends

As increasingly capable microprocessors are implemented onto UASs, the ability now exists to incorporate complex onboard processes to enhance mission performance and efficiency. Technological advances now allow the shift of unmanned system operations from predominantly deterministic to that of adaptive and multi-agent missions, incorporating cognitive, real-time decision making. Conventional methods for polar research operations include standard waypoint following guidance, using a fixed route pre-planned by human operators. However, countless research works have recently been conducted involving the integration of complex onboard processes onto UASs, including the use of nonlinear model predictive controllers, artificial intelligence, visual recognition, and search and track operations [28-30].

Due to the size of the survey areas and the lack of communication infrastructure in polar regions, the UASs are required to predominantly operate in the absence of human direction. Therefore, it is incredibly advantageous in terms of mission performance, reliability, and safety for the UAS to have onboard capabilities that enable cognitive, real-time decisions when exploring unstructured environments. Similar research has been performed for unmanned system operations in communication-denied environments has been the focus of much research for areas such as underwater surveying and Mars rovers [31-33].

Increasing the automation of the mission planning process can be used to shift significant workload from the vehicle operators to the onboard autonomy. This shift in workload enables rapid deployment of vehicles into survey areas, reduces the possibility of human error while manually inputting ordered mission waypoints, and frees up operator attention for other important tasks in the harsh polar environment. Additionally, it decreases the number of skilled personnel needed to operate the vehicles, and increases the feasibility of scaling the surveying operations by deploying multiple collaborative UASs within narrow time windows. Thanks to newly available onboard computational power, UASs in these multi-agent operations are now capable of real-time decentralized path planning based on the emergent behavior of the collaborating systems, forming and maintaining complex virtual structures to perform formation flights, as well as ensuring collision avoidance between airborne platforms with high speed and high inertia. Due to the immense advantageous of enhancing overall footprint, increasing mission reliability, and distributing payload among systems, multi-agent UASs have become the focus of vast recent and ongoing research works. In this work, autonomy features are designed and implemented in order to increase the operational capabilities of the UASs, and compensate for their limited footprint due to payload, range, and wind restrictions.

1.4 Mission Planning and Operational Considerations

This section breaks down the mission requirements and operational challenges involved in the remote sensing of ice sheets.

1.4.1 Mission Breakdown and Environmental Concerns

In order to obtain vital ice sheet measurements, UASs are used as a means of maneuvering the radiating antenna elements along the desired flight lines, where the resulting backscatter is collected to update existing models. These flight lines represent the cross sectional “slices” of interest in the ice sheet, and areas between flight lines can be subjected to interpolation techniques. While this data is highly desired by scientists, these operations come with numerous challenges.

The size and shape of the radiating elements, along with weight of the radar system, typically constrain the choice of vehicle type to that of a fixed-wing aircraft, which must continuously maintain above a minimum airspeed while airborne to prevent aerodynamic stalling and are constrained to a minimum turning radius. Additionally, this limits the location of the ground operations to the proximity of a suitable runway for the aircraft to take off and land on, which might consist of snow covered, icy, or rugged surfaces. This area must also be free of any surrounding features obstructing the climb out and final approach phases for the aircraft.

Mission operations typically consist of a takeoff procedure (1), the entrance of the system into an overhead home loiter circle (2), the deployment of the system over the horizon (3), the surveying of the mission flight lines (4), the return of the system into the overhead home loiter circle (5), the landing of the system on the runway (6), and finally, the extraction of the collected data (7). The overhead home loiter serves as a way for the aircraft to maintain airspeed while undergoing final checks of system diagnostics (for both the autopilot and radar systems) before the system is deployed to the flight lines, as well as a rendezvous point for the aircraft to return to before beginning landing procedures.

The remoteness of the surrounding area constraining the location of the ground operators, coupled with the length of these flight lines, necessitates over-the-horizon (OTH) operations in which telemetric links may be intermittent or lost altogether based on range and line-of-sight restrictions, such as ridges. The success of these missions then depends on the ability of the UAS to operate correctly in the absence of continuous human direction.

The windows of opportunity for operations in these regions are limited due to a variety of factors. While UAS operations are generally limited to operating below wind and gust thresholds defined by the airframe and propulsion system, the buildup of dense, cold air across the ice sheet, coupled with the differential elevation of its summit and terminus, lead to strong katabatic winds that can prevent operations for several days at a time. Other weather conditions that may prevent operations include snowfall and fog, impairing visual tracking of the aircraft from the ground station and potentially leading to icing conditions on the wings or pitot tube. Local area temperatures may drop below the operating threshold for aircraft avionics, control systems, onboard batteries, and the operating crew. Finally, seasonal daytime variations in these polar regions may also limit operational windows.

Due to these challenges, the efficiency of operations when suitable conditions are available is vital to the success of polar research deployments. Therefore, the demands for the onboard UAS autonomy include determining an efficient route to traverse the flight lines, following efficient trajectories when transitioning between flight lines, maintaining tight tracking of the flight lines, and making intelligent mission decisions in the absence of operator supervision, including ensuring a safe return to the operating area. These capabilities are crucial to enhancing sensing footprint, minimizing operational time, and improving mission reliability.

1.4.2 Synthetic Aperture Radar (SAR) Processes

The radar backscatter data collected from the aircraft while tracking a flight line is post-processed by synthetic aperture techniques in order to obtain fine-resolution imaging in the along-track direction. Synthetic aperture radar (SAR) has utilized for a wide variety of airborne radar platforms in order to use the motion of the radar system over the target area as a means of synthesizing a very large array along the direction of the motion. Vast improvements in SAR processing techniques for glacial ice has been made by researchers at CReSIS to produce coherent images from data collected during Operation IceBridge [34-37].

However one important consideration for SAR processing is that the angular motion of the radiating platform can cause distortion in the resulting images. While some compensation techniques can be applied to mitigate the effect of angular perturbations, steady-level flight is desired to be maintained by the autopilot of the aircraft while tracking the flight lines in order to obtain more coherent images. Additionally, as the maximum directivity of the aircraft antenna is typically designed to align with the Z-body axis for the aircraft, very low aircraft pitch and roll angles are desired while tracking flight lines. For aircraft carrying omnidirectional antennas (such as the bowtie-dipole on the G1X system), the off-nadir angle for the azimuth direction of the antenna can be disregarded (in this case the pitch axis of the aircraft).

Due to rugged terrain surfaces, ice-penetrating radars frequently experience surface clutter, which is the phenomenon of off-nadir signals reflecting off of surrounding terrain surfaces and returning to the receive antenna at the same instance as the desired basal reflection. The resulting signal-to-noise ratio of the surface clutter to the basal return is significantly large due to the difference between the free-space and the interglacial attenuation that is applied to the respective signals [38]. In order to mitigate the effect of surface clutter, researchers typically fly a series of closely-spaced passes along the desired flight lines and apply clutter-suppression techniques to the overall data in order to produce coherent images along the flight line.

Additionally, by separating the individual passes by a quarter of the operating wavelength, synthetic aperture techniques can be applied in the cross-track direction in order to improve resolution in that axis, which is typically limited due to the aperture length restrictions on the UAS platform. Various beamforming techniques can be applied to the data to take into account the tracking errors along the flight lines [39].

1.4.3 Multi-Agent Operations

The limited footprint of UASs can be dramatically compensated for through the use of collaborating, multi-agent operations. However, multi-agent operations can increase the number of necessary personnel needed to prepare and monitor the vehicles, as well as introduce the possibility of inter-agent collision, thus requiring the implementation of a collision avoidance mechanism. Additional considerations must be made when incorporating numerous, sensitive radar systems into a confined airspace. For snow accumulation radars, which typically radiate from 2-8 GHz, a highly directive antenna can be integrated onto the UAS platforms and can therefore operate in the same general area without experiencing cross-platform interference. However, glacial sounding applications require much lower frequencies (typically below 55 MHz), which necessitate the use of a very long, low-directivity antenna. These antennas have a very wide radiation pattern, and could result in cross-platform interference with a sufficiently large signal-to-noise ratio that could mask the desired basal returns. In these instances, only one UAS platform should be radiating at a single instance in order to ensure coherent data collection during distributed operations.

Chapter 2: Autonomy Methodology

In this chapter, the methodology behind the eight unique features of this mission-oriented autonomy will be discussed.

2.1 Automated Flight Line Generation

The first component of the mission-oriented autonomy developed in this work involves the automated generation of closely-spaced lines during sensor-driven mission planning for polar flight operations. The necessity of these closely-spaced lines for cross-track synthetic aperture processes and for clutter suppression are described in Section 1.4, where the spacing between these lines should be equal to one-quarter of the wavelength of the operating radar system. The automated generation of these lines reduces the workload on the human operators (who would otherwise have to manually input the waypoints for each line in the cluster), and reduces the chance of error corresponding to that manual process. Reducing this workload on the operator also enables them to direct their focus onto other portions of the flight operations and overall can increase operational efficiency and safety.

The flight line centroid about which the automated cluster will be generated can be described as a line segment formed between points $\mathrm{A} = \{\mathrm{A}_N, \mathrm{A}_E\}$ and $\mathrm{B} = \{\mathrm{B}_N, \mathrm{B}_E\}$ in a North-East local frame, using feet as the units relative to the origin location. These points can be converted from geodetic coordinates into a local frame using a transformation such as shown below in Equation 2.1, where an origin location $\mathrm{O} = \{\mathrm{O}_{Lat}, \mathrm{O}_{Lon}\}$ is defined and the transformation factors F_{Lat} and F_{Lon} refer to the conversions for latitude and longitude degrees into feet in the local frame. Note that the factors F_{Lat} and F_{Lon} vary based on the geodetic location, but in this work these factors are used based on the origin location O and assumed constant for the surrounding mission area. In order to determine the direction in which to generate lines, the heading angle for the flight line is determined using Equation 2.2, while the unit vector perpendicular to this heading is determined using Equation 2.3. To determine the required spacing, the wavelength of the radar system can be calculated based on its operating frequency in Hz, as shown in Equation 2.4, where "c" refers to the speed of light in m/s.

$$[\mathrm{A},\mathrm{B}]_{Local} = [(A_{Lat} - O_{Lat}) * F_{Lat}, (A_{Lon} - O_{Lon}) * F_{Lon}] \qquad \textit{Eq. 2.1}$$

$$\psi_{AB} = atan2\left(\frac{B_E - A_E}{B_N - A_N}\right) \qquad \textit{Eq. 2.2}$$

$$\widehat{\mathrm{V}}_{\mathrm{AB_P}} = \left[\cos\left(\psi_{\mathrm{AB}} + \frac{\pi}{2}\right), \sin\left(\psi_{\mathrm{AB}} + \frac{\pi}{2}\right)\right] \qquad \textit{Eq. 2.3}$$

$$\lambda = \frac{f}{c} \qquad \textit{Eq. 2.4}$$

The pseudocode for generating the number of flight lines (or rather, the number of passes) N_P for each centroid (A_0, B_0) is shown below. Example clusters are generated and shown in Figure 3 and Figure 4 for 7 and 8 lines, respectively. Note that the spacing factor η_λ is typically set to 0.25 but can vary depending on operational constraints and interpolation methods.

```
$\boldsymbol{for\ i = 1: N_P}$
    $\boldsymbol{A_i = A_0 + \eta_\lambda \lambda \left(i - \frac{N_P+1}{2}\right) \widehat{V}_{AB_P}}$
    $\boldsymbol{B_i = B_0 + \eta_\lambda \lambda \left(i - \frac{N_P+1}{2}\right) \widehat{V}_{AB_P}}$
***end for***
```

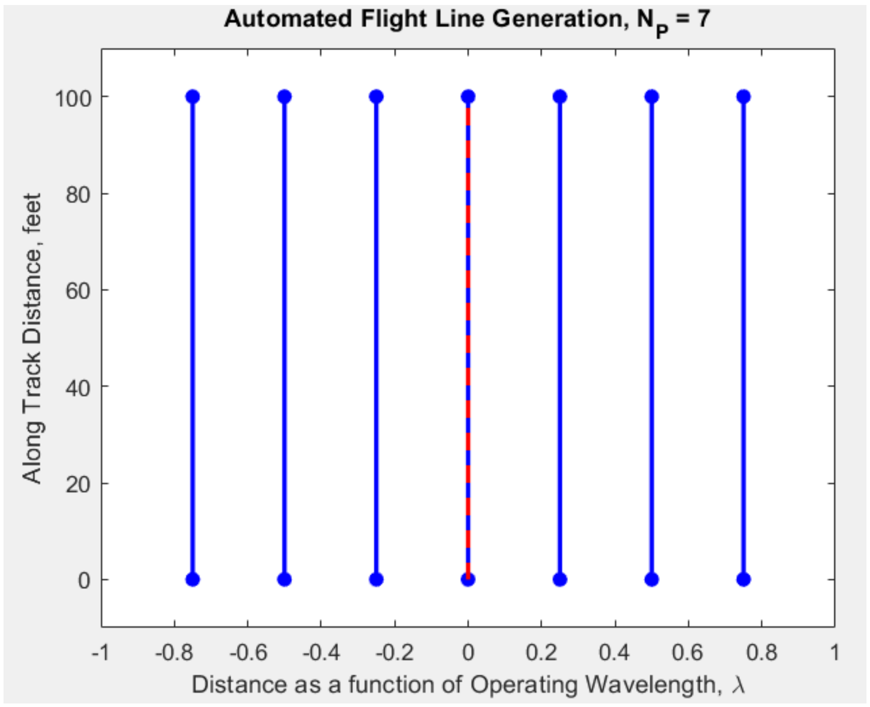


Figure 3: Automated Flight Line Generation Example, N_P = 7

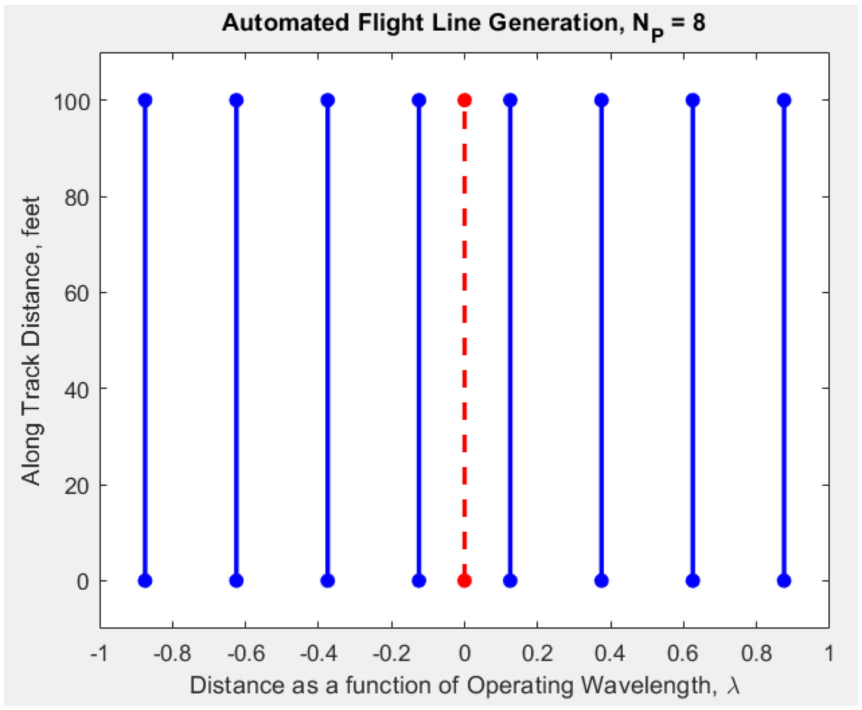


Figure 4: Automated Flight Line Generation Example, NP = 8

The cluster generation can then be performed on any number of centroids for their respective number of passes N_P. Figure 5 shows an example of three centroid flight lines, where Figure 6 shows the resulting clusters for a 55 MHz operating frequency and desired cluster N_P values of 4,5, and 6 from (left to right).

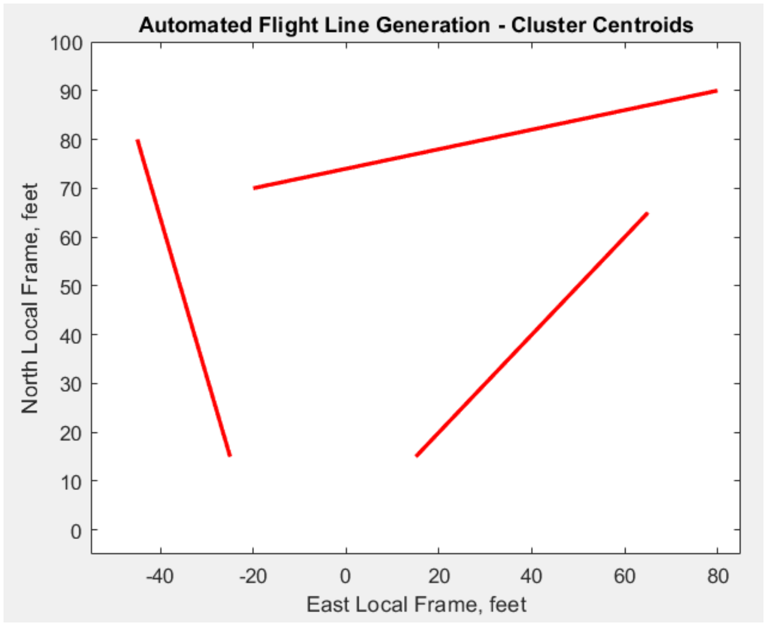


Figure 5: Automated Flight Line Generation - Cluster Centroids

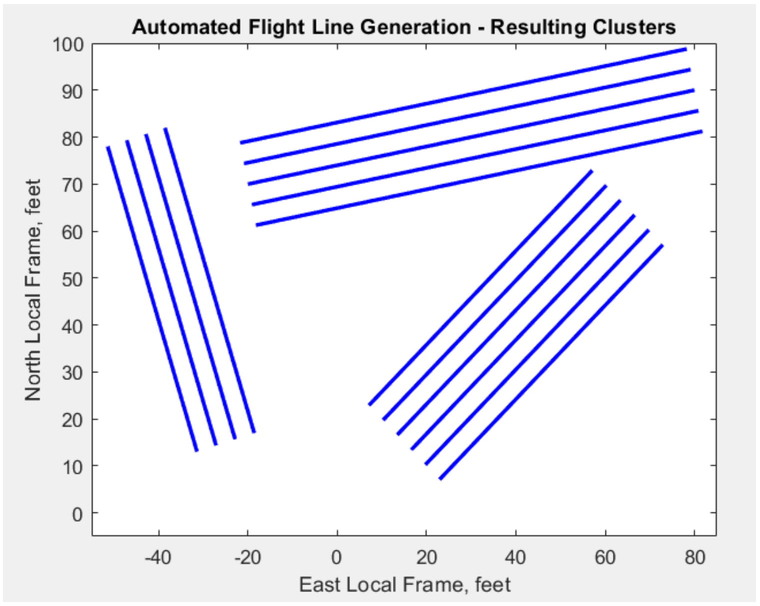


Figure 6: Automated Flight Line Generation - Resulting Clusters

2.2 Dubins Path Guidance

In order to quickly and efficiency transition onto survey flight lines, the aircraft ideally should follow an optimal path along which the aircraft will arrive at the start of the flight line in the shortest amount of time, with the respective flight line heading. Approaching along this trajectory would allow for full end-to-end coverage of the flight line with minimal off-track errors and aircraft bank angles. Additionally, the curvature of candidate paths are kinematically constrained by the minimum turning radius of the fixed-wing aircraft. The generation of this optimal path can be achieved by utilizing Dubins Paths architectures.

2.2.1 Dubins Paths

First introduced by Lester Dubins in 1957, Dubins Paths are the curves of minimum length between two points with prescribed initial and terminal headings in a two-dimensional plane, where the curve is subjected to a maximum curvature constraint and the path must be traversed forwards. In [40], Dubins proved that these optimal paths can be constructed by utilizing simple architectures involving combinations of straight line segments and arcs of maximum curvature. Later extensions of these paths included additional constraints or variations, namely backwards motion being incorporated into what are known as Reeds-Shepp curves [41]. Since their introduction, Dubins Paths have been used for their optimality in a wide variety of unmanned applications, including aerial vehicles [42-44], ground rovers [45-47], water surface vehicles [48], and underwater vehicles [49].

In this work, the inputs into the Dubins Path construction algorithms include the aircraft position in the north-east local frame, $\mathrm{P}_1 = \{\mathrm{P}_{1N}, \mathrm{P}_{1E}\}$, the aircraft heading angle (ψ_1), the position of the starting point of the flight line, $\mathrm{P}_2 = \{\mathrm{P}_{2N}, \mathrm{P}_{2E}\}$, the heading angle of the flight line (ψ_2), and the minimum turning radius of the aircraft (R_{Min}). This minimum turning radius can be calculated using the inertial ground speed V_g and the maximum bank angle ϕ_{Max}, as shown in Equation 2.5. These inputs are shown in Figure 7 in an example scenario for an approach onto a flight line. As will be demonstrated in Section 2.2.2., the optimal path for this scenario can be determined by constructing a series of candidate paths from predefined architectures and choosing the shortest existing solution.

$$R_{Min} = \frac{V_G}{g \tan \phi_{Max}} \qquad \textit{Eq. 2.5}$$

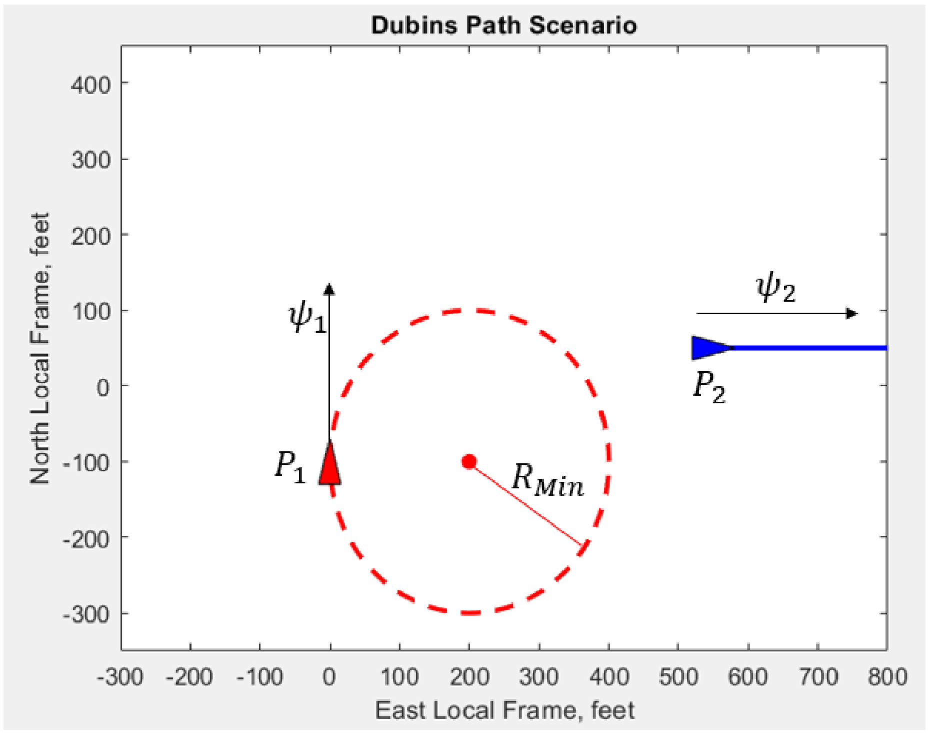


Figure 7: Dubins Path Example Scenario

2.2.2 Dubins Path Architectures

The distance-optimal path along which the aircraft can approach flight lines will consist of one of six predefined architectures outlined in this section. These architectures can be constructed using unique combinations of line segments and arcs of the minimum turning radius in either the clockwise (CW) or counter-clockwise directions (CCW). More commonplace terminology can be substituted that denotes line segments as straight (S), clockwise maneuvers as right-hand turns (R), and counter-clockwise maneuvers as left-hand turns (L) [50].

The first Dubins Path architecture consists of a clockwise arc, which exits on a tangential line segment, and enters onto another clockwise arc. This path is also referred to as an "RSR" architecture. The initial and terminal arcs lie on circles C_1 and C_2, respectively, which are of radii R_{Min}. The center locations for C_1 and C_2 can be determined by projecting points P_1 and P_2 a distance R_{Min} in the direction 90° clockwise from their respective headings, shown below in Equations 2.6 and 2.7. The heading of the tangent line between the arcs can be determined as parallel to the heading between the circle centers, due to their common radii, as shown in Equation 2.8. The length of this path can then easily be calculated by summing the length of the arc segments and the length of the tangent line, shown in Equation 2.9, where the $\mathrm{W}_0^{2\pi}$ function denotes a wrapping function of the angle between 0 and 2π. Note that this path architecture exists for all scenarios, as the atan2 function in Equation 2.8 is valid across the domain for circles C_1 and C_2. An example scenario for a Dubins Path of architecture "Type 1" is shown in Figure 8**.**

$$C_1 = P_1 + R\left\langle cos\left(\psi_1 + \frac{\pi}{2}\right),\ sin\left(\psi_1 + \frac{\pi}{2}\right)\right\rangle \qquad \textit{Eq. 2.6}$$

$$C_2 = P_2 + R\left\langle cos\left(\psi_2 + \frac{\pi}{2}\right),\ sin\left(\psi_2 + \frac{\pi}{2}\right)\right\rangle \qquad \textit{Eq. 2.7}$$

$$\psi_{12} = atan2\left(\frac{C_{2E} - C_{1E}}{C_{2N} - C_{1N}}\right) \qquad \textit{Eq. 2.8}$$

$$L_1 = \|C_2 - C_1\| + R\mathrm{W}_0^{2\pi}\left\langle \mathrm{W}_0^{2\pi}\left\{\psi_{12} - \frac{\pi}{2}\right\} - \mathrm{W}_0^{2\pi}\left\{\psi_1 - \frac{\pi}{2}\right\}\right\rangle$$

$$+ R\mathrm{W}_0^{2\pi}\left\langle \mathrm{W}_0^{2\pi}\left\{\psi_2 - \frac{\pi}{2}\right\} - \mathrm{W}_0^{2\pi}\left\{\psi_{12} - \frac{\pi}{2}\right\}\right\rangle \qquad \textit{Eq. 2.9}$$

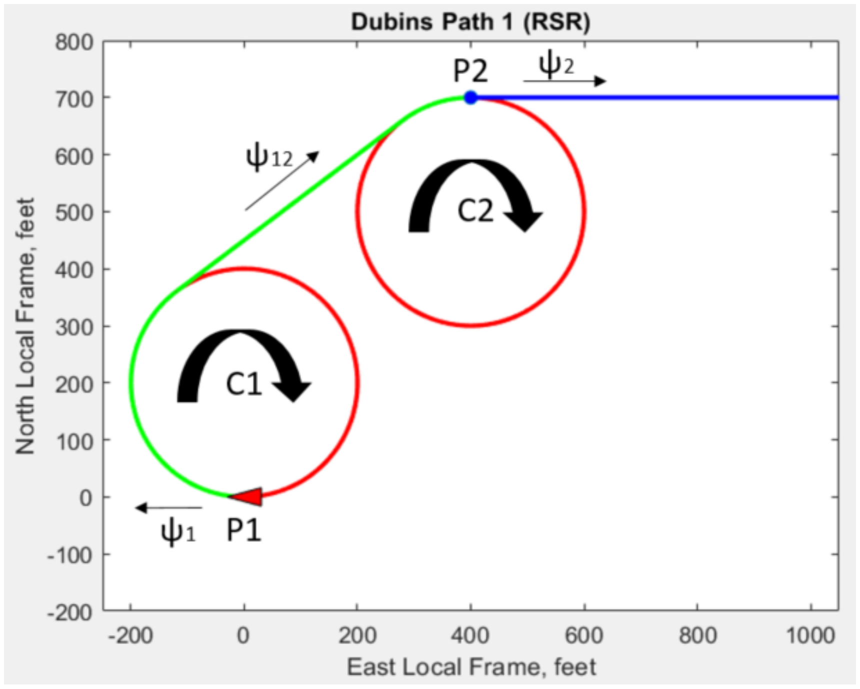


Figure 8: Dubins Path Architecture Type 1 (RSR) Example

Similarly, a "Type 2" architecture consists of a counter-clockwise arc, which exits on a tangential line segment, and enters onto another counter-clockwise arc. This path is also referred to as an "LSL" architecture. Similar inputs and notation as in Type 1 are used in construction, except for the circles C_1 and C_2 being positioned in the direction 90° counter-clockwise from the initial and terminal headings (Equations 2.10 and 2.11). Similar logic to the previous architecture is used to compute the tangent heading (Equation 2.12) and the total length of the path (Equation 2.13). Note that this path architecture exists for all scenarios, as the atan2 function in Equation 2.12 is valid across the domain for circles C_1 and C_2. An example scenario for a Dubins Path of architecture Type 2 is shown in Figure 9***.***

$$C_1 = P_1 + R\left\langle cos\left(\psi_1 - \frac{\pi}{2}\right),\ sin\left(\psi_1 - \frac{\pi}{2}\right)\right\rangle \qquad \textit{Eq. 2.10}$$

$$C_2 = P_2 + R\left\langle cos\left(\psi_2 - \frac{\pi}{2}\right),\ sin\left(\psi_2 - \frac{\pi}{2}\right)\right\rangle \qquad \textit{Eq. 2.11}$$

$$\psi_{12} = atan2\left(\frac{C_{2E} - C_{1E}}{C_{2N} - C_{1N}}\right) \qquad \textit{Eq. 2.12}$$

$$L_2 = \|C_2 - C_1\| + R\text{W}_0^{2\pi}\left\langle \text{W}_0^{2\pi}\left\{\psi_1 + \frac{\pi}{2}\right\} - \text{W}_0^{2\pi}\left\{\psi_{12} + \frac{\pi}{2}\right\}\right\rangle$$

$$+ R\text{W}_0^{2\pi}\left\langle \text{W}_0^{2\pi}\left\{\psi_{12} + \frac{\pi}{2}\right\} - \text{W}_0^{2\pi}\left\{\psi_2 + \frac{\pi}{2}\right\}\right\rangle \qquad \textit{Eq. 2.13}$$

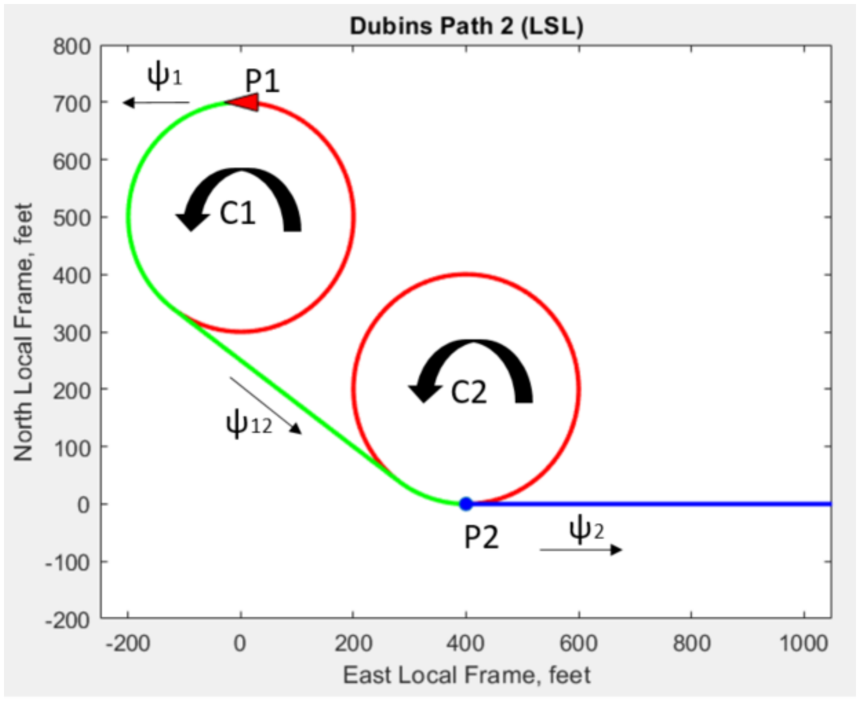


Figure 9: Dubins Path Architecture Type 2 (LSL) Example

The Dubins Path "Type 3" architecture involves a clockwise circle, which exits on a tangential line segment, and enters onto a counter-clockwise arc. This architecture is also referred to as an "RSL" architecture. Circles C_1 and C_2 are constructed according to their respective arc orientations, as shown in Equations 2.14 and 2.15. The heading of the tangent line between the circles can be determine by using the heading between the centers (ψ_{12}) and trigonometric relations between the radii and the distance between the centers (C_{12}), as shown in Equations 2.16, 2.17, and 2.18. The total length of the path can then be calculated by summing the length of the tangent line (determined from trigonometric relations between the radii and the distance between the centers) and the length of the arcs, shown in Equation 2.19. Note that this Type 3 architecture does not exist for conditions where the distance between circles C_1 and C_2

is smaller than twice the radius R_{Min}, as Equation 2.18 would produce an imaginary result, or rather, the circles C_1 and C_2 would overlap and an inner tangent between the circles would not exist. This existence criteria is shown below in Equation 2.20. An example scenario for a Dubins Path of architecture Type 3 is shown in Figure 10.

$$C_1 = P_1 + R\left\langle cos\left(\psi_1 + \frac{\pi}{2}\right),\ sin\left(\psi_1 + \frac{\pi}{2}\right)\right\rangle \qquad \textit{Eq. 2.14}$$

$$C_2 = P_2 + R\left\langle cos\left(\psi_2 - \frac{\pi}{2}\right),\ sin\left(\psi_2 - \frac{\pi}{2}\right)\right\rangle \qquad \textit{Eq. 2.15}$$

$$\psi_{12} = atan2\left(\frac{C_{2E}-C_{1E}}{C_{2N}-C_{1N}}\right) \qquad \textit{Eq. 2.16}$$

$$C_{12} = \|C_2 - C_1\| \qquad \textit{Eq. 2.17}$$

$$\psi_{Tan} = \psi_{12} + sin^{-1}\left(\frac{2R}{C_{12}}\right) \qquad \textit{Eq. 2.18}$$

$$L_3 = \sqrt{C_{12}^2 - 4R^2} + R\text{Ш}_0^{2\pi}\left\langle \text{Ш}_0^{2\pi}\left\{\psi_{Tan} - \frac{\pi}{2}\right\} - \text{Ш}_0^{2\pi}\left\{\psi_1 - \frac{\pi}{2}\right\}\right\rangle$$

$$+R\text{Ш}_0^{2\pi}\left\langle \text{Ш}_0^{2\pi}\left\{\psi_{Tan} + \frac{\pi}{2}\right\} - \text{Ш}_0^{2\pi}\left\{\psi_2 + \frac{\pi}{2}\right\}\right\rangle \qquad \textit{Eq. 2.19}$$

$$L_3\ DNE\ if\ (C_{12} < 2R) \qquad \textit{Eq. 2.20}$$

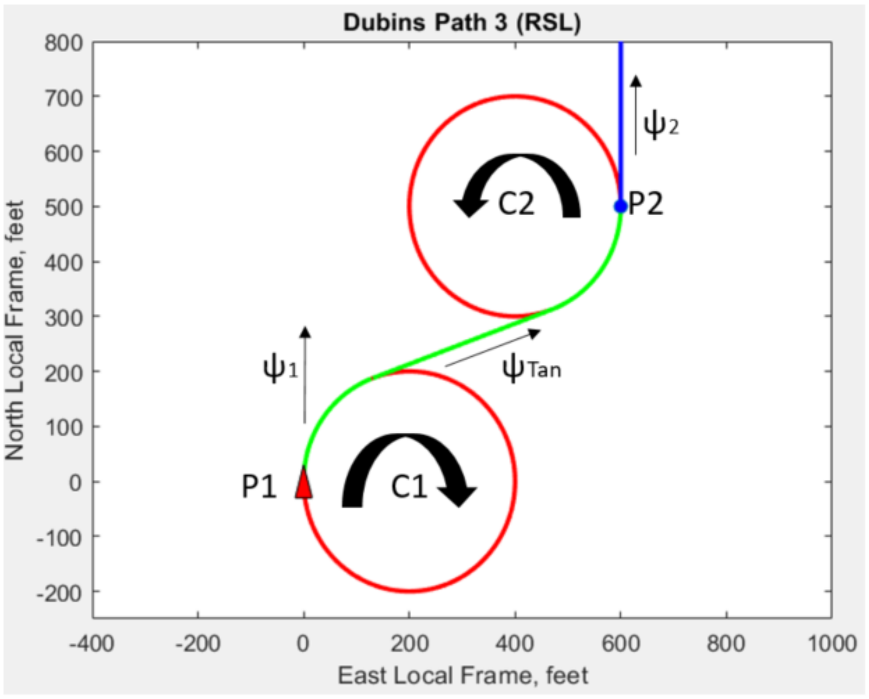


Figure 10: Dubins Path Architecture Type 3 (RSL) Example

The Dubins Path "Type 4" architecture involves a counter-clockwise circle, which exits on a tangential line segment, and enters onto a clockwise arc. This architecture is also referred to as an "LSR" architecture. Circles C_1 and C_2 are constructed according to their respective arc orientations, as shown in Equations 2.21 and 2.22. The heading of the tangent line between the circles, the total length of the path, and the existence criteria of Type 4 can be determine using similar trigonometric relations with Type 3 but modified for the respective orientations of the circles, shown below in Equations 2.23-2.27. An example scenario for a Dubins Path of architecture Type 4 is shown in Figure 11***.***

$$C_1 = P_1 + R\left\langle cos\left(\psi_1 - \frac{\pi}{2}\right),\ sin\left(\psi_1 - \frac{\pi}{2}\right)\right\rangle \qquad \textit{Eq. 2.21}$$

$$C_2 = P_2 + R\left\langle cos\left(\psi_2 + \frac{\pi}{2}\right),\ sin\left(\psi_2 + \frac{\pi}{2}\right)\right\rangle \qquad \textit{Eq. 2.22}$$

$$\psi_{12} = atan2\left(\frac{C_{2E} - C_{1E}}{C_{2N} - C_{1N}}\right) \qquad \textit{Eq. 2.23}$$

$$C_{12} = \|C_2 - C_1\| \qquad Eq.\ 2.24$$

$$\psi_{Tan} = \psi_{12} - sin^{-1}\left(\frac{2R}{C_{12}}\right) \qquad Eq.\ 2.25$$

$$L_4 = \sqrt{C_{12}^2 - 4R^2} + R\mathrm{W}_0^{2\pi}\left\langle \mathrm{W}_0^{2\pi}\left\{\psi_1 + \frac{\pi}{2}\right\} - \mathrm{W}_0^{2\pi}\left\{\psi_{Tan} + \frac{\pi}{2}\right\}\right\rangle$$

$$+R\mathrm{W}_0^{2\pi}\left\langle \mathrm{W}_0^{2\pi}\left\{\psi_2 - \frac{\pi}{2}\right\} - \mathrm{W}_0^{2\pi}\left\{\psi_{Tan} - \frac{\pi}{2}\right\}\right\rangle \qquad Eq.\ 2.26$$

$$L_4\ DNE\ if\ (C_{12} < 2R) \qquad Eq.\ 2.27$$

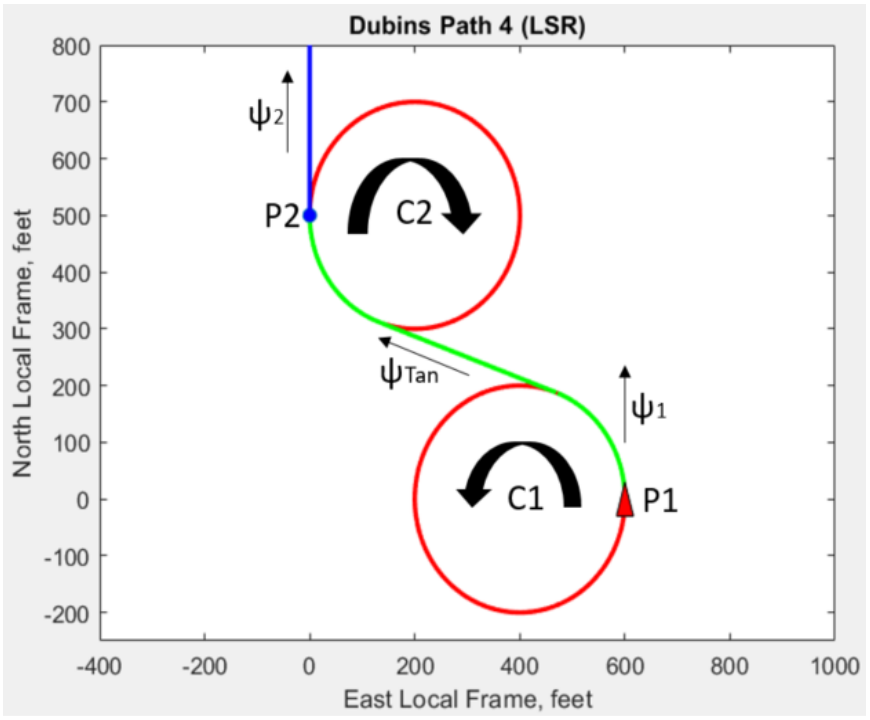


Figure 11: Dubins Path Architecture Type 4 (LSR) Example

The Dubins Path “Type 5” architecture involves three arc segments, beginning with a counter-clockwise circle, transitioning onto a clockwise circle, then back onto a counter-clockwise circle that has a tangent intersecting with the start of the flight line. This architecture is also referred to as an “LRL” architecture, and can be utilized for tight maneuvers between closely-spaced flight lines, as is common in polar research applications (Section 1.4.2 and Section 2.1). Circles C_1 and C_2 are constructed according to their CCW orientations, as shown in Equations 2.28 and 2.29, while circle C_3 is positioned an equal distance 2R from circles C_1 and C_2, using methods given below in Equations 2.30-2.33. The length of the path is then determined by a summation of the three arc lengths, shown in Equation 2.34. Note that path Type 5 does not exist if distance between circles C_1 and C_2 is greater than four times the radius R_{Min}, as it would violate the domain of the inverse cosine function in Equation 2.32, or rather, it would not be possible to intersect the two circles with a third circle of the similar radius. A larger circle could be used to complete the architecture, but this would not result in an optimal path. This existence criteria is shown below in Equation 2.35. An example scenario for a Dubins Path of architecture Type 5 is shown in Figure 12***.***

$$C_1 = P_1 + R\left\langle cos\left(\psi_1 - \frac{\pi}{2}\right),\ sin\left(\psi_1 - \frac{\pi}{2}\right)\right\rangle \qquad \textit{Eq. 2.28}$$

$$C_2 = P_2 + R\left\langle cos\left(\psi_2 - \frac{\pi}{2}\right),\ sin\left(\psi_2 - \frac{\pi}{2}\right)\right\rangle \qquad \textit{Eq. 2.29}$$

$$\psi_{12} = atan2\left(\frac{\mathrm{C_{2E}-C_{1E}}}{\mathrm{C_{2N}-C_{1N}}}\right) \qquad \textit{Eq. 2.30}$$

$$C_{12} = \|C_2 - C_1\| \qquad \textit{Eq. 2.31}$$

$$\psi_{13} = \psi_{12} - \cos^{-1}\left(\frac{\mathrm{C_{12}}}{\mathrm{4R}}\right) \qquad \textit{Eq. 2.32}$$

$$\mathrm{C_3 = C_1 + 2R\langle\cos(\psi_{13}),\ \sin(\psi_{13})\rangle} \qquad \textit{Eq. 2.33}$$

$$\mathrm{L_5} = \mathrm{R}\begin{pmatrix} \left|\mathrm{W}^{2\pi}{}_0\{atan2(\mathrm{P_1 - C_1}) - atan2(\mathrm{C_3 - C_1})\}\right| \\ + \left|\mathrm{W}^{2\pi}{}_0\{atan2(\mathrm{C_2 - C_3}) - atan2(\mathrm{C_1 - C_3})\}\right| \\ + \left|\mathrm{W}^{2\pi}{}_0\{atan2(\mathrm{C_3 - C_2}) - atan2(\mathrm{P_2 - C_2})\}\right| \end{pmatrix} \qquad \textit{Eq. 2.34}$$

$$L_5\ DNE\ if\ (C_{12} < 4R) \qquad \textit{Eq. 2.35}$$

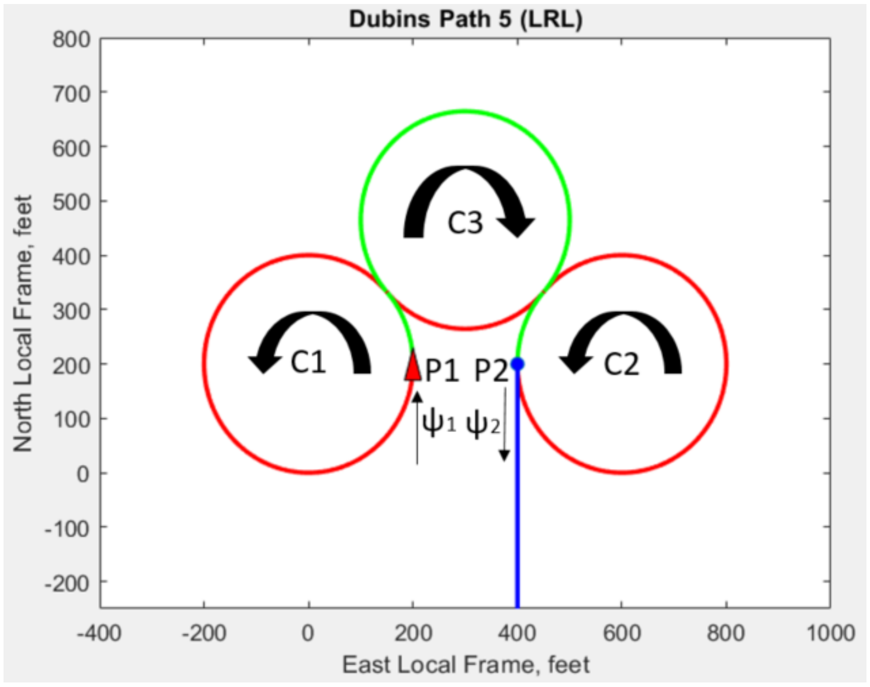


Figure 12: Dubins Path Architecture Type 5 (LRL) Example

Finally, the Dubins Path "Type 6" architecture involves three arc segments, beginning with a clockwise circle, transitioning onto a counter-clockwise circle, then back onto a clockwise circle that has a tangent intersecting with the start of the flight line. This architecture is also referred to as an "RLR" architecture, and can similarly be utilized for tight maneuvers between closely-spaced flight lines. Circles C_1 and C_2 are constructed according to their CW orientations, as shown in Equations 2.36 and 2.37. Circle C_3 is constructed in Equations 2.38-2.41 using similar methods used for Type 5, although the directions are modified for the respective circle orientations. The length for the total path is again the summation of the three arc lengths composing the path (Equation 2.42), and similar existence criterion to Type 5 is defined in Equation 2.43. An example scenario for a Dubins Path of architecture Type 6 is shown in Figure 13.

$$C_1 = P_1 + R\left\langle cos\left(\psi_1 + \frac{\pi}{2}\right),\ sin\left(\psi_1 + \frac{\pi}{2}\right)\right\rangle \qquad \textit{Eq. 2.36}$$

$$C_2 = P_2 + R\left\langle cos\left(\psi_2 + \frac{\pi}{2}\right),\ sin\left(\psi_2 + \frac{\pi}{2}\right)\right\rangle \qquad \textit{Eq. 2.37}$$

$$\psi_{12} = atan2\left(\frac{C_{2E}-C_{1E}}{C_{2N}-C_{1N}}\right) \qquad \textit{Eq. 2.38}$$

$$C_{12} = \|C_2 - C_1\| \qquad \textit{Eq. 2.39}$$

$$\psi_{13} = \psi_{12} + \cos^{-1}\left(\frac{C_{12}}{4R}\right) \qquad \textit{Eq. 2.40}$$

$$C_3 = C_1 + 2R\langle\cos(\psi_{13}),\ \sin(\psi_{13})\rangle \qquad \textit{Eq. 2.41}$$

$$L_6 = R\begin{pmatrix} \left|W^{2\pi}_{0}\{atan2(C_3 - C_1) - atan2(P_1 - C_1)\}\right| + \\ \left|W^{2\pi}_{0}\{atan2(C_1 - C_3) - atan2(C_2 - C_3)\}\right| \\ + \left|W^{2\pi}_{0}\{atan2(P_2 - C_2) - atan2(C_3 - C_2)\}\right| \end{pmatrix} \qquad \textit{Eq. 2.42}$$

$$L_6\ DNE\ if\ (C_{12} < 4R) \qquad \textit{Eq. 2.43}$$

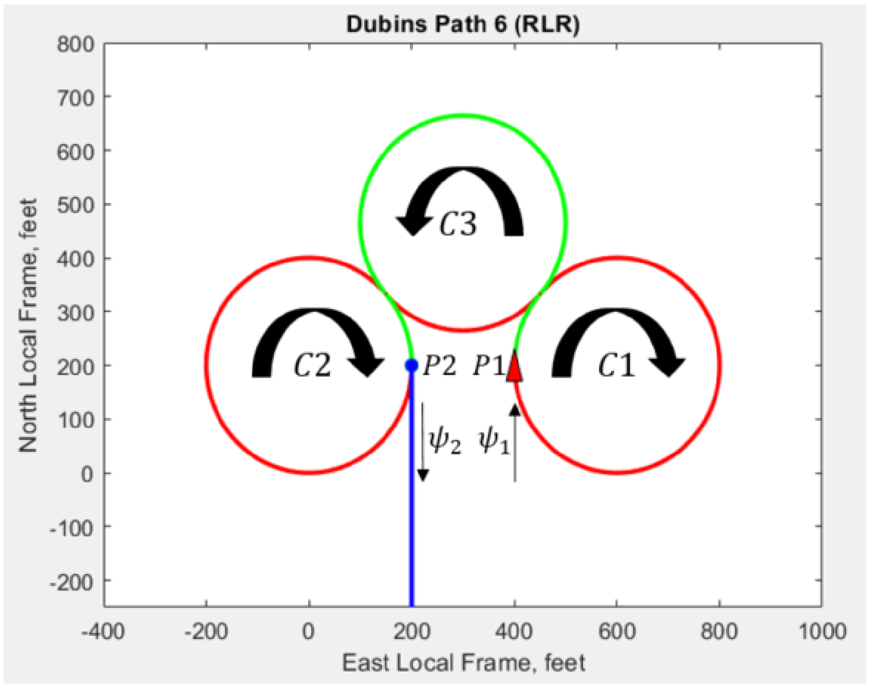


Figure 13: Dubins Path Architecture Type 6 (RLR) Example

Following construction of all Dubins path candidates, the optimal path can be determined by selecting the minimum length path from the existing candidates, as shown in Equation 2.44. Note that non-existent paths architectures would be removed from the array of valid candidate paths. An example scenario for

$P_1 = \{0,0\}$, $\psi_1 = 0, P_2 = \{50,550\}$, $\psi_2 = \frac{\pi}{2}$, and $R_{Min} = 200$, is given below in Figure 14, where each of the path architectures are generated. Note that this example is a rare situation in which all six paths exist. Table 1 shows the lengths of each of these candidate paths, where it can be seen that Path 3 is the optimal choice.

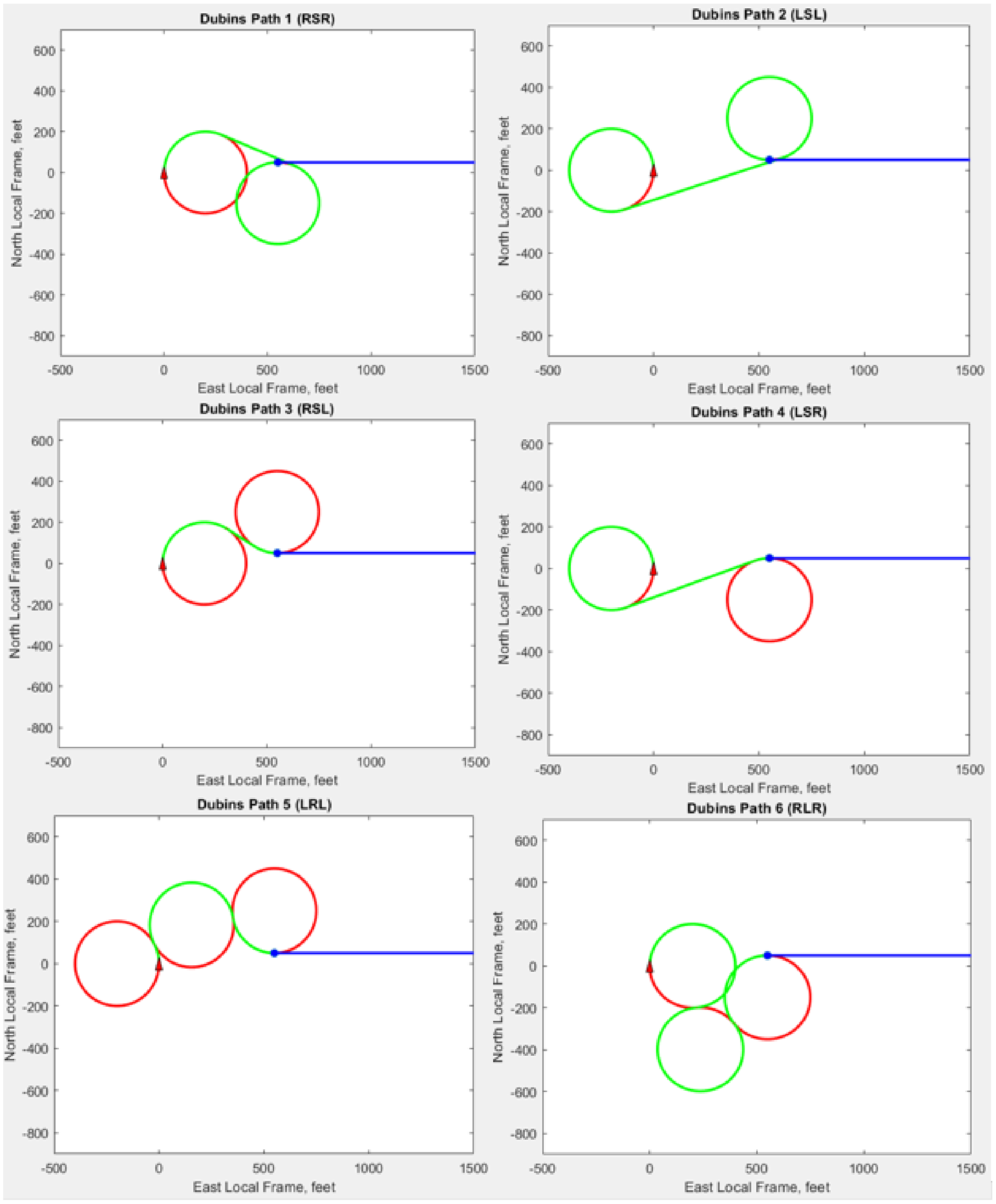


Figure 14: Dubins Example Scenario - All Candidate Paths

$$L_{Optimal} = min\{L_1, L_2, L_3, L_4, L_5, L_6\} \quad \text{Eq. 2.44}$$

Table 1: Dubins Example Scenario - All Candidate Path Lengths

Path Type	Path 1 (RSR)	Path 2 (LSL)	Path 3 (RSL)	Path 4 (LSR)	Path 5 (LRL)	Path 6 (RLR)
Total Distance (feet)	1,952	2,989	702	1,736	1,065	2,431

2.2.3 Home Loiter Modifications

As mission operations for fixed-wing aircraft typically involve the use of a “home loiter” circle near the takeoff/landing area, the entrance into these loiters ideally should also follow an optimal path in order to increase operational efficiency. These optimal paths can be constructed via a simple modification of the Dubins path algorithms provided previously. Shown in an example below in Figure 15, this problem involves creating an optimal path from the aircraft position ($\mathrm{P}_1 = \{\mathrm{P}_{1N}, \mathrm{P}_{1E}\}$) and heading ($\psi_1$) that merges onto a circle centered around coordinate $\mathrm{P}_2 = \{\mathrm{P}_{2N}, \mathrm{P}_{2E}\}$, which radius R_{Min}, where the curvature of the path is also kinematically constrained by the aircraft minimum turning radius.

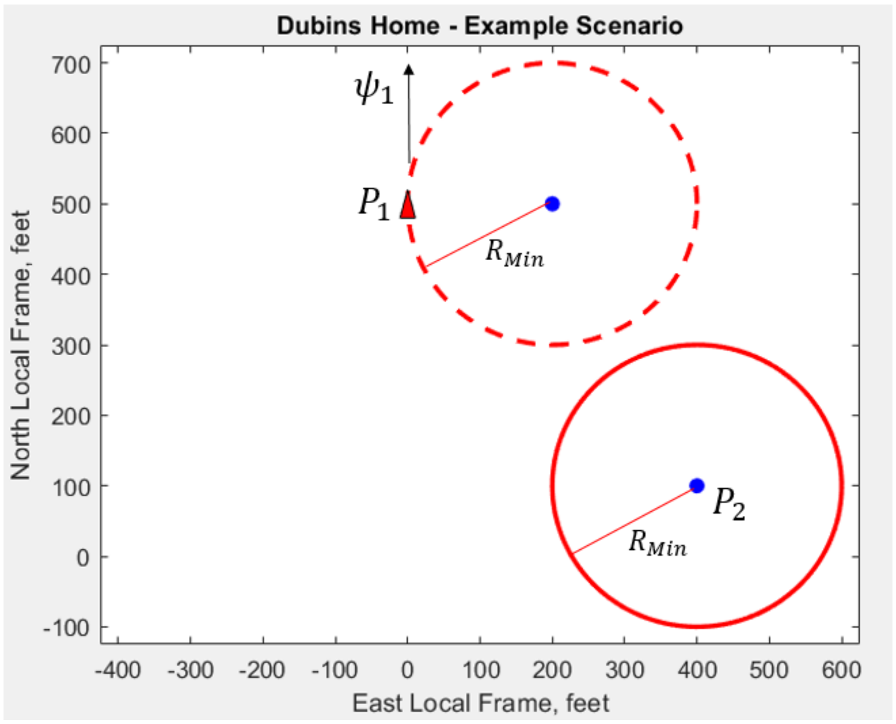


Figure 15: Dubins Home – Example Scenario

The Type 1 architecture for Dubins Paths introduced earlier can be modified for this application, by simply replacing circle center C_2 with the desired loiter center P_2. This architecture utilizes a clockwise arc, tangents onto a line segment, and then begins to loiter around point P_2 in a clockwise direction. The equations for constructing this path are shown below in Equations 2.45-2.49, where θ_1 indicates the angle of the required arc around circle C_1, which will be used in the path selection criteria. This criteria is selected in order to prioritize smooth transitions onto the loiter circle during flight operations. Note that this Type 1 architecture exists for all scenarios.

$$C_1 = P_1 + R\left\langle cos\left(\psi_1 + \frac{\pi}{2}\right),\ sin\left(\psi_1 + \frac{\pi}{2}\right)\right\rangle \qquad \textit{Eq. 2.45}$$

$$C_2 = P_2 \qquad \textit{Eq. 2.46}$$

$$\psi_{12} = atan2\left(\frac{C_{2E} - C_{1E}}{C_{2N} - C_{1N}}\right) \qquad \textit{Eq. 2.47}$$

$$L_1 = \|C_2 - C_1\| + R\mathrm{W}_0^{2\pi}\left\langle \mathrm{W}_0^{2\pi}\left\{\psi_{12} - \frac{\pi}{2}\right\} - \mathrm{W}_0^{2\pi}\left\{\psi_1 - \frac{\pi}{2}\right\}\right\rangle \qquad \textit{Eq. 2.48}$$

$$\theta_1 = \mathrm{W}_0^{2\pi}\left\langle \mathrm{W}_0^{2\pi}\left\{\psi_{12} - \frac{\pi}{2}\right\} - \mathrm{W}_0^{2\pi}\left\{\psi_1 - \frac{\pi}{2}\right\}\right\rangle \qquad \textit{Eq. 2.49}$$

The Type 2 architecture for Dubins Paths can be similarly modified, as shown below in Equations 2.50-2.54. This architecture utilizes a counter-clockwise arc, tangents onto a line segment, and then begins to loiter around point P_2 in a counter-clockwise direction. Note that this Type 2 architecture exists for all scenarios.

$$C_1 = P_1 + R\left\langle cos\left(\psi_1 - \frac{\pi}{2}\right),\ sin\left(\psi_1 - \frac{\pi}{2}\right)\right\rangle \qquad \textit{Eq. 2.50}$$

$$C_2 = P_2 \qquad \textit{Eq. 2.51}$$

$$\psi_{12} = atan2\left(\frac{C_{2E} - C_{1E}}{C_{2N} - C_{1N}}\right) \qquad \textit{Eq. 2.52}$$

$$L_2 = \|C_2 - C_1\| + R\mathrm{W}_0^{2\pi}\left\langle \mathrm{W}_0^{2\pi}\left\{\psi_1 + \frac{\pi}{2}\right\} - \mathrm{W}_0^{2\pi}\left\{\psi_{12} + \frac{\pi}{2}\right\}\right\rangle \qquad \textit{Eq. 2.53}$$

$$\theta_2 = \mathrm{W}_0^{2\pi}\left\langle \mathrm{W}_0^{2\pi}\left\{\psi_1 + \frac{\pi}{2}\right\} - \mathrm{W}_0^{2\pi}\left\{\psi_{12} + \frac{\pi}{2}\right\}\right\rangle \qquad \textit{Eq. 2.54}$$

Similarly, the Type 3 architecture for Dubins Paths can be similarly modified, as shown below in Equations 2.55-2.62. This architecture utilizes a clockwise arc, tangents onto a line segment, and then begins to loiter around point P_2 in a counter-clockwise direction. Note that this Type 3 architecture does not exists for scenarios in which the distance between the circles C_1 and C_2 is greater than twice the radius R_{Min}.

$$C_1 = P_1 + R\left\langle cos\left(\psi_1 + \frac{\pi}{2}\right),\ sin\left(\psi_1 + \frac{\pi}{2}\right)\right\rangle \qquad \textit{Eq. 2.55}$$

$$C_2 = P_2 \qquad \textit{Eq. 2.56}$$

$$\psi_{12} = atan2\left(\frac{C_{2E} - C_{1E}}{C_{2N} - C_{1N}}\right) \qquad \textit{Eq. 2.57}$$

$$C_{12} = \|C_2 - C_1\| \quad \text{Eq. 2.58}$$

$$\psi_{Tan} = \psi_{12} + sin^{-1}\left(\frac{2R}{C_{12}}\right) \quad \text{Eq. 2.59}$$

$$L_3 = \sqrt{C_{12}^2 - 4R^2} + R\mathrm{W}_0^{2\pi}\left\langle \mathrm{W}_0^{2\pi}\left\{\psi_{Tan} - \frac{\pi}{2}\right\} - \mathrm{W}_0^{2\pi}\left\{\psi_1 - \frac{\pi}{2}\right\}\right\rangle \quad \text{Eq. 2.60}$$

$$\theta_3 = \mathrm{W}_0^{2\pi}\left\langle \mathrm{W}_0^{2\pi}\left\{\psi_{Tan} - \frac{\pi}{2}\right\} - \mathrm{W}_0^{2\pi}\left\{\psi_1 - \frac{\pi}{2}\right\}\right\rangle \quad \text{Eq. 2.61}$$

$$L_3, \theta_3 \; DNE \; if \; (C_{12} < 2R) \quad \text{Eq. 2.62}$$

Finally, the Type 4 architecture for Dubins Paths can be similarly modified, as shown below in Equations 2.63-2.70. This architecture utilizes a counter-clockwise arc, tangents onto a line segment, and then begins to loiter around point P_2 in a clockwise direction. Note that this Type 4 architecture does not exists for scenarios in which the distance between the circles C_1 and C_2 is greater than twice the radius R_{Min}.

$$C_1 = P_1 + R\left\langle cos\left(\psi_1 - \frac{\pi}{2}\right),\; sin\left(\psi_1 - \frac{\pi}{2}\right)\right\rangle \quad \text{Eq. 2.63}$$

$$C_2 = P_2 \quad \text{Eq. 2.64}$$

$$\psi_{12} = atan2\left(\frac{C_{2E} - C_{1E}}{C_{2N} - C_{1N}}\right) \quad \text{Eq. 2.65}$$

$$C_{12} = \|C_2 - C_1\| \quad \text{Eq. 2.66}$$

$$\psi_{Tan} = \psi_{12} - sin^{-1}\left(\frac{2R}{C_{12}}\right) \quad \text{Eq. 2.67}$$

$$L_4 = \sqrt{C_{12}^2 - 4R^2} + R\mathrm{W}_0^{2\pi}\left\langle \mathrm{W}_0^{2\pi}\left\{\psi_1 + \frac{\pi}{2}\right\} - \mathrm{W}_0^{2\pi}\left\{\psi_{Tan} + \frac{\pi}{2}\right\}\right\rangle \quad \text{Eq. 2.68}$$

$$\theta_4 = \mathrm{W}_0^{2\pi}\left\langle \mathrm{W}_0^{2\pi}\left\{\psi_1 + \frac{\pi}{2}\right\} - \mathrm{W}_0^{2\pi}\left\{\psi_{Tan} + \frac{\pi}{2}\right\}\right\rangle \quad \text{Eq. 2.69}$$

$$L_4, \theta_4 \; DNE \; if \; (C_{12} < 2R) \quad \text{Eq. 2.70}$$

While it is possible to generate Dubins Paths of architecture Types 5 and 6 for entering onto a home loiter circle, the paths would never be an optimal choice as unless there is a required entrance point and loiter direction, which does not apply to this application, and as such is not demonstrated here.

An example is given below in Figure 16 in which an aircraft at location $\mathrm{P}_1 = \{500{,}0\}$ and heading $\psi_1 = 0$ must enter into a loiter about position $\mathrm{P}_2 = \{100{,}400\}$ with radius $R_{Min} = 200$. Note that this is a unique scenario in which Types 1-4 all exist. Table 2 displays the resulting lengths and arcs for each of the paths, and Equation 2.71 shows that the criteria for selecting the optimal path in this work is not the minimal length path, but the path requiring the least angular arc of circle C_1. This prevents unnecessary maneuvers in flight operations and smoother transitions into the loiter. Note that in this example, while Type 3 is the minimal length path, Type 1 requires less angular tracking of circle C_1, and is thus our optimal path choice.

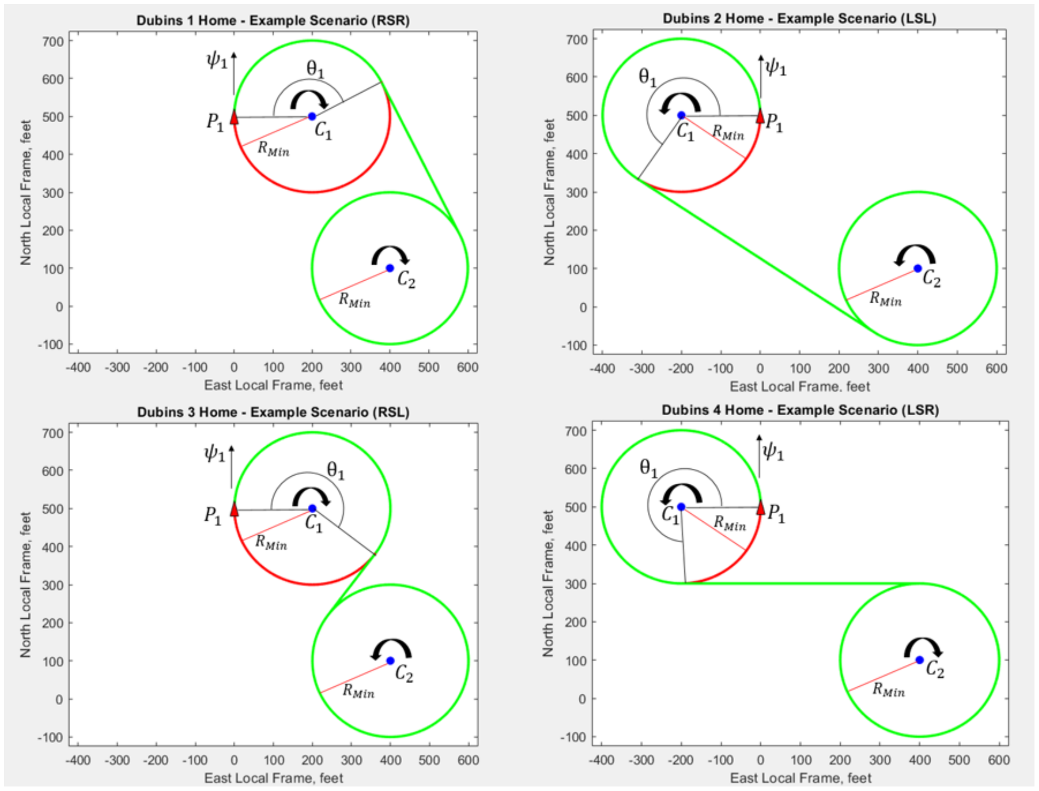


Figure 16: Dubins Home Example - Candidate Paths

Table 2: Dubins Home Example Analysis

Path Type	1	2	3	4
Length of Path (L), feet	983	1,546	957	1,543
Arc 1 Angle (θ_1), degrees	153	236	216	270

$$\theta_{Optimal} = min\{\theta_1, \theta_2, \theta_3, \theta_4\} \quad \textit{Eq. 2.71}$$

However, in some operations it may be desirable to loiter around a location at a radius not necessarily equal to the minimum turning radius of the aircraft. Namely for this application, it may be desirable to transition onto and between flight lines using the minimum turning radius R_{Min}, but to use a larger radius for the home loiter in order to reduce the required steady state bank angle during the loiter. Similar procedure as before can be used for the Dubins Paths onto the loiters, however the calculation for the tangent heading between the circles C_1 and C_2 needs to be determined using Direct Common Tangents (for Types 1 and 2) and Transverse Common Tangents (for Types 3 and 4).

Direct common tangents between two circles of unique radii can be determine by first determining the so called "Focus Point" of the circles, via Equation 2.72. Figure 17 shows an example of direct common tangents, where the focus point F_P can be used to determine the tangent headings (shown in green) of segments A_1A_2 (for Type 1) or B_1B_2 (for Type 2) architectures if the radius of the home loiter location is larger than the minimum turning radius of the aircraft. The tangent headings can be determined using Equation 2.73 (with sign depending on Type 1/2 architectures), and similar procedure as before can be used to determine the lengths of the paths and the required angular arcs around circle C_1. Note that these paths will exist in all scenarios in which the smaller circle is not fully encompassed inside of the larger circle.

$$F_P = \left\{\frac{C_2R_1 - C_1R_2}{R_1 - R_2}\right\} \qquad \textit{Eq. 2.72}$$

$$\psi_{12} = atan2\left(\frac{C_{2E} - C_{1E}}{C_{2N} - C_{1N}}\right) +/- \ sin^{-1}\left(\frac{R_1}{|C_1 - F_P|}\right) \qquad \textit{Eq. 2.73}$$

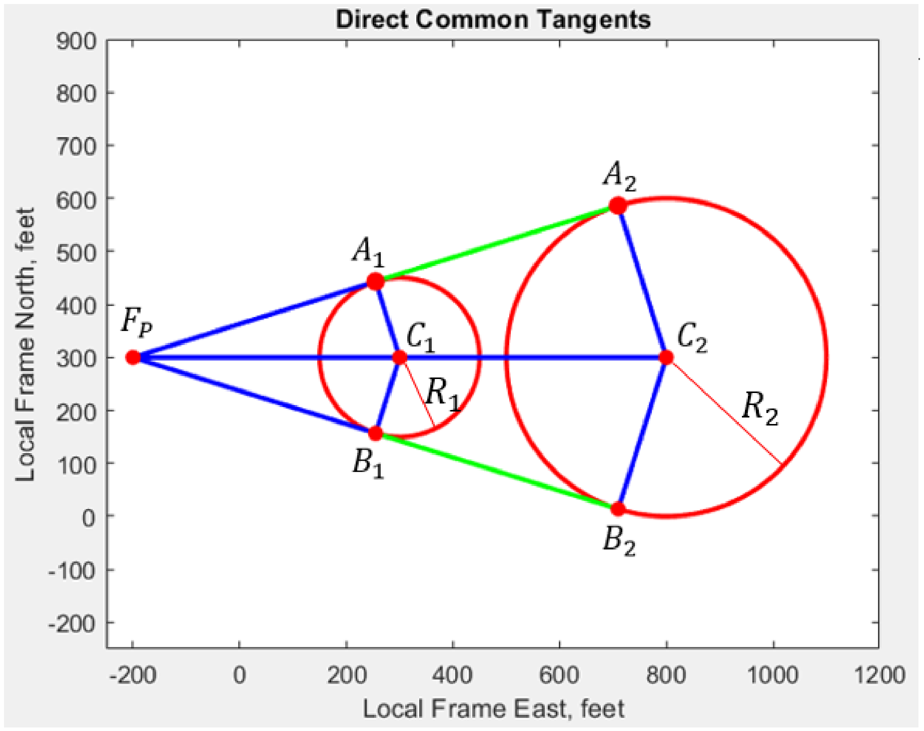


Figure 17: Direct Common Tangents

Similarly, Dubins Paths Types 3 and 4 can be constructed for arbitrary radii using transverse common tangents, as shown in Figure 18. The location of the focus point can be determined using Equation 2.74, while the heading of the tangent line can be determined using Equation 2.75 (with sign depending on Type 3/4 architectures). Note that the existence criteria for Types 3 and 4 are a function of radii R_1 and R_2, as shown in Equation 2.76.

$$F_P = \left\{\frac{C_2R_1 + C_1R_2}{R_1 + R_2}\right\} \qquad \textit{Eq. 2.74}$$

$$\psi_{12} = atan2\left(\frac{C_{2E} - C_{1E}}{C_{2N} - C_{1N}}\right) +/- \ sin^{-1}\left(\frac{R_2}{|C_2 - F_P|}\right) \qquad \textit{Eq. 2.75}$$

$$L_{3,4}, \theta_{3,4}\ DNE\ if\ respective\ C_{12} < (R_1 + R_2) \qquad \text{Eq. 2.76}$$

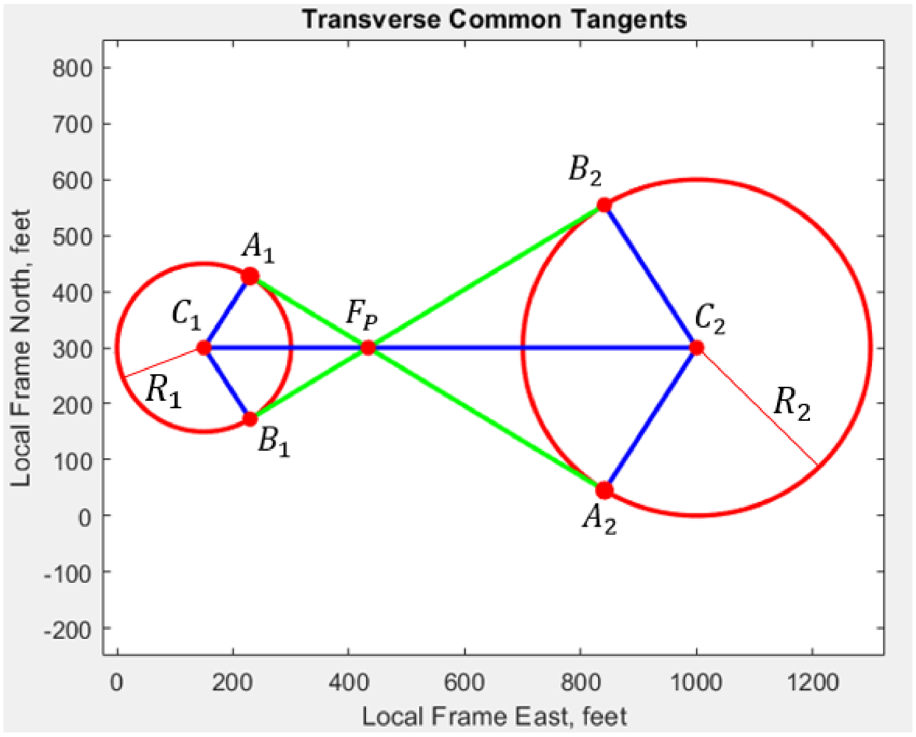


Figure 18: Transverse Common Tangents

Once all possible paths are constructed using this methodology for arbitrary radii, similar criteria as before is used to determine the optimal path for the aircraft to follow onto the loiter circle.

2.3 TSP Heuristics

Intelligent route planning through the desired survey flight lines can be used to significantly increase mission efficiency for polar research operations. The order in which the flight lines are traversed can directly determine the required operational time, and therefore an optimal path through the flight lines is desired. However, conventional mentions for these missions simply consist of a pre-planned route determined by the human operators based on personal intuition, limiting the efficiency of the routes with respect to more optimal solutions. Solving for more efficient paths using automated methods could help further reduce operator workload to simply selecting the desired flight lines along the glacier for which data collection is desirable. The automated routing methods developed in this work can be utilized to run offline prior to the mission deployment and hard-coded into fixed, ordered waypoints for a COTS autopilot systems. However, in this work these routing algorithms are utilized onboard the aircraft in real-time to provide the capabilities for intelligent mission re-planning in order to adapt to changing mission elements.

2.3.1 Traveling Salesman Problem

The optimal mission route can be determined by finding the order in which to survey the flight lines that results in the shortest length path. However, determining this order introduces one of the most well-known and extensively studied transportation problems, known as the "Traveling Salesman Problem" (TSP). In this problem, a salesman must travel to each of a list of "N" number of cities exactly once and must do so by taking the shortest path [51,52]. The naïve solution to this problem is to simply generate every possible permutation of the N cities and to find the route with the shortest distance. However, this strategy quickly becomes impractical for anything more than a handful of cities, as the total possible permutations to search grows rapidly as the factorial of N. For this reason, TSP is categorized as a combinational optimization problem with an NP-hard complexity, denoting that a truly optimal solution cannot be computed in polynomial time. However, mathematicians have devised many heuristics that can be utilized to quickly obtain approximate solutions to the global optimal. As the performance of these heuristics typically depends on their computational run times, this work will apply several different heuristics and analyze the tradeoffs between their results and required run times, and assess their suitability for real time UAS applications.

The traditional TSP problem can be solved by viewing the list of cities as a complete, edge-weighted graph, G = {V,E}, where the vertices are the city locations and the edges correspond to roads connecting the cities. The weighting applied to each edge in the graph can directly correspond to the distance of the road between the respective cities, denoting the "cost" of traveling between the respective cities. The optimal solution of the TSP can then be reduced to finding the minimum length Hamiltonian path through the graph, along which each vertex is visited exactly once.

However, several modifications are made in this work in order to account for the differences between traditional TSP (involving distinct locations to travel to) and polar research applications (involving flight lines to survey). The cities populating the vertices of the graph are replaced by the series of waypoint pairs constructing the desired flight lines $V = \{W_{11}, W_{12} \dots, W_{N1}, W_{N2}\}$, where the first subscript represents the line number, and the seconds subscript represents the ordered pair. Each of these individual waypoints can be described by a North and East component: $W_i = \{W_{i_N}, W_{i_E}\}$. The edge weightings between these waypoints is determined by calculating the length of the Dubins Path (see Section 2.2) between the respective waypoints, with initial and terminal headings corresponding to their respective flight line headings. As shown in Equation 2.76, the edge cost E_{ij} from W_i to W_j is calculated using the length of the Dubins Path D_{ij} resulting from W_i to W_j, with initial heading exiting the flight line corresponding to W_i, and terminal heading entering onto the flight line corresponding to W_j. If however, W_i and W_j are pairs constructing a similar flight line, the edge cost is set to zero in order to constrain the route to incorporate the end-to-end flight line surveys. That is, graph exploration traverses onto a new waypoint vertex, its next immediate vertex is constrained to be the other waypoint in the pair constructing the flight line. Note that for a series of "N" flight lines, there will be 2N waypoints in the graph, and by including this waypoint-pair constraint, the total number of possible permutations for the waypoint orderings is on the order of $2^N(N!)$. A final constraint is imposed that the route must begin at the current aircraft location and finish at the home waypoint, both are which are treated as an initial and terminal vertex in the graph.

$$E_{ij} = D_{ij} \qquad \textit{Eq. 2.76}$$

2.3.2 Heuristics

Researchers have applied many methods towards solving TSP problems, including both exact and approximate methods. While still taking considerable amounts of computational time, exact methods such as the Held-Karp algorithm [53] have been shown to solve the TSP problem with a computational complexity of $O(n^2 2^n)$, which is a vast improvement from $O(N!)$, but still considerably long for relatively large N. On the other hand, approximate methods, or heuristics, have been shown to quickly produce near-optimal solutions suitable from an engineering standpoint. These heuristics include dynamic programming, genetic algorithms, neural networks, tabu search, and simulated annealing, among many others [54-56].

Approximate methods for solving TSP can be further divided into two sub-categories with various benefits and drawbacks: Deterministic and Stochastic methods. Deterministic methods can be used to quickly and reliably produce repeatable solutions to the problem, producing a “good” estimate of the optimal solution in a reasonable computational time. However, the performance of a deterministic method can be constrained by their respective search methods getting trapped in local optima. On the other hand, stochastic methods utilize probabilistic search methods and random exploration in order to extensively explore the search space. These methods have the potential to find better solutions than deterministic methods, although they generally have longer computational run times, and do not guarantee a “good” solution as an end-result. However, utilizing both method types in parallel could result in achieving an acceptable baseline solution from the deterministic method while also possibly finding an even better solution from the stochastic method. In this work, two variations of Greedy heuristics have been implemented in order to quickly obtain baseline deterministic TSP solutions, while Ant Colony Optimization is utilized for its stochastic exploration potential.

2.3.3 Forward Greedy Heuristic

The first heuristic utilized in this work is a Forward Greedy method, which will be used for its simplicity, reliability, and quick computation runtime. This heuristic will utilize "greedy" methods, in which the local optima is iteratively chosen in order to approximate the global optima. This heuristic begins at the current aircraft position, and iteratively adds the nearest flight line to create the route. Note that this method is also referred to as the "nearest-neighbor" heuristic in traditional TSP methods, where it is likewise used as a benchmark to describe the relative performance improvements that can be obtained by more complex and computationally costly methods. The pseudocode for the Forward Greedy heuristic is given below.

```
Forward-Greedy Pseudocode:
Initialize Route at the Current UAS Location
While Flight Lines Remaining
        Select Closest Flight Line
        Update Route Order
        Update UAS position
        Update Remaining Flight Lines
End While
Return Route Order
```

2.3.4 Global Greedy Heuristic

The second heuristic utilized in this work is referred to as the Global Greedy method, in which greedy methods are utilized with several different initial conditions in order to generate candidate routes. The Global Greedy method begins a candidate route at each of the flight lines in the graph, iteratively connecting the closest vertices to its endpoints until a route is created through the flight lines. Additionally, the Forward Greedy method is utilized to create a candidate route. The final candidate route is created using a Backwards Greedy method, which begins at the home loiter and routes through the flight lines and back to the aircraft initial position using greedy methods. The Global Greedy algorithm then uses the shortest of these (N+2) candidate routes for N flight lines. While more computational expensive than the Forward Greedy method, the Global Greedy method is used in this work to achieve improved results, as different local optima are discovered using different initial conditions for the greedy algorithms. The pseudocode for the Global Greedy heuristic is given below.

```
Global-Greedy Pseudocode
Perform Forward-Greedy Algorithm
for i = 1:N_Flight_Lines
    Initialize Candidate Route at Flight Line i
    While Flight Lines Remaining
        Select Closest Flight Line to Current Route
        Update Candidate Route Order
        Update Remaining Flight Lines
    End While
    Store Candidate Route
End for
Perform Backwards-Greedy Algorithm
Return Minimum Length Route from Candidate Routes
```

2.3.5 Ant Colony Optimization

The third and final heuristic used in this work is Ant Colony Optimization (ACO), which is a biologically-inspired approach that utilizes virtual “ants” that stochastically explore the graph based on “pheromone” levels along edges [57-60]. These pheromones levels affect the probability that the ant will choose the respective edge between the vertices. Initial pheromone levels can be either constructed as uniform, or initialized as the inverse of the Dubins path lengths between the respective edges, as in Equation 2.77. Note that this method initially creates higher pheromones along the shorter routes, aiding the ants in the initial generations of graph exploration. Following an iteration of ant exploration through the graph, the performance of the resulting route is used to update the pheromone levels along the edges that were chosen, as shown in Equation 2.78, where L_k is the length of the current ant's route and L^* is the current best overall route discovered. Future generations of ants will then have a higher likelihood of exploring shorter routes, and after many generations of ant exploration, efficient routes can be determined from very complex graphs. Tuning parameters for ACO include a pheromone reinforcement factor Q and a pheromone decay rate ρ, which is used to prevent the excess buildup of pheromones that could wrongfully constrain the exploration onto a single path (Equation 2.79). The benefit of ACO is that through its probabilistic searching, optimal routes that seem counter-intuitive can be discovered. However, as mentioned in Section 2.3.2, this stochastic method does not guarantee a "good" solution, and typically requires much longer computational time and tuning in order to provide better results than the greedy methods. The pseudocode used for ACO is given below.

$$\sigma_{ij} = \frac{1}{D_{ij}} \qquad \textit{Eq. 2.77}$$

$$\sigma_{ij} = \sigma_{ij} + Q * \frac{L^*}{L_k} \qquad \textit{Eq. 2.78}$$

$$\sigma_{ij} = \sigma_{ij} * (1 - \rho) \qquad \textit{Eq. 2.79}$$

Ant Colony Optimization Pseudocode

```
Initialize Pheromones Levels
For i = 1:N_generations
    For j = 1:N_ants
        Initialize ant
        Perform stochastic ant exploration of graph
        Determine route length
    End For
    Update pheromones along route based on ant route length
    Apply pheromone decay factor to entire graph
```

```
        End For
        Return Route with minimum length
```

2.4 Fuel-Constrained Route Planning

In order to improve mission reliability for over-the-horizon polar research missions, the real-time routing algorithms in this work have been modified to take into account vehicle range constraints, ensuring that the aircraft can safely return to the home loiter location before running out of fuel. In the event that a mission planned for the vehicle cannot feasibly accomplished due to aircraft range limitations, the onboard autonomy should determine what alternative actions to take. In this case, the autonomy will decide which of the flights lines can be feasibly surveyed and what order in which to do so. This capability both provides a safety net for mission planning operators, as well as could reduce mission planning workload altogether by simply inputting all the desired flight lines for a large glacial area and having the aircraft iteratively survey them over a series of flights. However, it should be noted that the safety assurances in a practical application of these methods is directly correlated with the accuracy of an onboard real-time range estimator.

2.4.1 Flight Line Utility Weightings

As the scientific value of the collected data can vary depending on the survey flight line, the relative importance of the flight lines should be considered when making decisions on which subset of lines to survey in fuel-constrained missions. To incorporate this, a "utility" weighting factor has been introduced into the fuel-constrained routing algorithms in this work, describing the relative desirability of the flight lines for data collection. This utility factor is used to augment the cost function used for the graph edges, as shown below in Equation 2.80, where the Dubins path length between W_i and W_j is divided by the utility for the flight line associated with W_j. As the utility weighting factors are equal to or greater than 1, this directs graph exploration towards flight lines of higher utility via a reduction in the costs of the connecting graph edges. Note that the goal of the fuel-constrained TSP algorithms used in this work is not to determine a feasible path that has the most flight lines, *but rather to find a feasible path that captures the highest net utility*. The feasibility of adding a flight line to a route is determined by calculating the new total length of the route and ensuring that it is less than the remaining range estimate. This total length is determined by adding the length of the Dubins path onto each of the flight lines in the route, the length of the flight lines themselves, and the Dubins path from the last flight line to the home loiter circle, as shown below in Equation 2.81.

$$E_{ij} = \frac{D_{ij}}{U_j} \qquad \textit{Eq. 2.80}$$

$$R_{Remaining} \geq \sum_{i=1}^{i=n}\left(D_{i,i-1} + L_i\right) + D_{H_n} \qquad \textit{Eq. 2.81}$$

2.4.2 Fuel-Constrained Forward Greedy Heuristic

The Forward Greedy heuristic described in Section 2.3.3 has been modified to incorporate fuel-constrained mission routing in the pseudocode provided below. In this method, the route begins at the aircraft's current location, and iteratively adds the graph edge with the lowest cost until no more flight lines can be feasibly added to the route without preventing the aircraft from feasibly returning to the home loiter circle within the remaining range constraint.

```
Fuel-Constrained Forward-Greedy Pseudocode:
Modify Graph Edges based on Flight Line Utility Weightings
Initialize Route Order
Initialize Route Utility
Initialize Remaining Range
Initialize Feasible Flight Lines
While Flight Lines Remaining
    Update Remaining Flight Lines based on Remaining Range
    Select Flight Line Based on Minimum Graph Edge Value
```

```
        Update Route Order
        Update Route Utility
        Update UAS position
        Update Remaining Range
        Update Remaining Flight Lines based on Selection
    End While
    Return Route Order
```

2.4.3 Fuel-Constrained Global Greedy Heuristic

The Global Greedy heuristic described in Section 2.3.4 has been modified to incorporate fuel-constrained mission routing in the pseudocode provided below. This approach creates candidate routes using greedy methods following the edge weighting modifications and flight line feasibility assessments, and chooses the candidate route with the highest net utility. Note that for two routes returning similar utility, the shorter of the routes is selected. As before, one candidate route is initialized at each of the respective flight lines, while two additional candidate routes are constructed using a Fuel-Constrained Forward Greedy method and a Fuel-Constrained Backwards Greedy method.

```
Fuel-Constrained Global Greedy Pseudocode:
Modify Graph Edges based on Flight Line Utility Weightings
Perform Fuel-Constrained Forward-Greedy Algorithm
For i = 1:N_Flight_Lines
    Initialize Candidate Route at Flight Line i
    Initialize Route Utility
    Initialize Remaining Range
    Initialize Feasible Flight Lines
    While Flight Lines Remaining
        Update Remaining Flight Lines based on Remaining Range
        Select Flight Line Based on Minimum Graph Edge Value
        Update Candidate Route Order
```

```
                Update Candidate Route Utility
                Update Remaining Flight Lines
                Update Remaining Range
            End While
            Store Candidate Route
        End For
        Perform Fuel-Constrained Backwards-Greedy Algorithm
        Return highest utility Route from all candidate Routes
```

2.4.4 Fuel-Constrained Ant Colony Optimization

The Ant Colony Optimization algorithms described in Section 2.3.5 have likewise been modified to incorporate fuel-constraints and flight line utility weightings, and the pseudocode for this fuel-constrained ACO method can be found below. Notable differences from the standard ACO include the modification of the initial pheromone levels using the flight line utility weightings (Equation 2.82), and using the total route utility (U_k) to update the pheromones based on the current highest route utility achieved (U^*), as shown in Equation 2.83. Similar decay rate is used for the pheromones (Equation 2.84). Note that the ant exploration through the graph is fuel-constrained by updating the feasible flight lines remaining after each selection.

$$\sigma_{ij} = \frac{U_j}{D_{ij}} \qquad \textit{Eq. 2.82}$$

$$\sigma_{ij} = \sigma_{ij} + Q * \frac{U_k}{U^*} \qquad \textit{Eq. 2.83}$$

$$\sigma_{ij} = \sigma_{ij} * (1 - \rho) \qquad \textit{Eq. 2.84}$$

```
Fuel-Constrained Ant Colony Optimization Pseudocode
Initialize Pheromones and Modify using Utility Weightings
Initialize Remaining Range
For i = 1:N_generations
```

```
        For i = 1:N_ants
            Perform range constrained stochastic ant exploration of graph
            Determine overall route utility
        End For
        Update pheromones based on route utilities
    End For
    Return highest utility Route from all candidate Routes
```

2.5 Multi-Agent Collaborative Surveying

Numerous benefits can be achieved by incorporating multi-agent operations into polar research missions. Notably, vast glacial regions could be more rapidly surveyed if multiple vehicles are utilized to compensate for limited sensor footprint and vehicle range constraints. Additionally, standard operations could be accomplished more quickly, which is very advantageous given limited windows of opportunity in polar environments. From an operational cost standpoint, several smaller systems have the capability to significantly outperform a single larger system in terms of data collection, due to the dramatic correlation in unmanned vehicle cost with wingspan. Finally, the overall mission reliability is enhanced by introducing multiple systems into the operations, as a single system failure would not terminate the entire operational capabilities of the research deployment.

However, incorporating multi-agent operations introduces increased complexity into the autonomy framework, namely the intelligent allocation of mission flight lines amongst the agents and integrating collision avoidance methods into aircraft path planning. In this section, the methodology for scalable multi-agent collaborative surveying will be discussed. This methodology involves a decentralized approach for global task assignment, in which each agent uses the information available to it about the other agents positions and mission statuses in order to determine the best course of action.

2.5.1 Hungarian Assignment

The balanced assignment problem is a combinational optimization problem in which N agents must be uniquely assigned to perform N tasks in an optimal way. The assignment problem can be viewed in a mathematical sense as determining the matching with minimum edge weight sum of a bipartite weighted graph [61]. As there are (N!) number of possible assignments, determining the optimal solution using a brute force method is generally not practical. In this work, Hungarian Assignment methods will be used in order to achieve optimal assignment in real-time.

Hungarian Assignment is a combinational optimization algorithm that can be used to solve the assignment problem in polynomial time, with a computational complexity of $O(n^3)$, a significant reduction from the general assignment problem. For this reason, Hungarian Assignment has been utilized in many applications for teams of unmanned systems [62-64].

A detailed step-by-step process for the matrix manipulation required in Hungarian Assignment can be found in [65], and is summarized in the following example in Figure 19 for a balanced assignment, in which the number of available agents is equal to the number of available tasks. In this example, four unmanned aircraft agents A1-A4 must be optimally assigned to four flight line survey tasks T1-T4. Note that the aircraft are kinematically constrained by their minimum turning radius.

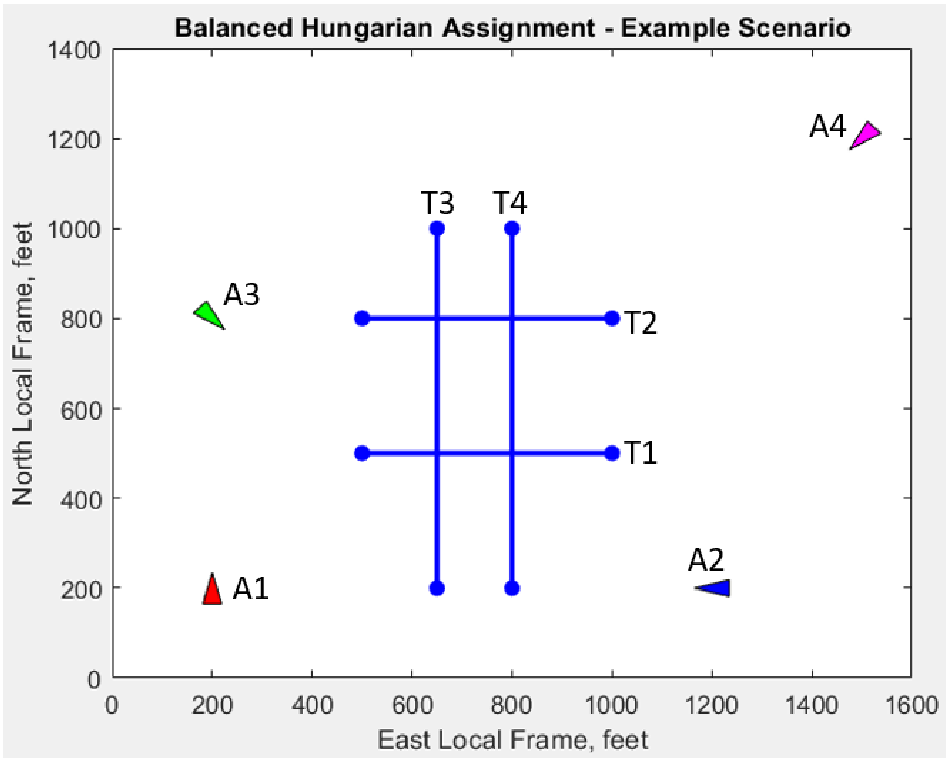
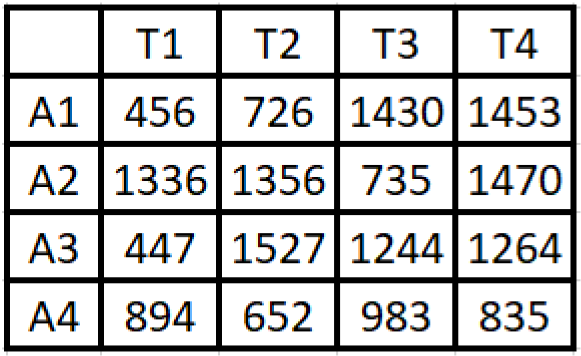


Figure 19: Balanced Hungarian Assignment - Example Scenario

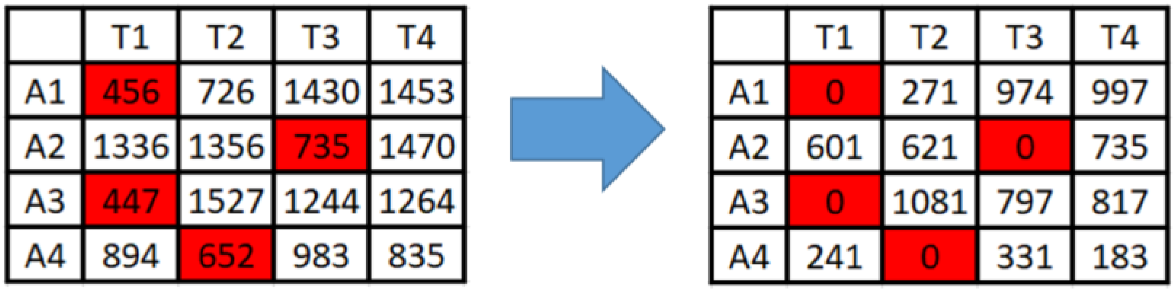
Using the minimum distance Dubins path length from each aircraft to the beginning of each flight line (See Section 2.2), a cost matrix of size 4x4 is formed in Figure 20, describing the cost (in units of feet) of each agent with each task. Note that our goal is to find the optimal matching between the agents and tasks resulting in the least possible sum cost of the assignments.



	T1	T2	T3	T4
A1	456	726	1430	1453
A2	1336	1356	735	1470
A3	447	1527	1244	1264
A4	894	652	983	835

Figure 20: **Balanced *Hungarian Assignment Example - Initial Cost Matrix***

In the first step of Hungarian Assignment, the minimum value element of each row is determined, and this value is subtracted from its entire respective row, as shown below in Figure 21. This will create at least N zeros in the matrix.

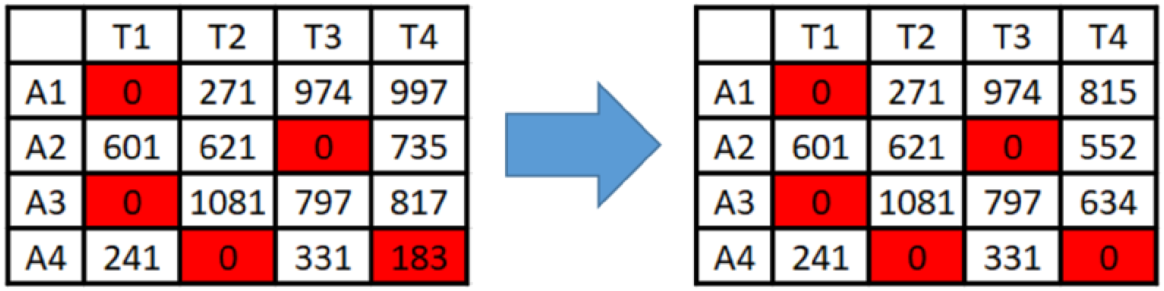


	T1	T2	T3	T4
A1	456	726	1430	1453
A2	1336	1356	735	1470
A3	447	1527	1244	1264
A4	894	652	983	835

	T1	T2	T3	T4
A1	0	271	974	997
A2	601	621	0	735
A3	0	1081	797	817
A4	241	0	331	183

Figure 21: Balanced Hungarian Assignment Example - Step 1

Step 2 of Hungarian Assignment involves taking the resulting matrix from Step 1, and determining the minimum valued element from each column, and subtracting this value from its entire respective column, as shown below in Figure 22.

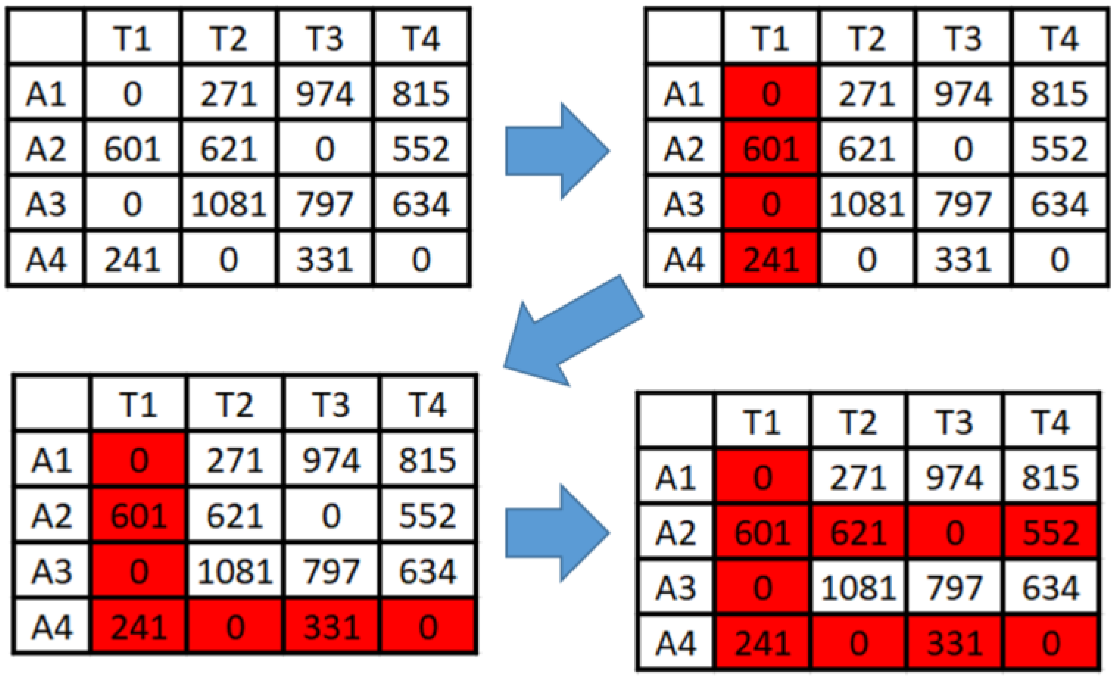


	T1	T2	T3	T4
A1	0	271	974	997
A2	601	621	0	735
A3	0	1081	797	817
A4	241	0	331	183

	T1	T2	T3	T4
A1	0	271	974	815
A2	601	621	0	552
A3	0	1081	797	634
A4	241	0	331	0

Figure 22: Balanced Hungarian Assignment Example - Step 2

Using the resulting matrix from previous steps, Step 3 involves determining if all of the zeros in the resulting matrix can be covered in no less than N rows or columns. If all of the zeros can only be covered by N rows/columns, then one should proceed to Step 5, as an optimal assignment is possible in the matrix. If not, proceed to Step 4 for some additional matrix manipulation. An example of this coverage is shown below in Figure 23, where it is shown that all of the zeros in this example matrix can be covered in N-1 rows/columns.



	T1	T2	T3	T4
A1	0	271	974	815
A2	601	621	0	552
A3	0	1081	797	634
A4	241	0	331	0

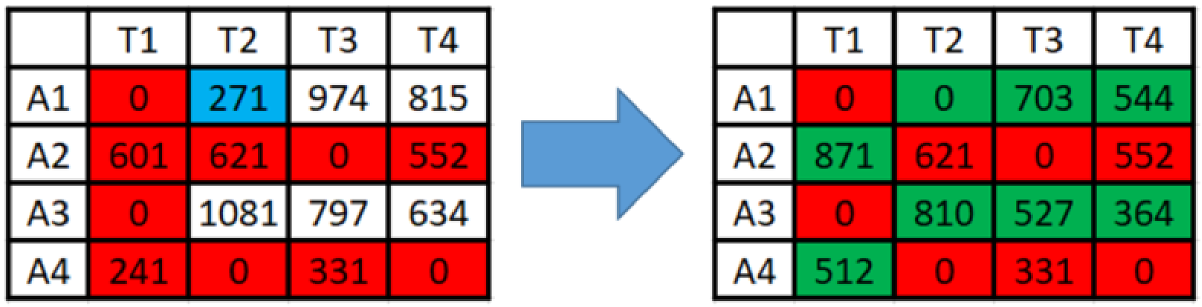
	T1	T2	T3	T4
A1	0	271	974	815
A2	601	621	0	552
A3	0	1081	797	634
A4	241	0	331	0

	T1	T2	T3	T4
A1	0	271	974	815
A2	601	621	0	552
A3	0	1081	797	634
A4	241	0	331	0

	T1	T2	T3	T4
A1	0	271	974	815
A2	601	621	0	552
A3	0	1081	797	634
A4	241	0	331	0

Figure 23: Balanced Hungarian Assignment Example - Step 3

Because the matrix failed the coverage criteria in Step 3, an optimal assignment cannot yet be found. In Step 4, the minimum valued element is determined from the uncovered region of the rows/columns used in the Step 3 coverage process. Next, this value is subtracted from all the uncovered cells, as well as added to any intersections in the rows/columns used for the coverage. An example of this is shown below in Figure 24.

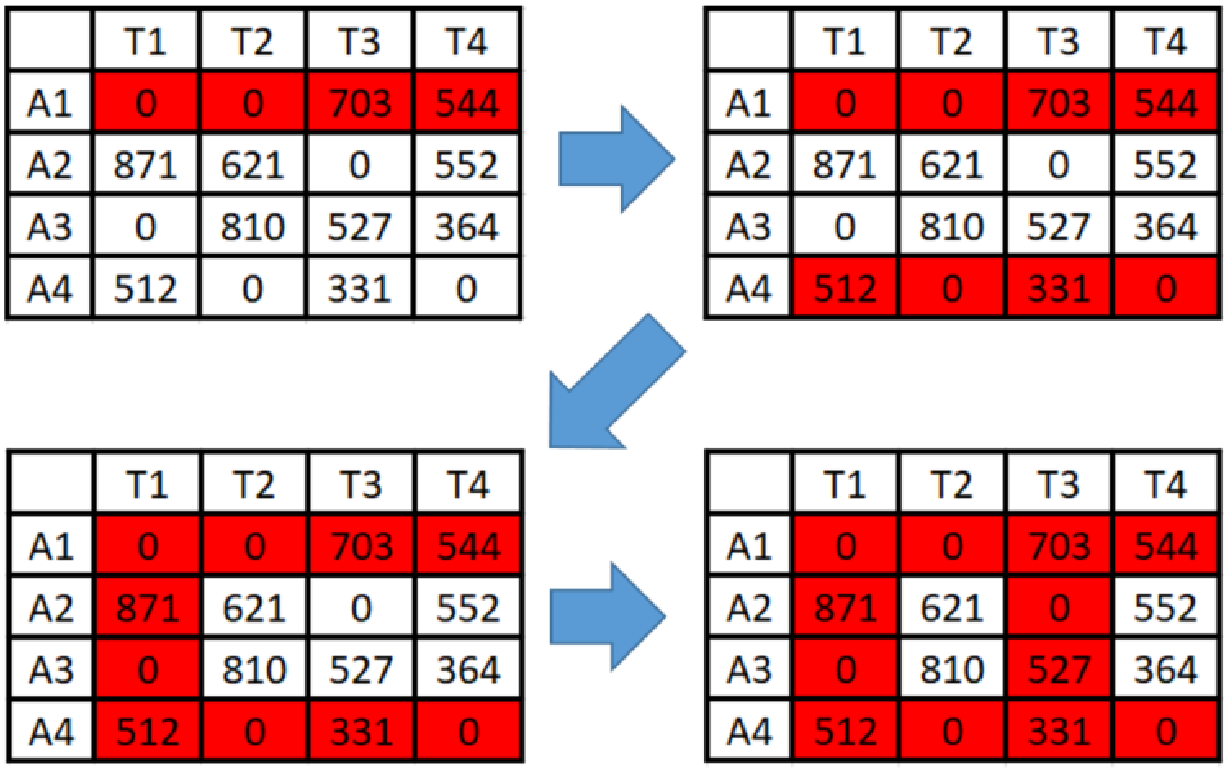


	T1	T2	T3	T4
A1	0	271	974	815
A2	601	621	0	552
A3	0	1081	797	634
A4	241	0	331	0

	T1	T2	T3	T4
A1	0	0	703	544
A2	871	621	0	552
A3	0	810	527	364
A4	512	0	331	0

Figure 24: Balanced Hungarian Assignment Example - Step 4

Following Step 4, the process returns to the Step 3 coverage assessment, and these matrix manipulation steps are iterated until an optimal assignment is determined that results in a zero sum assignment. The coverage assessment in Step 3 is shown below in Figure 25, where it can be seen that all zeros in the matrix can only be covered in N rows/columns, and therefore an optimal assignment is possible, and the Hungarian Assignment process can proceed to Step 5.



	T1	T2	T3	T4
A1	0	0	703	544
A2	871	621	0	552
A3	0	810	527	364
A4	512	0	331	0

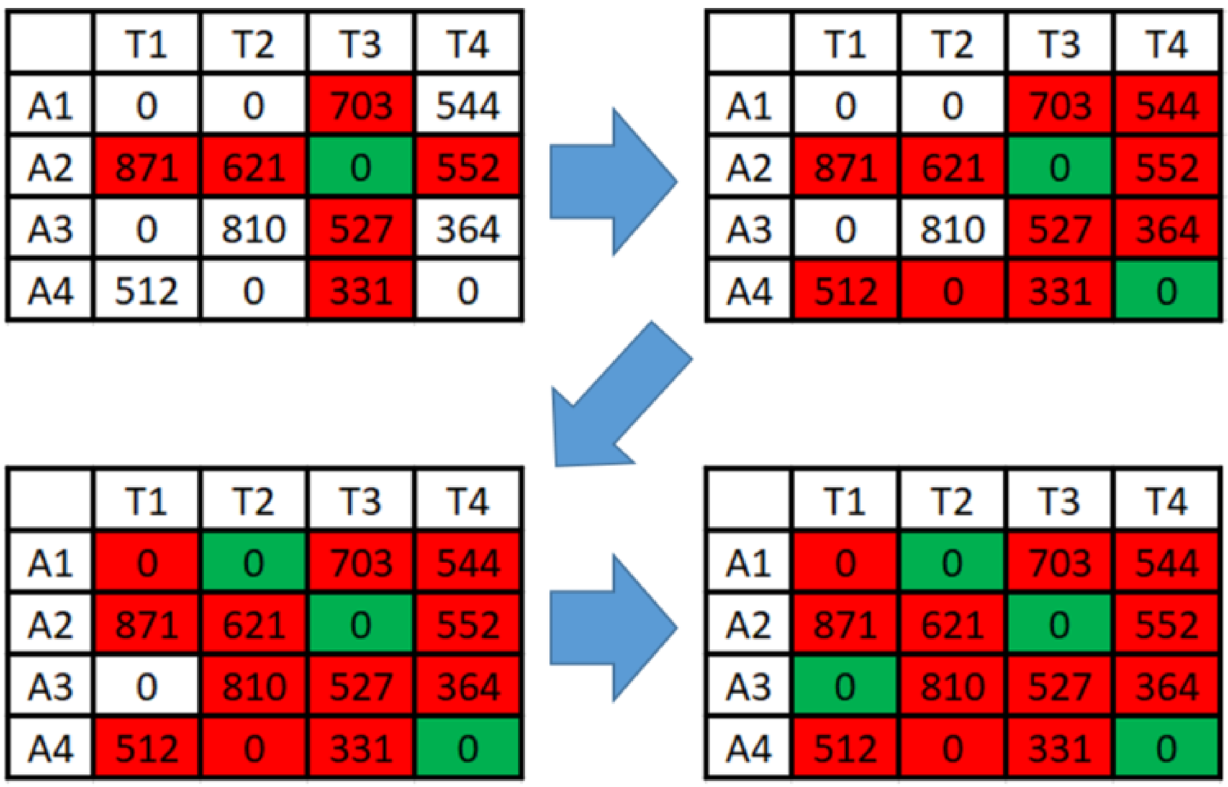
	T1	T2	T3	T4
A1	0	0	703	544
A2	871	621	0	552
A3	0	810	527	364
A4	512	0	331	0

	T1	T2	T3	T4
A1	0	0	703	544
A2	871	621	0	552
A3	0	810	527	364
A4	512	0	331	0

	T1	T2	T3	T4
A1	0	0	703	544
A2	871	621	0	552
A3	0	810	527	364
A4	512	0	331	0

Figure 25: Balanced Hungarian Assignment Example - Step 3 - Repeated

In Step 5 of the Hungarian Assignment process, an optimal assignment is determined from the matrix. This is achieved by determining which rows/columns contain only a single zero, and iteratively assigning these agents to the respective tasks. If for any reason there remains multiple zeros in all columns/rows, then there exists multiple solutions with a sum zero cost, all of which are optimal. An example for making the assignment using the matrix is shown below in Figure 26.



	T1	T2	T3	T4
A1	0	0	🟥 703	544
A2	🟥 871	🟥 621	🟩 0	🟥 552
A3	0	810	🟥 527	364
A4	512	0	🟥 331	0

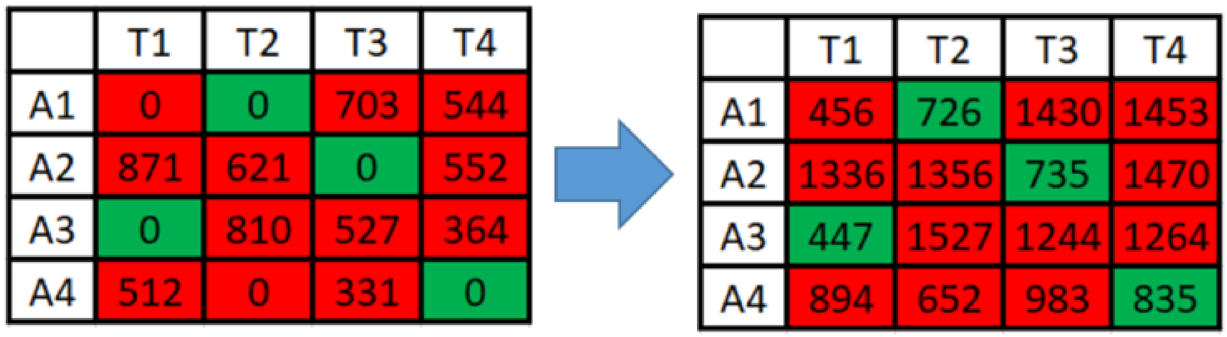
	T1	T2	T3	T4
A1	0	0	🟥 703	🟥 544
A2	🟥 871	🟥 621	🟩 0	🟥 552
A3	0	810	🟥 527	🟥 364
A4	🟥 512	🟥 0	🟥 331	🟩 0

	T1	T2	T3	T4
A1	🟥 0	🟩 0	🟥 703	🟥 544
A2	🟥 871	🟥 621	🟩 0	🟥 552
A3	0	🟥 810	🟥 527	🟥 364
A4	🟥 512	🟥 0	🟥 331	🟩 0

	T1	T2	T3	T4
A1	🟥 0	🟩 0	🟥 703	🟥 544
A2	🟥 871	🟥 621	🟩 0	🟥 552
A3	🟩 0	🟥 810	🟥 527	🟥 364
A4	🟥 512	🟥 0	🟥 331	🟩 0

Figure 26: Hungarian Assignment Example - Step 5

The resulting assignment from Step 5 can then be mapped back to the initial cost matrix to produce the optimal assignment, as shown below in Figure 27. Note that the total sum of the costs associated with the optimal assignment in this example is 2,743 feet.

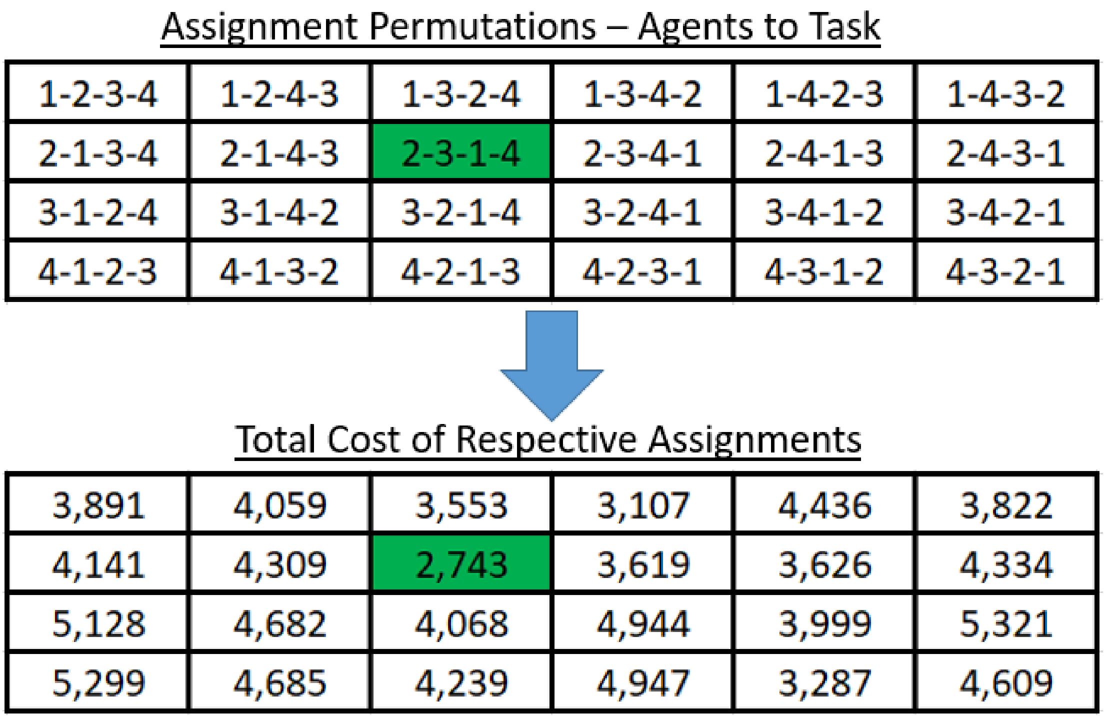


	T1	T2	T3	T4
A1	0	0	703	544
A2	871	621	0	552
A3	0	810	527	364
A4	512	0	331	0

	T1	T2	T3	T4
A1	456	726	1430	1453
A2	1336	1356	735	1470
A3	447	1527	1244	1264
A4	894	652	983	835

Figure 27: Hungarian Assignment Example - Mapping to Initial Cost Function

By utilizing brute-force methods, it can be determined that the solution resulting from Hungarian Assignment is in fact the optimal assignment of the agents to the tasks. Figure 28 shows the total cost of the N! possible assignments between the four agents and the four tasks in this example, of which the minimum total cost is the one determined from the Hungarian Assignment process. Note that the assignment permutations refers to the order in which the tasks T1-T4 are assigned to the agents A1-A4.



Assignment Permutations – Agents to Task

1-2-3-4	1-2-4-3	1-3-2-4	1-3-4-2	1-4-2-3	1-4-3-2
2-1-3-4	2-1-4-3	2-3-1-4	2-3-4-1	2-4-1-3	2-4-3-1
3-1-2-4	3-1-4-2	3-2-1-4	3-2-4-1	3-4-1-2	3-4-2-1
4-1-2-3	4-1-3-2	4-2-1-3	4-2-3-1	4-3-1-2	4-3-2-1

Total Cost of Respective Assignments

3,891	4,059	3,553	3,107	4,436	3,822
4,141	4,309	2,743	3,619	3,626	4,334
5,128	4,682	4,068	4,944	3,999	5,321
5,299	4,685	4,239	4,947	3,287	4,609

Figure 28: Optimal Assignment - Brute Force Validation

The Hungarian Assignment result produced in this example is shown below in Figure 29.

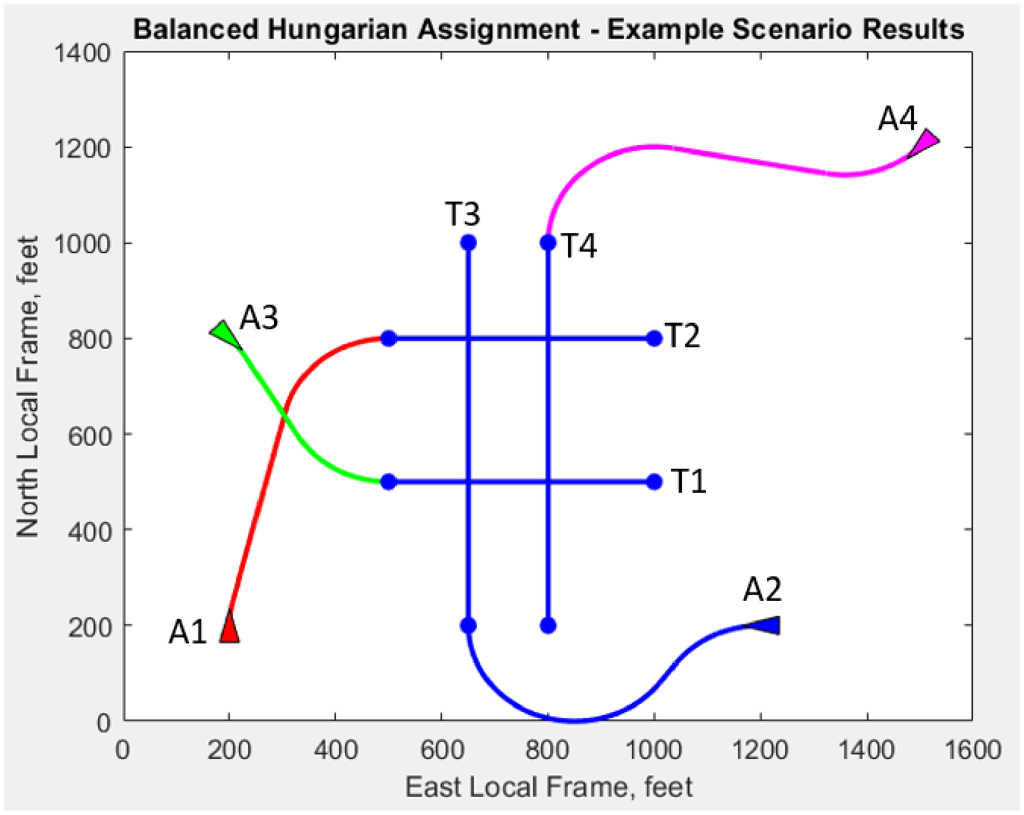


Figure 29: Balanced Hungarian Assignment - Example Scenario Result

2.5.2 Unbalanced Assignment

Hungarian Assignment inherently works for balanced assignment problems, where there is an equal number of agents and tasks, for which the assignment cost matrix is square. However, this is not usually the case during mission operations. However, the Hungarian Assignment process can be modified in order to incorporate unbalanced assignment problems. This is done by introducing synthetic agents or tasks in order to augment the cost matrix into a square matrix. The square matrix can then be subjected to the Hungarian Assignment steps detailed in the previous section, and the resulting assignment can be mapped back to the initial unbalanced matrix to produce the optimal assignment. Note that any synthetic agent/tasks assignments in the resulting square matrix are simply disregarded in the final solution. The unbalanced Hungarian Assignment process is detailed in this section for the example scenario shown in Figure 30, where an optimal assignment of four agents to three tasks must be made.

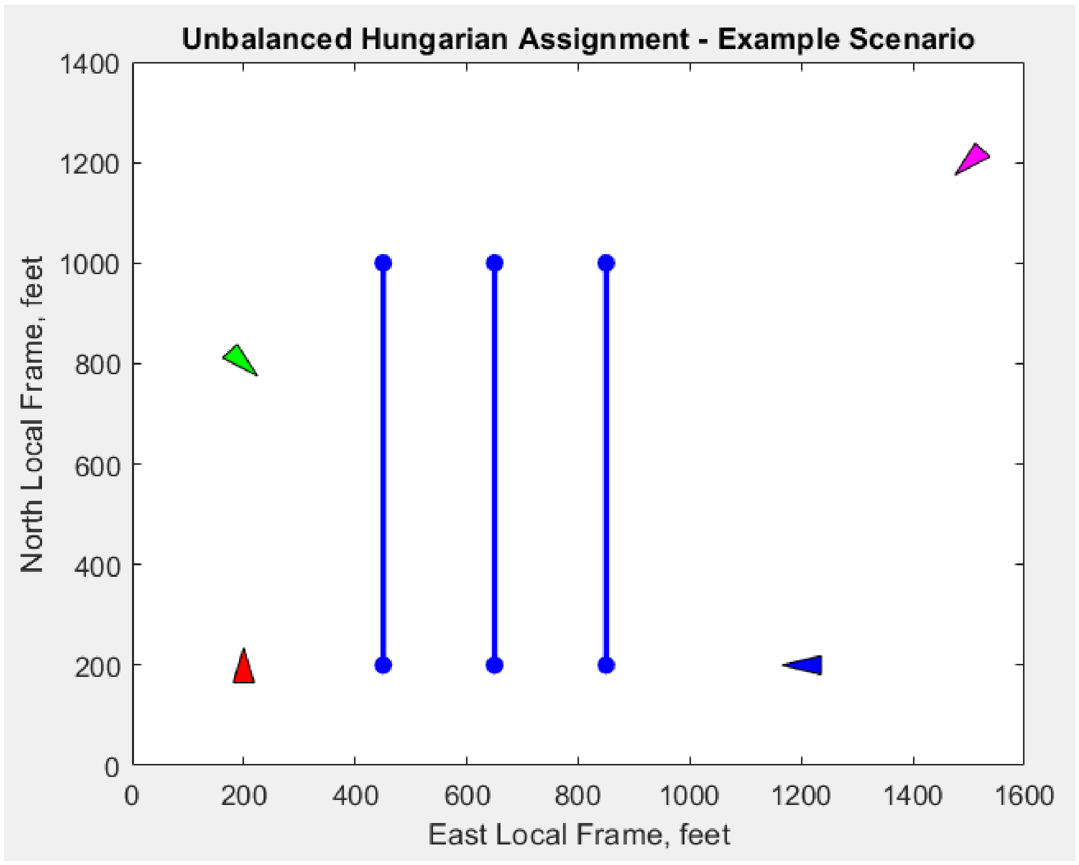
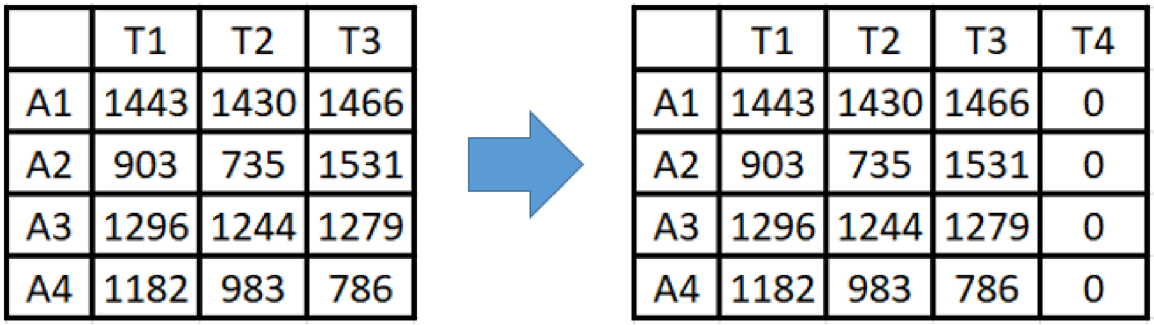


Figure 30: Unbalanced Hungarian Assignment - Example Scenario

As before, a cost matrix is formed for each agent and task using the respective minimum length Dubins Path from the aircraft initial conditions to the beginning of each flight line. This cost matrix (in units of feet) is then augmented into a square matrix by introducing a synthetic fourth agent, shown in Figure 31.

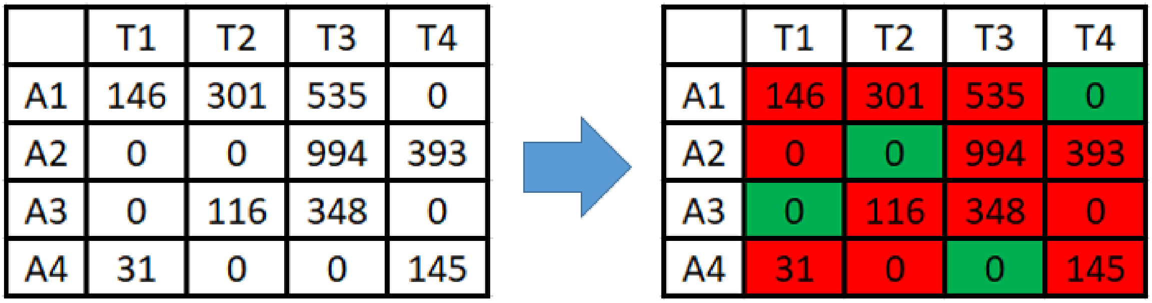


	T1	T2	T3
A1	1443	1430	1466
A2	903	735	1531
A3	1296	1244	1279
A4	1182	983	786

	T1	T2	T3	T4
A1	1443	1430	1466	0
A2	903	735	1531	0
A3	1296	1244	1279	0
A4	1182	983	786	0

Figure 31: Unbalanced Hungarian Assignment - Augmenting Cost Matrix

The resultant square matrix is then subjected to the Hungarian Assignment steps 1-5 detailed in the previous section. The resulting matrix and assignment is shown in Figure 32.

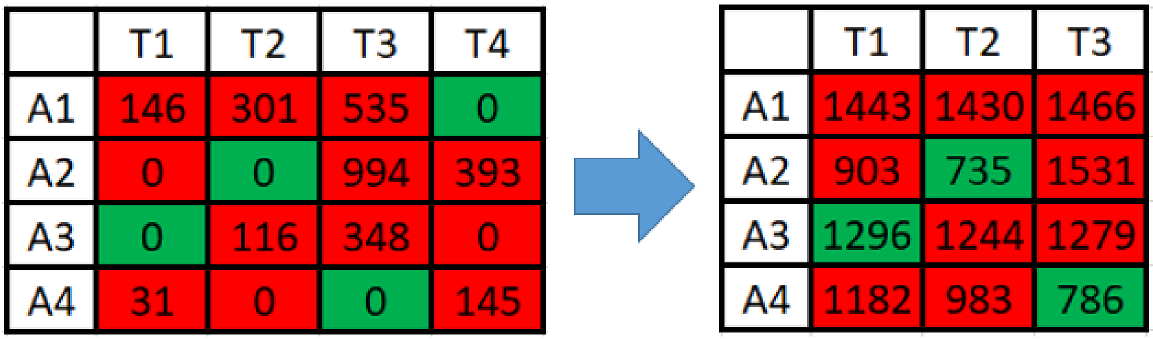


	T1	T2	T3	T4
A1	146	301	535	0
A2	0	0	994	393
A3	0	116	348	0
A4	31	0	0	145

	T1	T2	T3	T4
A1	🟥 146	🟥 301	🟥 535	🟩 0
A2	🟥 0	🟩 0	🟥 994	🟥 393
A3	🟩 0	🟥 116	🟥 348	🟥 0
A4	🟥 31	🟥 0	🟩 0	🟥 145

Figure 32: Unbalanced Hungarian Assignment - Optimal Solution Selection

The resulting assignment can then be mapped back to the initial unbalanced cost matrix, where it can be noted that the total sum of the costs associated with the optimal assignment in this example is 2,817 feet.

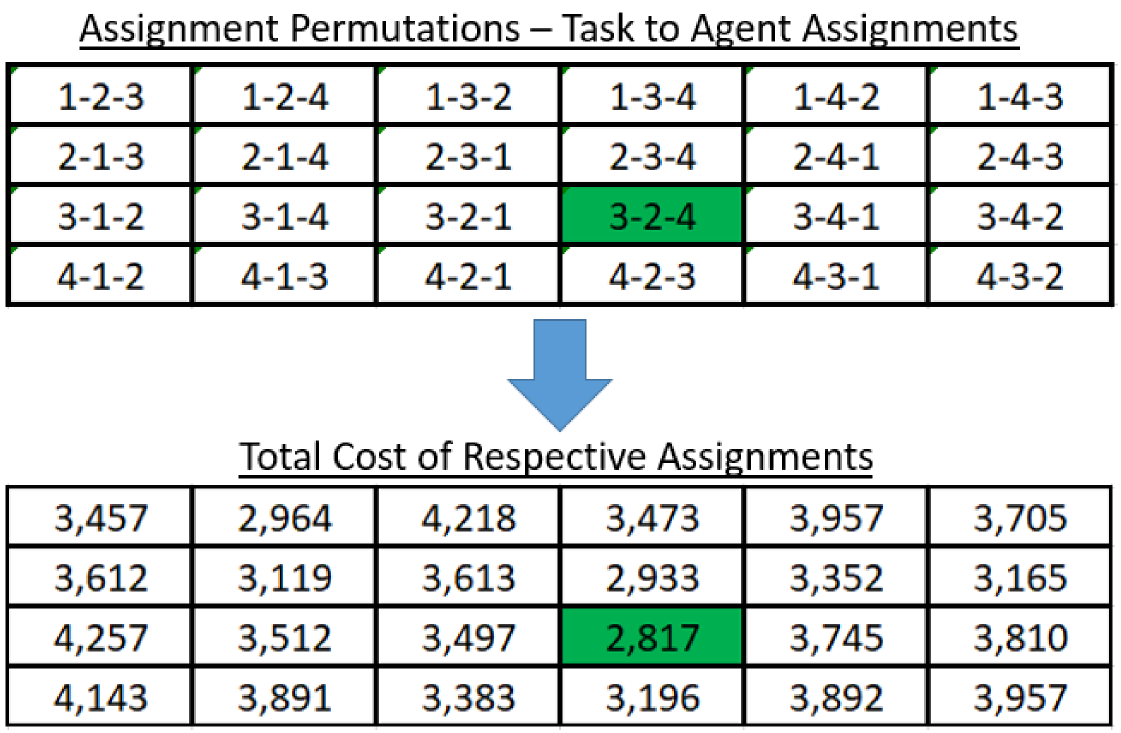


	T1	T2	T3	T4
A1	🟥 146	🟥 301	🟥 535	🟩 0
A2	🟥 0	🟩 0	🟥 994	🟥 393
A3	🟩 0	🟥 116	🟥 348	🟥 0
A4	🟥 31	🟥 0	🟩 0	🟥 145

	T1	T2	T3
A1	🟥 1443	🟥 1430	🟥 1466
A2	🟥 903	🟩 735	🟥 1531
A3	🟩 1296	🟥 1244	🟥 1279
A4	🟥 1182	🟥 983	🟩 786

Figure 33: Unbalanced Hungarian Assignment - Mapping Back to Initial Cost Matrix

Using Brute-Force methods, the total cost of every possible assignment between the four agents and the three flight lines can be calculated, and from these candidates in Figure 34 it can be proven that the assignment created using the unbalanced Hungarian Assignment is the optimal solution. Note that the assignment permutations refers to the order of the agents A1-A4 that is assigned to the flight lines T1-T3.



Assignment Permutations – Task to Agent Assignments

1-2-3	1-2-4	1-3-2	1-3-4	1-4-2	1-4-3
2-1-3	2-1-4	2-3-1	2-3-4	2-4-1	2-4-3
3-1-2	3-1-4	3-2-1	3-2-4	3-4-1	3-4-2
4-1-2	4-1-3	4-2-1	4-2-3	4-3-1	4-3-2

Total Cost of Respective Assignments

3,457	2,964	4,218	3,473	3,957	3,705
3,612	3,119	3,613	2,933	3,352	3,165
4,257	3,512	3,497	2,817	3,745	3,810
4,143	3,891	3,383	3,196	3,892	3,957

Figure 34: Unbalanced Hungarian Assignment - Brute Force Validation

The resulting optimal solution for this unbalanced assignment example is shown in Figure 35, where Agent 1 is simply not assigned to one of the three flight lines.

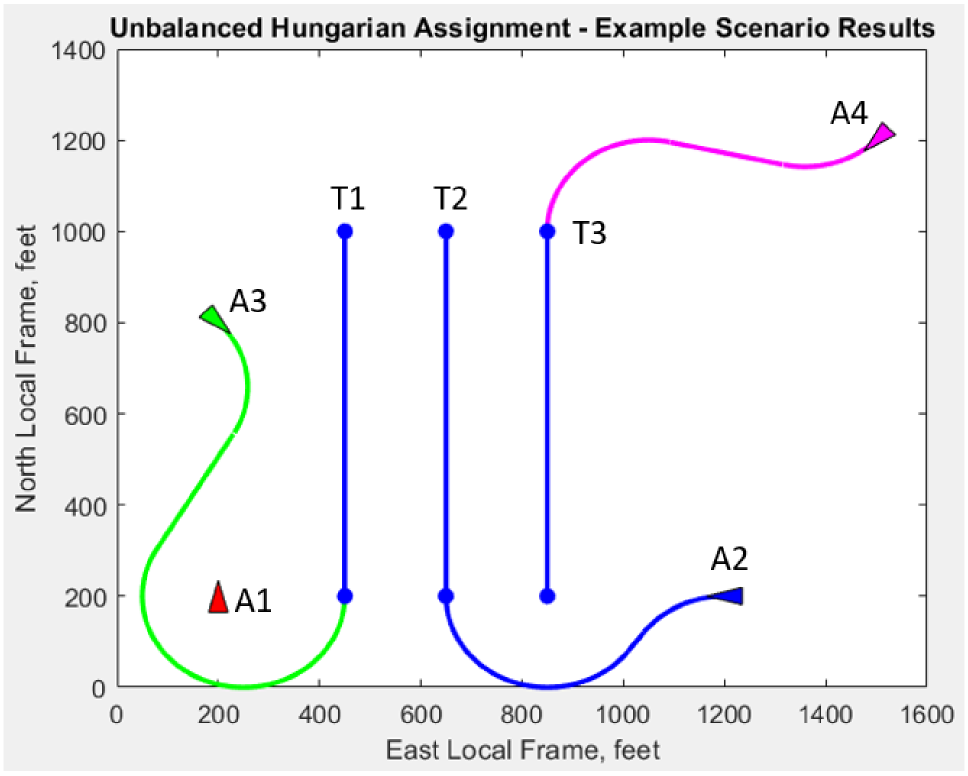


Figure 35: Unbalanced Hungarian Assignment - Example Scenario Result

Note that for scenarios in which there are a larger number of flights lines than agents, synthetic agents are introduced in order to develop an optimal assignment using the Hungarian Assignment methods. However this has the potential to create very large matrices, requiring increasing computational expense to determine optimal solutions. Various methods can be utilized in certain scenarios order to reduce the size of the assignment matrix, saving computational runtime for the assignment. The simplest of these methods is for scenarios in which there is a single agent/task that must be matched to multiple agents/tasks, in which case a cost vector can be created for the assignment and the minimum valued element can be used for the assignment, avoiding the Hungarian Assignment matrix processes altogether. More complex methods for matrix reduction involves identifying overlapping candidate assignments for each agent/task and eliminating other agents/tasks that would inevitably be matched with synthetic agents/tasks in the unbalanced Hungarian Assignment process.

2.5.3 Iterative Optimal Assignment

Due to the assignment between agents and flight lines generally being unbalanced during flight operations, the Hungarian Assignment in this work is applied iteratively, with only the minimum cost selection of the optimal assignment being used in each iteration. That is, after an optimal assignment is determined between the number of available agents and flight lines, only the minimal cost assignment is actually used to assign a flight line to an agent. If the assignment is feasible based on the remaining range for the agent, that agent's position, heading, initial costs, and remaining range are updated accordingly to that of the completed assigned flight line, and this updated information is used to perform subsequent Hungarian Assignment iterations. If the assignment is not feasible based on fuel constraints, the agent is removed from future assignment iterations. By using Hungarian Assignment iteratively in this way, an approximate to the global optimal assignment is selecting local optimal solutions.

The motivation for this single-selection iteration is demonstrated in the following example shown in Figure 36, where four agents must survey five flight lines in an optimal manner, with minimum turning radii of 200 feet.

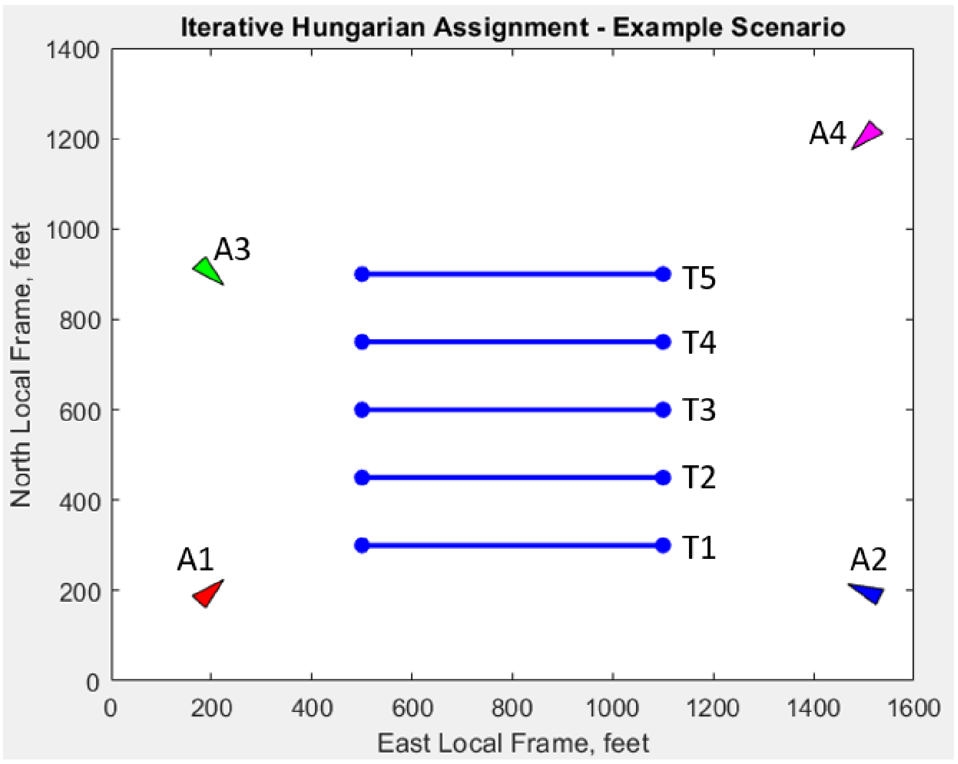
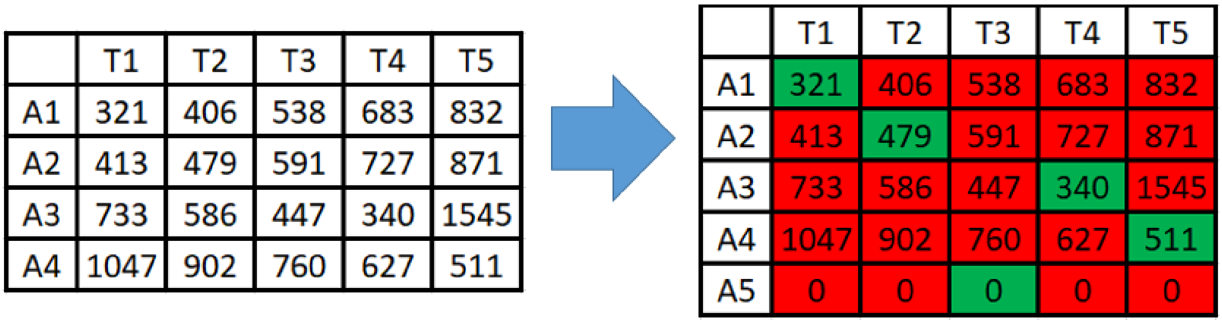


Figure 36: Iterative Hungarian Assignment - Example Scenario

The initial assignment cost matrix for this example scenario is shown in Figure 37, as well as the optimal solution following the unbalanced Hungarian Assignment processes outlined in Section 2.5.2, which introduces a synthetic agent A5. This optimal solution is shown in Figure 38.



	T1	T2	T3	T4	T5
A1	321	406	538	683	832
A2	413	479	591	727	871
A3	733	586	447	340	1545
A4	1047	902	760	627	511

	T1	T2	T3	T4	T5
A1	🟩 321	🟥 406	🟥 538	🟥 683	🟥 832
A2	🟥 413	🟩 479	🟥 591	🟥 727	🟥 871
A3	🟥 733	🟥 586	🟥 447	🟩 340	🟥 1545
A4	🟥 1047	🟥 902	🟥 760	🟥 627	🟩 511
A5	🟥 0	🟥 0	🟩 0	🟥 0	🟥 0

Figure 37: Initial Cost Matrix and Optimal Solution

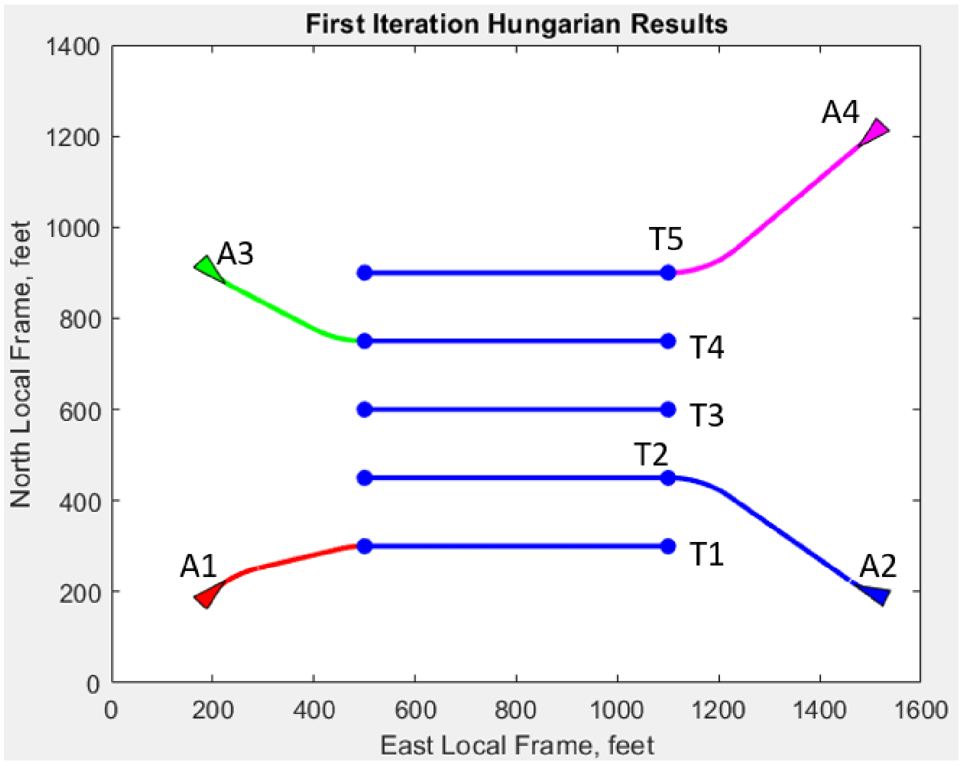


Figure 38: First Iteration Hungarian Results

However, only the minimum cost selection is made and applied to the agents/tasks. In this scenario, Agent 1 (A1) is assigned to Flight Line 1 (T1), and the next iteration of Hungarian Assignment is performed using updated position, heading, initial cost, and remaining range for Agent 1. This instance that the next iteration is applied to is shown in Figure 39.

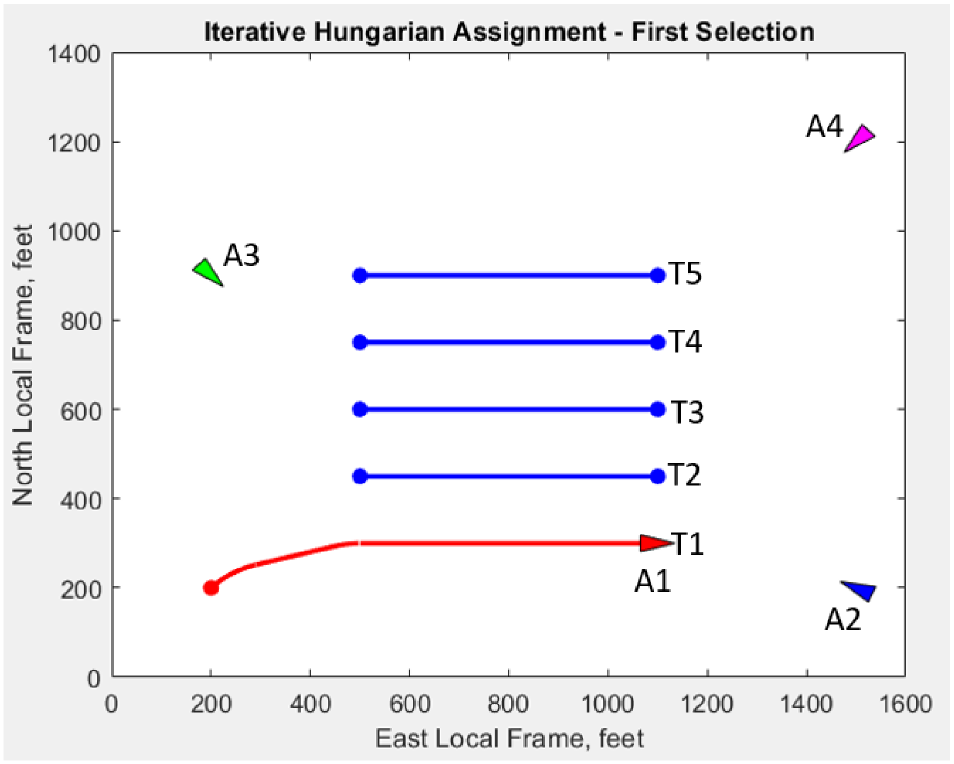
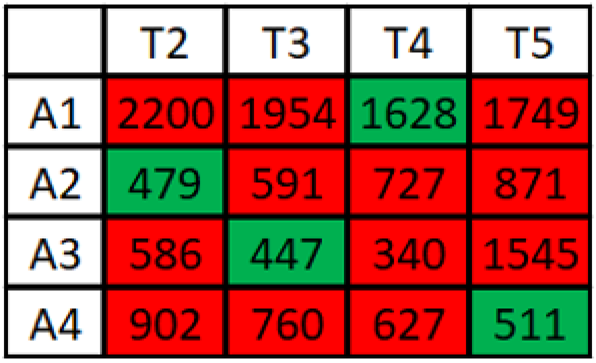


Figure 39: Iterative Hungarian Assignment - First Selection

The assignment cost matrix and optimal solution for the remaining flight lines is shown in Figure 40. Note that the cost for Agent 1 has been updated to reflect the distance already traveled from the initial starting point through the survey of Flight Line 1. This cost is added to the Dubins Path length to each of the remaining flight lines T2-T5 in order to construct the A1 row in the cost matrix. This optimal assignment for the remaining lines is shown in Figure 41. Following four more iterations of Hungarian Assignment where only the minimum assignment is applied and the respective agent costs are updated, the resulting final assignment is determined.



	T2	T3	T4	T5
A1	2200	1954	1628	1749
A2	479	591	727	871
A3	586	447	340	1545
A4	902	760	627	511

Figure 40: Hungarian Assignment - 2nd Iteration Result

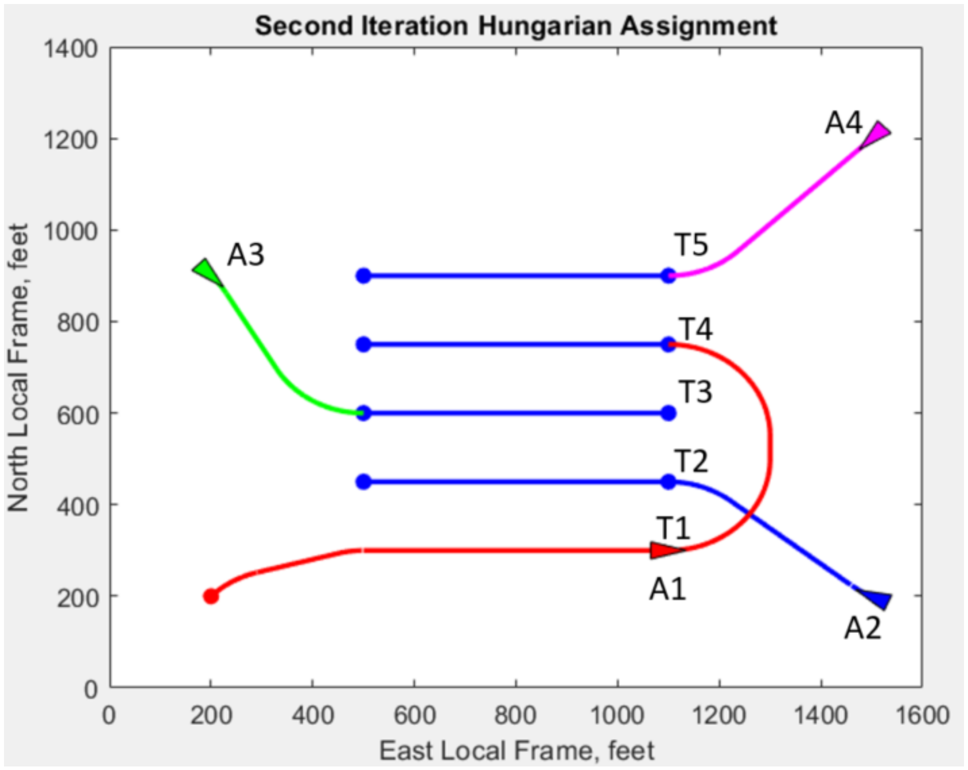


Figure 41: Hungarian Assignment - Second Iteration Result

However, if all four flight lines from the first iteration assignment had been used to assign flight lines to the agents, the resulting flight line allocation and survey operation would differ. Figure 42 shows the scenario in which all four assignments were applied following the first iteration of Hungarian Assignment, in which case a second iteration of Hungarian Assignment would only determine which of the agents should survey the remaining flight line T3. The cost matrix for this new scenario is shown in Figure 43, where it can be seen that either Agent 1 or Agent 4 are valid optimal assignments due to the symmetric Dubins Path lengths.

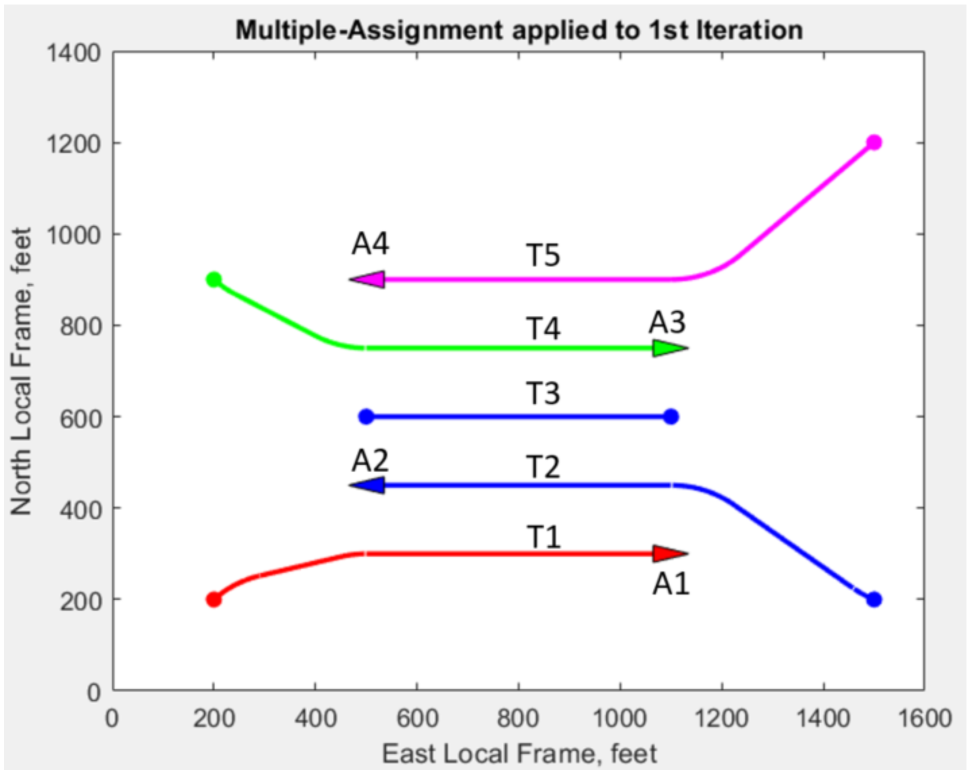
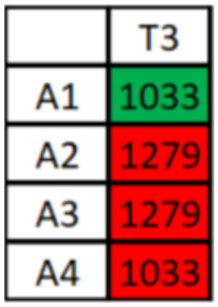


Figure 42: Multiple-Assignment applied to 1st Iteration



	T3
A1	1033
A2	1279
A3	1279
A4	1033

Figure 43: Multiple-Assignment - Second Iteration Cost Matrix

The resulting difference between single and multiple assignments per iteration of Hungarian Assignment can be seen in Figure 44, where ultimately the difference lies in Agent 3 being assigned flight line T3 instead of the initially closer flight line T4, as the difference is compensated by Agent 1’s Dubins Path length to the respective remaining flight line (T4 vs T3). This example highlights the motivation in this work to use iterative single-assignment Hungarian processes in order to better approximate the global optimal assignment between agents and mission flight lines.

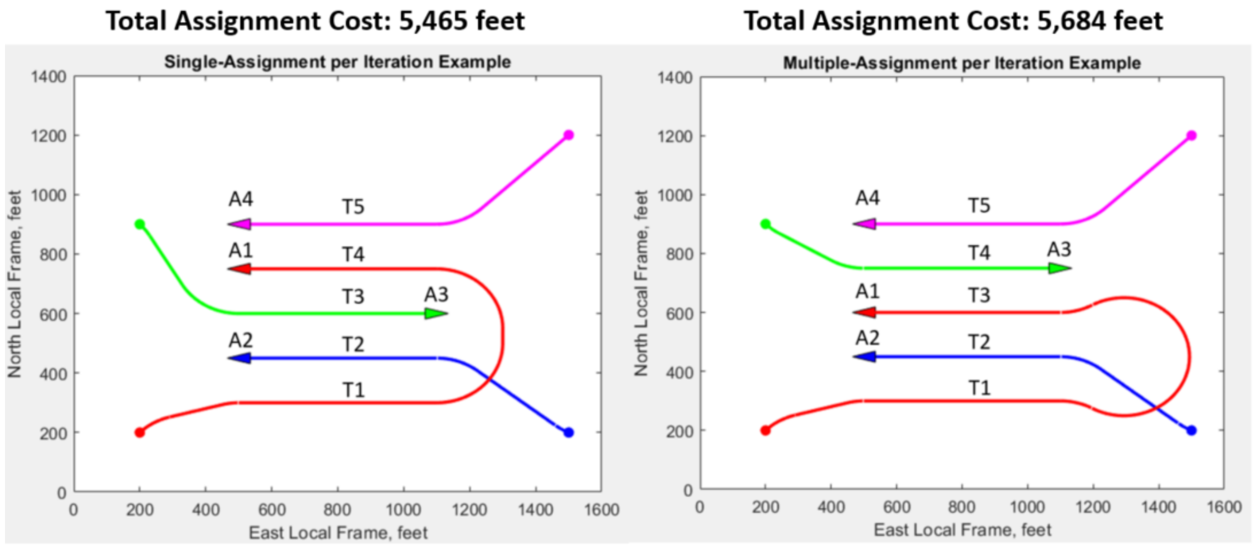


Figure 44: Single vs Multiple Assignment per Iteration

2.5.4 Clustering Algorithms

As mentioned in Section 2.1, many of these polar research missions consist of groups of closely spaced parallel flight lines. While these groups of lines are beneficial in terms of capturing more coherent and higher resolution imaging, increasing the number of total flight lines dramatically increases the computational expense of the Hungarian Assignment used for allocating the lines amongst the agents. For this reason, space partitioning is used on the entire flight lines set in order to subdivide them into "clusters". This is accomplished by determining the headings of the individual flight lines and grouping closely-spaced parallel lines into clusters, defined by a centroid flight line and the number of lines in the cluster. Once the flight lines have been grouped into these clusters, Hungarian Assignment is used to allocate entire clusters amongst the agents, using the iterative assignment method described in Section 2.5.3. Shown below in Equation 2.85, the cost function used for the clusters in the Hungarian Assignment matrices can be defined as the summations of lengths of the Dubins Path onto the beginning of the cluster's centroid flight line (D_1), the number of flight lines in the cluster (N_{FL}) multiplied by the total length of the flight lines (L_{FL}), and the number of times the aircraft must turn around between sweeps in the cluster $(N_{FL} - 1)$ multiplied by the length of the Dubins Path for this maneuver (D_T). By doing this, the true "cost" of surveying the entire cluster is incorporated into the assignment problem. Note that the length of the Dubins Paths in the clusters are a direct function of the respective agent's minimum turning radius. Using the automated flight line generation example in Section 2.1, Figure 45 shows an example of how using space partitioning can reduce 15 flight lines into 3 clusters, significantly reducing the complexity of the assignment among the agents.

$$J_{Cluster} = D_1 + N_{FL}L_{FL} + (N_{FL} - 1)D_T \qquad \textit{Eq. 2.85}$$

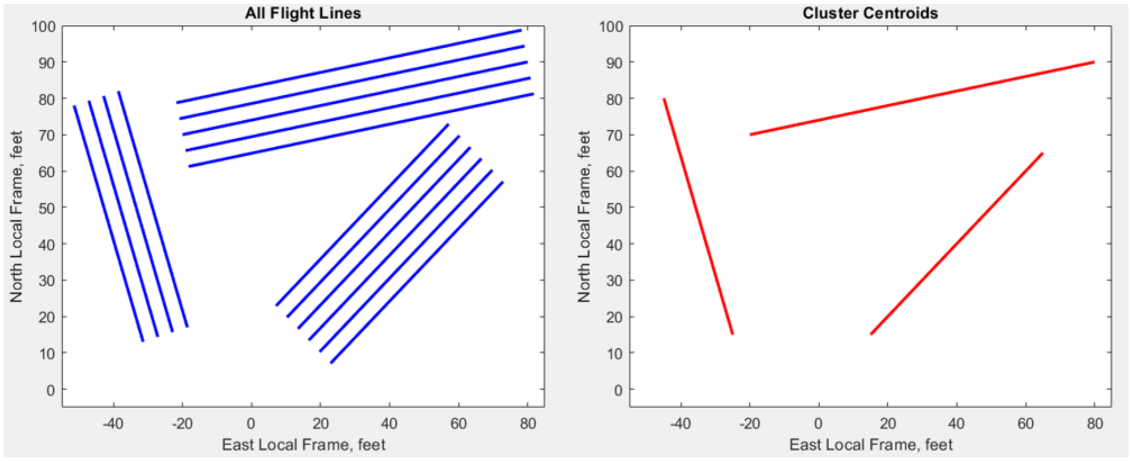


Figure 45: Clustering Example

Note that while the space partitioning described in this section is done to significantly reduce the computational complexity of the real-time multi-agent assignment, it also has several benefits for flight operations due to the effective separation of the agents from unique cluster assignments. For example, preventing the allocation of flight lines within the same cluster to multiple agents reduces the chance of accidental collision between aircraft while surveying the lines. This would also prevent unnecessary avoidance maneuvers if collision avoidance methods are utilized, such as the Morphing Potential Fields described in Section 3.5. While safely avoiding collision, those methods would result in loss of coherent data due to the off-track positions and high bank angles required for the avoidance maneuver. Additionally, the onboard radar systems would conceivably receive more noise from nearby agents if radiating together in close proximity, so assigning entire clusters to the agents would create more separation between the systems and improve the data collection integrity. Finally, an entire cluster should ideally be surveyed by a single agent in order to prevent unknown factors from causing discrepancies in post-processing synthetic aperture methods.

2.5.5 Collaborative Surveying

The processes for the multi-agent collaborative surveying autonomy developed in this work can be summarized the autonomy roadmap shown in Figure 46.

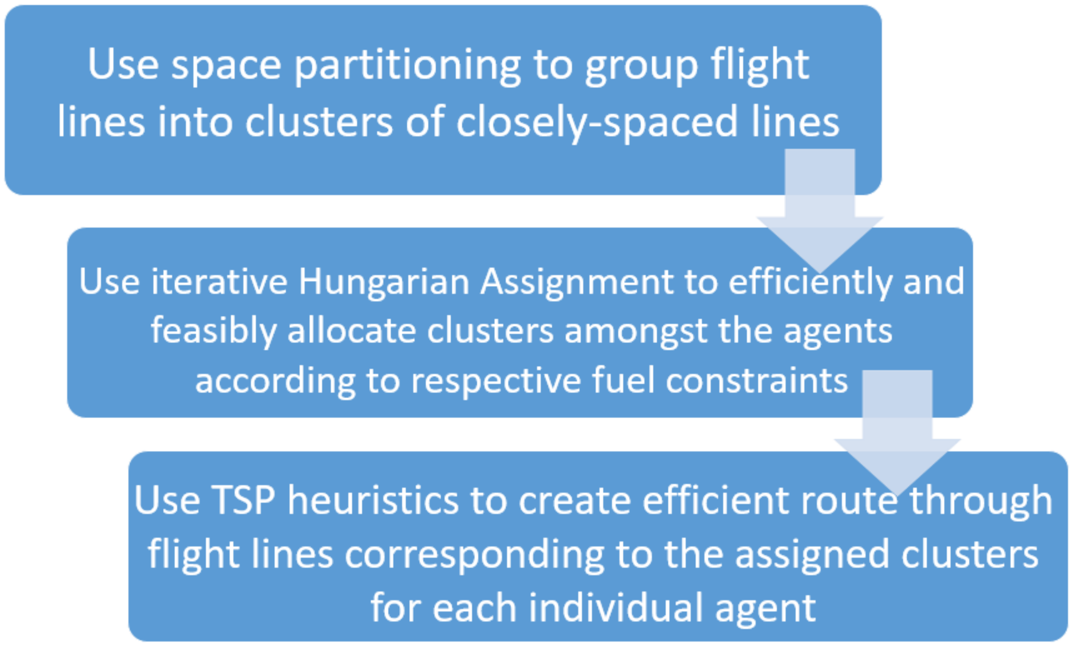

Use space partitioning to group flight lines into clusters of closely-spaced lines
Use iterative Hungarian Assignment to efficiently and feasibly allocate clusters amongst the agents according to respective fuel constraints
Use TSP heuristics to create efficient route through flight lines corresponding to the assigned clusters for each individual agent

Figure 46: Collaborative Surveying Autonomy Roadmap

2.6 Multi-Agent Scheduling

While the developed autonomy for multi-agent operations outlined in Section 2.5 has the potential to vastly increase the capabilities of polar research operations, considerations must be made when incorporating numerous, sensitive radar systems into a confined airspace. For snow accumulation radars, which typically radiate from 2-18 GHz, a highly directive antenna can be integrated onto the UAS platforms and can therefore operate in the same general area without experiencing cross-platform interference [66]. However, glacial sounding applications require much lower frequencies (typically below 55 MHz), which necessitate the use of a very long, low-directivity antenna. Given the limited available space along the wing planforms of small UAS, these antennas have a very wide radiation pattern and multi-agent operations could possibly result in cross-platform interference with a sufficiently large signal-to-noise ratio that would mask the desired basal returns. If this interference is above an acceptable threshold, it is desirable for only one UAS platform to be radiating at a single instance for these systems in order to ensure coherent data collection during operations.

However, the overall efficiency of the surveying operation can be improved through intelligent flight line assignment and deployment scheduling of systems into the survey area [67-69]. This can be achieved using a similar assignment framework as outlined in Section 2.5, but only physically deploying one of the systems into the field at a time. The Hungarian Assignment is performed using all available agents and remaining mission flight lines, however only the deployed agent is assigned its respective flight lines to survey, while other agent await deployment. The agent performing the over-the-horizon operation will continue to track flight lines until the fuel constrained assignment commands the deployed agent to return to its respective home loiter position. When the deployed aircraft finishes its last survey line, it can signal the onboard radar system to cease transmitting, and the subsequent agent is immediately deployed to survey the remaining flight lines. The returned agent is then removed from the decentralized Hungarian Assignment process for flight line assignment and agent deployment scheduling.

By incorporating similar assignment and routing frameworks as collaborative surveying, the operating system is effectively making flight line assignments based on the yet-to-be-deployed agents being available for subsequent surveying of the remaining mission flight lines. For homogenous platforms, the ordering of the system deployments is arbitrary, and in this work follows sequential ordering of the unique agent-identifier numberings. The timing of subsequent agent deployment is made as soon as the deployed agent is transmitting a “return to home” status. Agents awaiting deployment should ideally be grounded in order to preserve fuel, but fixed-wing platforms can be placed into an autonomous loiter immediately prior to the return of deployed agents. However, this deployment scheduling autonomy is best suited for next-generation polar research VTOL platforms, which can even be actively charging while awaiting automated deployment. Following return from surveying operations, agents can even transfer collected radar data to ground station operators, clear their available storage space, and recharge for subsequent deployments. Depending on the number of available systems and their respective endurance and charging rates, these survey operations could be performed continuously while environmental factors allow.

2.7 Multi-Agent Heterogeneous Operations

The decentralized Hungarian Assignment processes utilized in this work can easily be modified for survey operations using heterogeneous unmanned systems. This is done be modifying the cost functions used for flight line allocation from the required distance to the required time for the agent to survey the flight line. The two vehicle-specific parameters driving this survey time are the vehicle ground speed and the minimum turning radius. As the Dubins Path length from the agent to a flight line is a direct function of the aircraft’s minimum turning radius, smaller, more mobile aircraft are inherently more suitable for tight grids. On the other hand, larger, faster aircraft are more preferable for long stretches of survey lines. By modifying the cost function, the Hungarian Assignment processes can be used to intelligently allocate the flight lines according to the agents based on these considerations. The equation used in heterogeneous multi-agent operations is given in Equation 2.86, where the cost function J is dependent on the Dubins

Path Length D_L and the aircraft ground speed V_G. Note that this heterogeneous modification can be used for both collaborative survey or for scheduled deployment missions. If multi-rotor systems are integrated in future polar research missions, note that they no turning radius constraints, and therefore their Dubins Path lengths would simply consist of the linear distance to the flight line start points.

$$J = \frac{D_L}{V_G}, \quad where\ D_L = f(R_{Min}) \qquad \text{Eq. 2.86}$$

2.8 Robustness Towards System Failures and Communication Constraints

The introduction of multi-agent systems inherently increases the operational capabilities of polar research missions at the cost of increasing the required autonomy complexity. In order to properly take advantage of these benefits, the developed autonomy should incorporate robustness towards foreseeable issues during flight operations. In this work, contingencies for onboard system failures have been incorporated into the onboard autonomy.

Following the detection of onboard system failures, the malfunctioning systems have been designed to notify other agents, and subsequently cease flight line surveying and attempt to return to their home loiter. If the agent was tracking a flight line and did not complete it, the flight line is designated as incomplete by the other agents, and is added back to the list of remaining flight lines to survey. The Hungarian Assignment process is updated in order to allocate the remaining flight lines among the remaining agents, now that the malfunctioning agent is no longer available to aid in the flight line surveying. Note that the decentralized nature of the autonomy in this work is critical to this operational robustness.

Chapter 3: Autonomy Integration

In this chapter, the methodology for integrating the developed autonomy into an autopilot system for an unmanned aircraft will be discussed.

3.1 6-DoF Aircraft Equations of Motion

In this work, nonlinear aircraft six-degree-of-freedom equations of motion are utilized for simulation purposes and for controller design. The stability and control derivatives used in these equations of motion are obtained from dynamic models developed in an aircraft modeling software known as Advanced Aircraft Analysis (AAA) [70] in which the aircraft is perturbed about a steady state trim, level-flight condition. The software uses a combination of high fidelity physics-based and semi-empirical methods based on vast databases of historical aircraft relations, trends, and design philosophy outlined in [71].

The aircraft state vector is shown in Equation 3.1, and consists of the total airspeed, angle of attack, sideslip angle, inertial Euler angles, and the angular rates in the body frame, respectively. The control vector is shown in Equation 3.2, and includes throttle, elevator, aileron, and rudder deflections, respectively.

$$x = \begin{bmatrix} V_T \\ \alpha \\ \beta \\ \varphi \\ \theta \\ \psi \\ P \\ Q \\ R \end{bmatrix} \qquad \textit{Eq. 3.1}$$

$$u = \begin{bmatrix} \delta_T \\ \delta_E \\ \delta_A \\ \delta_R \end{bmatrix} \qquad \textit{Eq. 3.2}$$

In this work, servo dynamics are modeled using first order delays between the previous control states and commanded values from the aircraft control block in order to more realistically model the delay in actuator deflection experienced in real flight. These delays are modeled below in Equation 3.3, and are used for propagating control states using integration schemes such as RK-4.

$$\begin{bmatrix} \dot{\delta}_T \\ \dot{\delta}_E \\ \dot{\delta}_A \\ \dot{\delta}_R \end{bmatrix} = \begin{bmatrix} A_{Throttle} \\ A_{Elevator} \\ A_{Aileron} \\ A_{Rudder} \end{bmatrix} \begin{bmatrix} \delta_T \\ \delta_E \\ \delta_A \\ \delta_R \end{bmatrix} + \begin{bmatrix} B_{Throttle} \\ B_{Elevator} \\ B_{Aileron} \\ B_{Rudder} \end{bmatrix} \begin{bmatrix} \delta_{T\,cmd} \\ \delta_{E\,cmd} \\ \delta_{A\,cmd} \\ \delta_{R\,cmd} \end{bmatrix} \qquad \text{Eq. 3.3}$$

Velocities in the body frame are derived by transforming the total velocity by the airflow angles (angle of attack and sideslip), as shown below in Equation 3.4.

$$\begin{bmatrix} U \\ V \\ W \end{bmatrix} = V_T \begin{bmatrix} \cos\alpha\cos\beta \\ \sin\beta \\ \sin\alpha\cos\beta \end{bmatrix} \qquad \text{Eq. 3.4}$$

Non-dimensional force coefficients and stability-frame moment coefficients are derived using a linearized build-up method of perturbations from the trim conditions. The following equations obtain these coefficients for lift, drag, and side force, as well as roll, pitch, and yaw moments in the stability frame, respectively.

$$C_L = C_{L1} + C_{L_\alpha}(\alpha - \alpha_{Trim}) + C_{L_q}(Q - Q_{Trim})\left(\frac{\bar{c}}{2U_{Trim}}\right) + C_{L_{\dot{\alpha}}}\dot{\alpha}\left(\frac{\bar{c}}{2U_{Trim}}\right) +$$

$$C_{L_U}\left(\frac{U - U_{Trim}}{U_{Trim}}\right) + C_{L_{\delta E}}\left(\delta_E - \delta_{E_{Trim}}\right) \qquad \text{Eq. 3.5}$$

$$C_D = C_{D_0} + \frac{C_L^2}{\pi A_R e} \qquad \text{Eq. 3.6}$$

$$C_Y = C_{Y_\beta}\beta + C_{Y_P}(P - P_{Trim})\left(\frac{b}{2U_{Trim}}\right) + C_{Y_R}(R - R_{Trim})\left(\frac{b}{2U_{Trim}}\right) + C_{Y_{\delta_A}}\delta_A + C_{Y_{\delta_R}}\delta_R \qquad \text{Eq. 3.7}$$

$$C_l^S = C_{l_\beta}\beta + C_{l_P}(P - P_{Trim})\left(\frac{b}{2U_{Trim}}\right) + C_{l_R}(R - R_{Trim})\left(\frac{b}{2U_{Trim}}\right) + C_{l_{\delta_A}}\delta_A + C_{l_{\delta_R}}\delta_R \qquad \text{Eq. 3.8}$$

$$C_M^S = C_{M_1} + C_{M_\alpha}(\alpha - \alpha_{Trim}) + C_{M_Q}(Q - Q_{Trim})\left(\frac{\bar{c}}{2U_{Trim}}\right) + C_{M_U}\left(\frac{U - U_{Trim}}{U_{Trim}}\right) +$$

$$C_{M_{\delta_E}}\left(\delta_E - \delta_{E\,Trim}\right) + \ 2C_{M_1}\left(\frac{U-U_{Trim}}{U_{Trim}}\right) + \left(C_{M_{T_U}} + 2C_{M_{T_1}}\right)\left(\frac{U-U_{Trim}}{U_{Trim}}\right) + C_{M_{T_\alpha}}(\alpha - \alpha_{Trim}) \qquad \textit{Eq. 3.9}$$

$$C_N^S = \ C_{N_\beta}\beta + \ C_{N_P}(P - P_{Trim})\left(\frac{b}{2U_{Trim}}\right) + C_{N_R}(R - R_{Trim})\left(\frac{b}{2U_{Trim}}\right) + \ C_{N_{\delta_A}}\delta_A + C_{N_{\delta_R}}\delta_R \qquad \textit{Eq. 3.10}$$

Normalized coefficients for aerodynamic forces in the body frame are derived by transforming lift and drag coefficients into the body frame, as shown below in Equation 3.11.

$$\begin{bmatrix} C_{X_A} \\ C_{Y_A} \\ C_{Z_A} \end{bmatrix} = \begin{bmatrix} C_L \sin\alpha - \ C_D \cos\alpha \\ C_Y \\ -C_L \cos\alpha - \ C_D \sin\alpha \end{bmatrix} \qquad \textit{Eq. 3.11}$$

Normalized moment coefficients in the body frame are determined by transforming the stability-frame moment coefficients by the angle of attack, shown below in Equation 3.12.

$$\begin{bmatrix} C_l \\ C_M \\ C_N \end{bmatrix} = \begin{bmatrix} C_l^S \cos\alpha - C_N^S \sin\alpha \\ C_M^S \\ C_l^S \sin\alpha + C_N^S \cos\alpha \end{bmatrix} \qquad \textit{Eq. 3.12}$$

The total thrust force is approximated using Equation 3.13, where values for the 2nd order coefficients are derived from experimental data obtained from propulsion testing of the aircraft engines.

$$T = \ X_{T_0} + X_{T_1}\delta_T + \ X_{T_2}\delta_T^2 \qquad \textit{Eq. 3.13}$$

Gravitational accelerations in the body frame are determined through Euler-angle transformations, shown in Equation 3.14.

$$\begin{bmatrix} G_X \\ G_Y \\ G_Z \end{bmatrix} = \ G \begin{bmatrix} -\sin\theta \\ \sin\varphi\cos\theta \\ \cos\varphi\cos\theta \end{bmatrix} \qquad \textit{Eq. 3.14}$$

The dynamic pressure experienced by the airframe is calculated from the density and total airspeed, shown below in Equation 3.15.

$$\bar{q} = \frac{1}{2}\rho V_T^2 \qquad \textit{Eq. 3.15}$$

Accelerations in the body frame due to aerodynamic and propulsive forces are obtained using their respective normalized coefficients and the aircraft mass, shown below in Equation 3.16.

$$\begin{bmatrix} A_X \\ A_Y \\ A_Z \end{bmatrix} = \begin{bmatrix} C_{X_A}\bar{q}S + T \\ C_{Y_A}\bar{q}S \\ C_{Z_A}\bar{q}S \end{bmatrix} m^{-1} \qquad \textit{Eq. 3.16}$$

Total moment calculations due to aerodynamic and propulsive moments are determined using their respective normalized coefficients, as shown below in Equation 3.17, where l_T indicates the moment arm along the Z-body axis between the thrust centerline and the center of gravity of the aircraft.

$$\begin{bmatrix} L \\ M \\ N \end{bmatrix} = \begin{bmatrix} C_l\bar{q}Sb \\ C_m\bar{q}S\bar{c} - Tl_T \\ C_n\bar{q}Sb \end{bmatrix} \qquad \textit{Eq. 3.17}$$

Derivatives for aircraft body velocities are calculated using respective aerodynamic and propulsive accelerations, gravitational accelerations, body velocities, and body rotational velocities, as shown below in Equation 3.18.

$$\begin{bmatrix} \dot{U} \\ \dot{V} \\ \dot{W} \end{bmatrix} = \begin{bmatrix} A_X + G_X + RV - QW \\ A_Y + G_Y - RU + PW \\ A_Z + G_Z + QU - PV \end{bmatrix} \qquad \textit{Eq. 3.18}$$

Finally, using these calculations, the state derivatives can be determined and used to propagate the aircraft states using integration schemes such as RK-4. The state derivatives can be calculated using the following equations.

$$\dot{V}_T = \frac{U\dot{U} + V\dot{V} + W\dot{W}}{V_T} \qquad \textit{Eq. 3.19}$$

$$\dot{\alpha} = \frac{U\dot{W} - W\dot{U}}{U^2 + W^2} \qquad \textit{Eq. 3.20}$$

$$\dot{\beta} = \frac{V(U^2+W^2)-\dot{V}(\dot{U}U+\dot{W}W)}{{V_T}^2\sqrt{U^2+W^2}} \qquad \textit{Eq. 3.21}$$

$$\dot{\varphi} = P + \ (R\cos\varphi + Q\sin\varphi)\tan\theta \qquad \textit{Eq. 3.22}$$

$$\dot{\theta} = Q\cos\varphi - R\sin\varphi \qquad \textit{Eq. 3.23}$$

$$\dot{\psi} = \ (Q\sin\varphi + R\cos\varphi)\cos\theta \qquad \textit{Eq. 3.24}$$

$$\dot{P} = \frac{I_{ZZ}L+I_{XZ}N-\left(I_{XZ}(I_{YY}-I_{XX}-I_{ZZ})P+\left(I_{XZ}^2+I_{ZZ}(I_{ZZ}-I_{YY})\right)R\right)Q}{I_{XX}I_{ZZ}-I_{XZ}^2} \qquad \textit{Eq. 3.25}$$

$$\dot{Q} = \frac{M-(I_{XX}-I_{ZZ})PR-I_{XZ}(P^2-R^2)}{I_{YY}} \qquad \textit{Eq. 3.26}$$

$$\dot{R} = \frac{I_{XZ}L+I_{XX}N+\left(I_{XZ}(I_{YY}-I_{XX}-I_{ZZ})R+\left(I_{XZ}^2+I_{XX}(I_{XX}-I_{YY})\right)P\right)Q}{I_{XX}I_{ZZ}-I_{XZ}^2} \qquad \textit{Eq. 3.27}$$

3.2 Aircraft Platforms

In this section, the aircraft platforms used for simulation and flight test validation will be discussed.

3.2.1 CReSIS Unmanned Aircraft Platforms

The developed autonomy in this work will be demonstrated in simulations using CReSIS UAS platforms from both past and future deployments in order to assess improvements in operational capabilities and performance. The G1X platforms, shown in Figure 47, are a class of modified 40% scale Yak-54 model aircraft that have been repurposed as research platforms by the University of Kansas for polar missions and flight test research. The G1X platforms, along with 33% scale YAK-54 models, have been successfully operated at both the Greenland and Antarctic ice caps, and their deployment history is outlined in Section 1.2. The role of these platforms transitioned from mere pilot training to payload-carrying mission systems following the miniaturization of the onboard radar system.

For polar research deployments, the autopilot system utilized was a COTS system developed by AeroVironment (formerly Pulse Aerospace), a local company founded by KU Aerospace Graduates who were vital in the development and testing of the Meridian aircraft [21]. For flight control research, the KU Flight Research lab has also developed a custom autopilot system for the G1X systems that has been utilized for flight test validation of research projects including optimal control, detect-and-avoid, and cascading autopilot control systems [72].

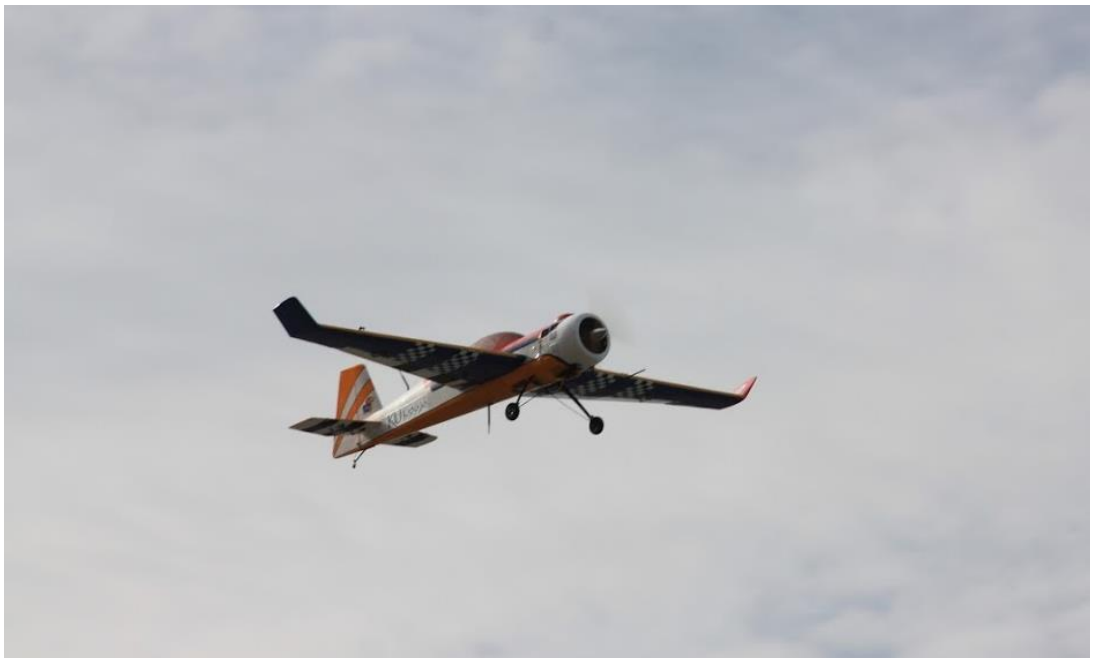


Figure 47: G1X Aircraft (Photo Credit: KU Flight Research Lab)

The CReSIS UAS team has chosen a series of twin-boom configuration VTOL unmanned systems, developed by Mugin UAV Systems, for upcoming deployments to the Helheim Glacier. The incorporation of VTOL capabilities are necessary for operations in this region due to the rugged terrain surrounding the outlet glacier. Two sizes of platforms are being developed for these operations, with the size tradeoff involving incorporated cost and risk, as well as associated antenna frequency and overall mission endurance. The Mugin-2930 platform, shown in Figure 48, is being developed by the CReSIS UAS team for the initial trial deployment to the region, while the larger Mugin-4450 platform is being developed for subsequent deployment for extensive data collection. The Mugin-4450 platform incorporates a similar design and configuration as the Mugin-2930, but is scaled in size based on the increased wingspan, and is structurally reinforced appropriately to carry increased payload weight and volume.

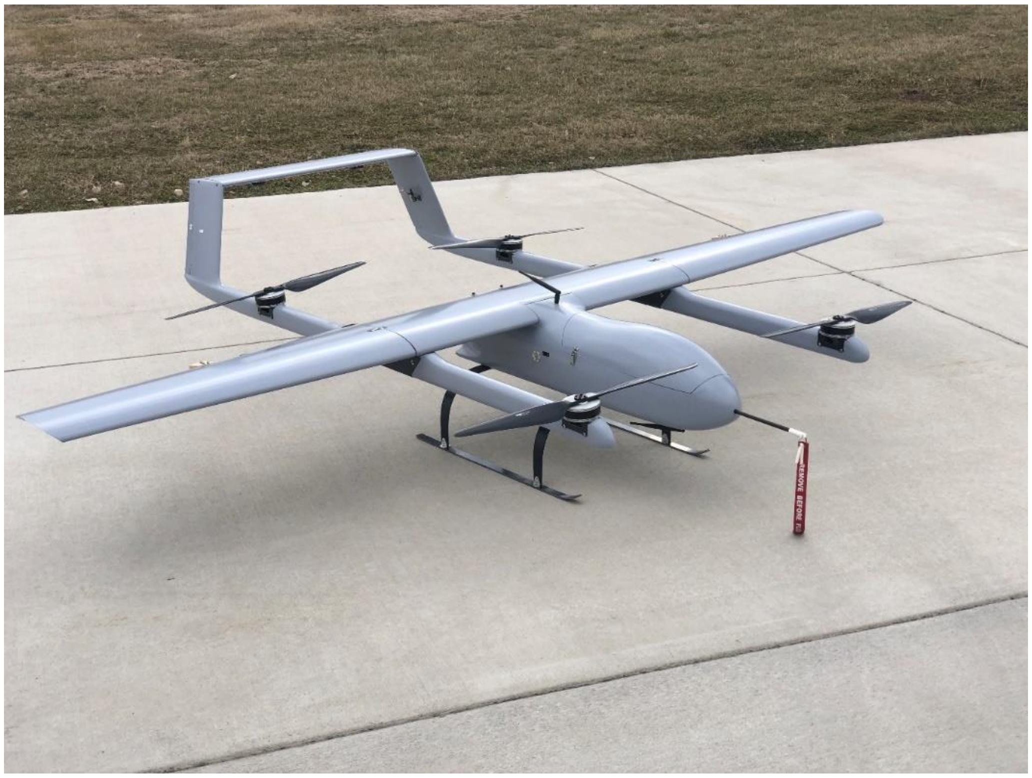


Figure 48: Mugin-2930 VTOL (Photo Credit: KU Flight Research Lab)

Flight control of the Mugin systems will be achieved using ArduPilot QuadPlane software aboard a Pixhawk Cube unit. Ardupilot is an open source autopilot project with various functionality and flight modes for a myriad of unmanned aircraft configurations, as well as other unmanned robotic platforms such as rovers, surface, and underwater vehicles. The QuadPlane package incorporates both fixed-wing and multi-rotor flight modes, as well as uses the multi-rotors to assist fixed-wing flight for transitioning between flight modes, stall prevention, and aircraft stability robustness. It also has extensive functionalities for various mission profiles, and has a user-friendly ground station interface known as Mission Planner. The Pixhawk Cube is a commercially available all-in-one flight control unit with triple redundant, vibration isolated, and internally heated IMUs. The Pixhawk Cube has 14 PWM servo outputs, and has the ability to integrate various sensors such as Differential GPS and altimeters, which would improve our polar mission capabilities. Future validation for the developed autonomy aboard the Mugin systems is planned to be achieved utilizing a cascading autopilot design. In this framework, an onboard computer running the autonomy will command PWM (pulse-width modulation) signals into the Pixhawk system corresponding to roll, pitch, and airspeed commands, and the ArduPilot flight control system will regulate these aircraft states using the various motors and control surfaces of the aircraft. This cascading autopilot framework is shown in Figure 49, where an Arduino board is used to switch between manual flight control commands and the autonomous commands from the onboard computer, generating the corresponding PWM signals as inputs into the Pixhawk flight control unit.

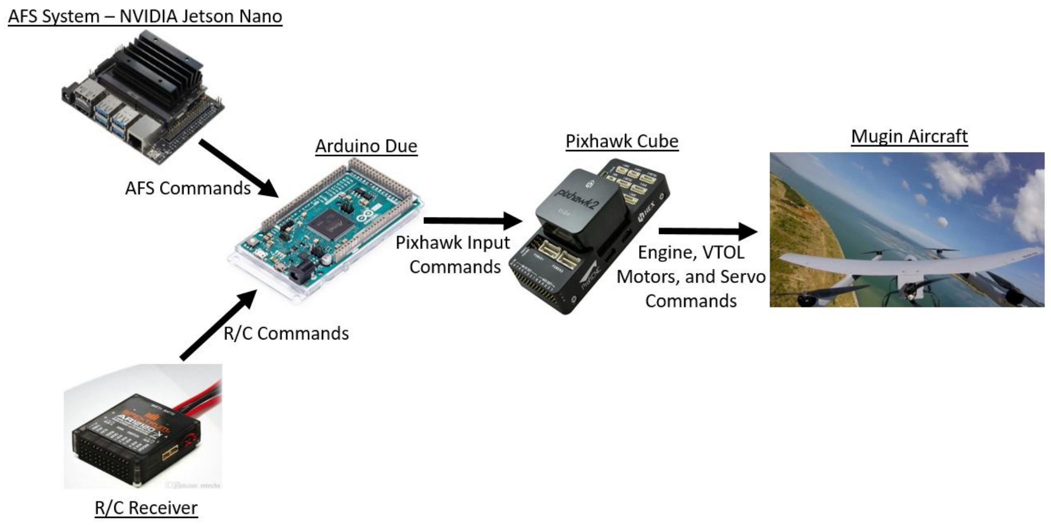


Figure 49: Cascading Autopilot Framework

3.2.2 Surrogate Platforms

Flight testing validation of the developed autonomy in this work is conducted using SkyHunter platforms, which are small commercially available fixed-wing platforms commonly used for FPV applications. The SkyHunter is a twin-boom configuration aircraft composed of EPO foam, and the KU Flight Research Lab has been utilizing these platforms for multi-agent research since 2017. The relatively high payload to weight ratio of these platforms makes them extremely suitable for implementing various airborne sensor packages, and their reduced airframe cost and size minimizes the associated risks involved in local flight testing operations. Because of this, the KU Flight Research Lab has utilized these aircraft for low-cost, low-risk flight testing of airborne collision avoidance, formation flight, and advanced propulsion systems [73]. Additional aerodynamic and structural modifications have been made by the KU Flight Research Team in order to improve flight performance and system reliability [74].

The SkyHunter platforms incorporate a Pixhawk Cube unit for data acquisition and PWM generation. An onboard computer running the developed autonomy is used to generate commands corresponding to the desired PWM signals for motor and control surface deflections. The Pixhawk flight control unit toggles control commands from the RC pilot and the onboard computer based on the “RC_OVERRIDE” flag in its PX4 firmware. The SkyHunter aircraft is shown in Figure 50.

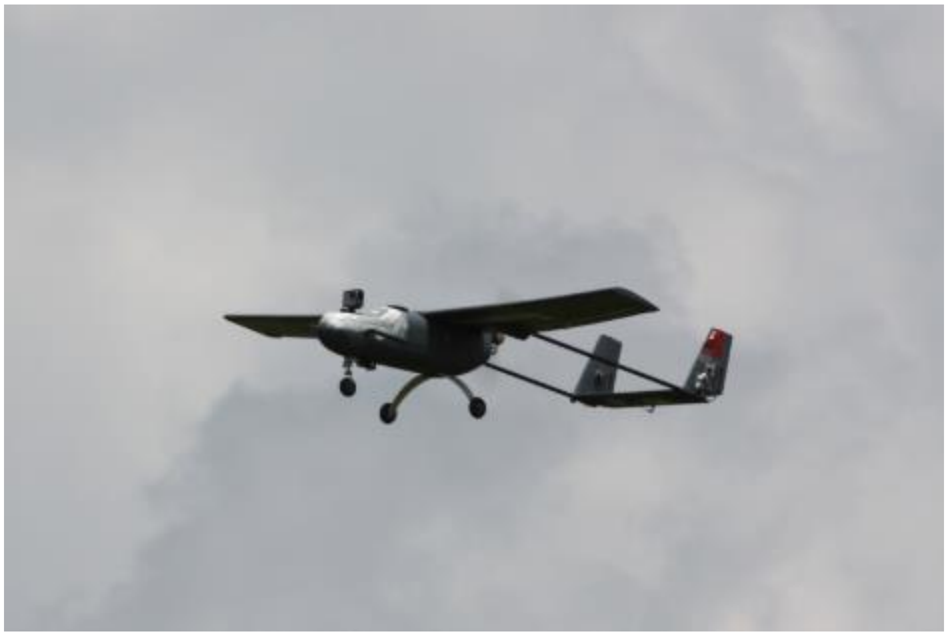


Figure 50: SkyHunter UAS (Photo Credit: KU Flight Research Lab)

3.2.3 Platforms Characteristics

Key characteristics of the platforms used in this work are presented in Table 3, where aircraft dynamic modes were determined using AAA dynamic modeling software.

Table 3: Platform Characteristics

Characteristic	G1X	Mugin-2930	Mugin-4450	SkyHunter
Wingspan	14.5 feet	9.6 feet	14.6 feet	5.9 feet
Takeoff Weight	73 lbs	55 lbs	90 lbs	8.5 lbs
Trim Airspeed	65 knots	48 knots	61 knots	30 knots
Range	82 nmi	72 nmi	435 nmi	15 nmi
Aspect Ratio	6.85	9.5	8.4	9.47
Wing Area	24.66 ft^2	9.4 ft^2	25.4 ft^2	4.79 ft^2
Payload Capacity	5 lbs	13 lbs	30 lbs	1 lb
Endurance	75 minutes	90 minutes	7 hours	30 minutes
Propulsion Type	ICE	Electric/ICE	ICE	Electric

3.3 Lateral Guidance

The role of the lateral guidance in this work is to determine the desired sideslip (β_{cmd}) and roll angle (ϕ_{cmd}) that the aircraft should obtain in order to precisely track the desired lateral trajectory (Equation 3.28). In this work, the sideslip command is always set to zero in order to achieve coordinated turns

(Equation 3.29). The method for determining the roll angle command depends on the desired trajectory type, which will be discussed in the following subsections.

$$G_{Lat} = \begin{bmatrix} \beta_{cmd} \\ \phi_{cmd} \end{bmatrix} \qquad \textit{Eq. 3.28}$$

$$\beta_{cmd} = 0 \qquad \textit{Eq. 3.29}$$

3.3.1 Loiter Guidance

For tracking the home loiter circle, the aircraft utilizes a nonlinear guidance method introduced in [75], which projects a reference point R a fixed distance L_{1C} ahead of the aircraft onto the desired trajectory. This method then uses the reference point location to determine the required lateral acceleration needed to intercept the point. In order to track a circular trajectory defined by center C = $\{C_N, C_E\}$ and radius C_R, the aircraft first determines the distance from the aircraft position P = $\{P_N, P_E\}$ to the center C (Equation 3.30), and then determines the unit vector from C to P (Equation 3.31) to create projection point D (Equation 3.32). Next, the heading angle from P to C is found (Equation 3.33) and used to solve for the relative heading change from the aircraft's heading (Equation 3.34). In order to place point R onto the circumference of the circle a fixed distance L_{1C} away from P, the angular change from ψ_{PC} can be found by using the law of cosines and the three known legs of the triangle (Equations 3.35). Using the relative heading between the aircraft heading and C, the tracking orientation of the circle is determined to be either clockwise or counterclockwise, and this determination is used to apply the angular change needed to appropriately place point R (Equation 3.36). This heading is then used to determine the unit vector (Equation 3.37), and place point R (Equation 3.38). However, note that if the aircraft is not within L_{1C} of the circumference of the circle, point R is simply placed L_{1C} towards point D. Once the reference point R has been placed, the heading from the aircraft position to point R is determined (Equation 3.39), and used to determine the required heading change and is saturated between $\pm\frac{\pi}{2}$ (Equation 3.40). This resulting angle is then used to calculated the required lateral acceleration needed to intercept point R (Equation 3.41). Finally, this lateral acceleration can be translated into an aircraft bank angle command (Equation 3.42). These lateral guidance equations for the loiter tracking are provided below, as well as depicted in Figure 51.

$$D_C = \|C - P\| \qquad \textit{Eq. 3.30}$$

$$\hat{V}_{CP} = \frac{P-C}{\|P-C\|} \qquad \textit{Eq. 3.31}$$

$$D = C + C_R\hat{V}_{CP} \qquad \textit{Eq. 3.32}$$

$$\psi_{PC} = atan2\left(\frac{C_E-P_E}{C_N-P_N}\right) \qquad \textit{Eq. 3.33}$$

$$\eta_C = W_{-\pi}^{\pi}[\psi_A - \psi_{PC}] \qquad \textit{Eq. 3.34}$$

$$\theta_R = \cos^{-1}\left(\frac{C_R^2 + D_C^2 - L_{1C}^2}{2RD_C}\right) \qquad \textit{Eq. 3.35}$$

$$\psi_{CR} = \begin{cases} W_{-\pi}^{\pi}[(\psi_{PC} + \pi) + \theta_R] & if(\eta_C < 0) \\ W_{-\pi}^{\pi}[(\psi_{PC} + \pi) - \theta_R] & if(\eta_C \geq 0) \end{cases} \qquad \textit{Eq. 3.36}$$

$$\hat{V}_{CR} = \langle \cos(\psi_{CR}), \sin(\psi_{CR}) \rangle \qquad \textit{Eq. 3.37}$$

$$R = \begin{cases} P + L_{1C}\hat{V}_{CP} & if\big(D_C \leq (C_R - L_{1C})\big) \\ P - L_{1C}\hat{V}_{CP} & if\big(D_C \geq (C_R + L_{1C})\big) \\ C + C_R\hat{V}_{CR} & if((C_R - L_{1C})) < D_C < (C_R + L_{1C})) \end{cases} \qquad \textit{Eq. 3.38}$$

$$\psi_R = atan2\left(\frac{R_E-P_E}{R_N-P_N}\right) \qquad \textit{Eq. 3.39}$$

$$\eta_{Lat} = sat_{-\pi/2}^{\pi/2}[W_{-\pi}^{\pi}[\psi_R - \psi_A]] \qquad \textit{Eq. 3.40}$$

$$a_{cmd} = \frac{2V_G^2\sin(\eta_{Lat})}{L_{1C}} \qquad \textit{Eq. 3.41}$$

$$\varphi_{cmd} = sat_{\varphi_{Min}}^{\varphi_{Max}}\left(\tan^{-1}\left(\frac{a_{cmd}}{g}\right)\right) \qquad \textit{Eq. 3.42}$$

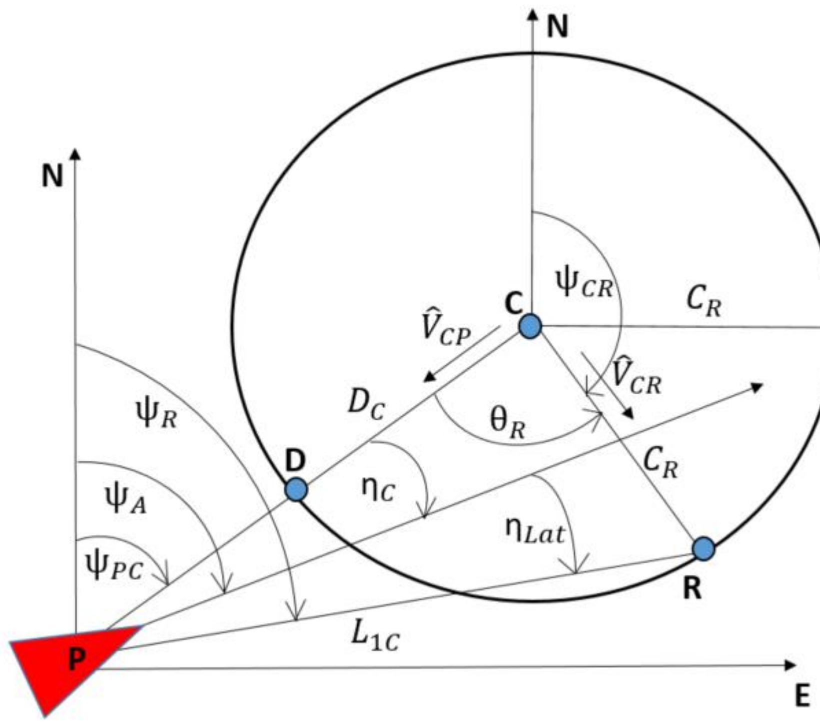


Figure 51: Circular Lateral Guidance Graphic

3.3.2 Flight Line Guidance

When the aircraft is surveying a mission flight line, tight tracking and low oscillations are desired. For this reason, L2+ methods are used for lateral guidance [76]. In order to track a line segment defined by starting point A = {A_N, A_E} and endpoint B = {B_N, B_E}, the aircraft first determines the trackline unit vector $\hat{V}_{TL}$ in the local frame from point A to point B (Equation 3.43). This unit vector is then used to project the aircraft's position onto the line as point D (Equation 3.44). The distance from the aircraft to point D is then determined (Equation 3.45), along with the unit vector from the aircraft to point D (Equation 3.46). If the distance to point D is greater than the trackline projection length parameter L_{1TL}, then point R is simply projected a distance L_{1TL} along the unit vector from the aircraft towards the projection point D. Otherwise, point R is placed L_{1TL} away from the aircraft and onto the line, by projecting ahead of point D along trackline AB a distance determined using Pythagorean theorem (Equation 3.47). Note that this projection length parameter L_{1TL} is differentiated from the L_{1C} parameter used for curved trajectories. Once point R has been placed, the heading from the aircraft position to point R is determined (Equation 3.48), and the desired heading change η_{Lat} is found and saturated between $\pm\frac{\pi}{2}$ (Equation 3.49). This angle is then used to determine the required lateral acceleration using the tuning parameter T^*, which accounts for the delay due to the roll mode of the aircraft, as well as the flight controller. Additionally, an integral term is applied to the resulting lateral acceleration in order to reduce the steady state tracking error to zero (Equation 3.50). Note that this integral is reset to zero following completion of the flight line. Finally, once the lateral acceleration command has been determined, it is used to calculate the commanded aircraft bank angle and is saturated between the minimum and maximum desired bounds (Equation 3.51). These equations for flight line lateral guidance are provided below, as well as depicted in Figure 52.

$$\hat{V}_{TL} = \frac{B-A}{\|B-A\|} \qquad \text{Eq. 3.43}$$

$$D = \frac{\hat{V}_{TL}\hat{V}_{TL}^T}{\hat{V}_{TL}^T\hat{V}_{TL}}P + \left(I - \frac{\hat{V}_{TL}\hat{V}_{TL}^T}{\hat{V}_{TL}^T\hat{V}_{TL}}\right)A \qquad \text{Eq. 3.44}$$

$$D_{TL} = \|D - P\| \qquad \text{Eq. 3.45}$$

$$\hat{V}_{PD} = \frac{D-P}{\|D-P\|} \qquad \text{Eq. 3.46}$$

$$R = \begin{cases} P + L_{1TL}\hat{V}_{PD} & if(D_{TL} \geq L_{1TL}) \\ D + \hat{V}_{TL}\sqrt{L_{1TL}^2 - D_{TL}^2} & if(D_{TL} < L_{1TL}) \end{cases} \qquad \text{Eq. 3.47}$$

$$\psi_R = atan2\left(\frac{R_E - P_E}{R_N - P_N}\right) \qquad \text{Eq. 3.48}$$

$$\eta_{Lat} = sat_{-\pi/2}^{\pi/2}\{wrap_{-\pi}^{\pi}[\psi_R - \psi_A]\} \quad ` \qquad \text{Eq. 3.49}$$

$$a_{cmd} = \frac{2V_G \sin(\eta_{Lat})}{T^*} + K_I \int D_{TL} \qquad \textit{Eq. 3.50}$$

$$\varphi_{cmd} = sat_{\varphi_{Min}}^{\varphi_{Max}} \left\{ tan^{-1}\left(\frac{a_{cmd}}{g}\right)\right\} \qquad \textit{Eq. 3.51}$$

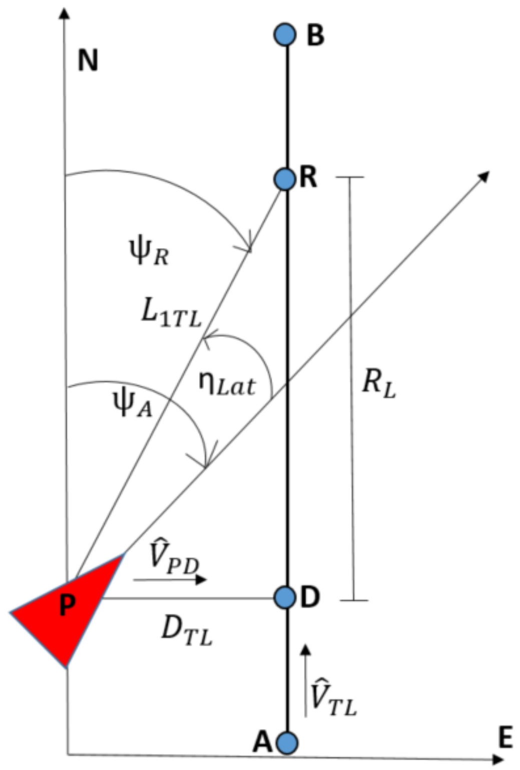


Figure 52: Flight Line Lateral Guidance Graphic

3.3.3 Dubins Path Guidance

The lateral guidance method used for tracking Dubins Paths in this work is a modified version of the nonlinear trajectory guidance methods for curved trajectories outlined in Section 3.3.1. However, instead of using a fixed distance between the aircraft and the reference point, the reference point is simply placed at location of the end of the first arc in the Dubins Path. By doing this, the resulting lateral acceleration command is equal to the centripetal acceleration of the first arc component in the Dubins Path. An example projection of the reference point onto the Dubins Path for architecture Type 3 is shown in Figure 53.

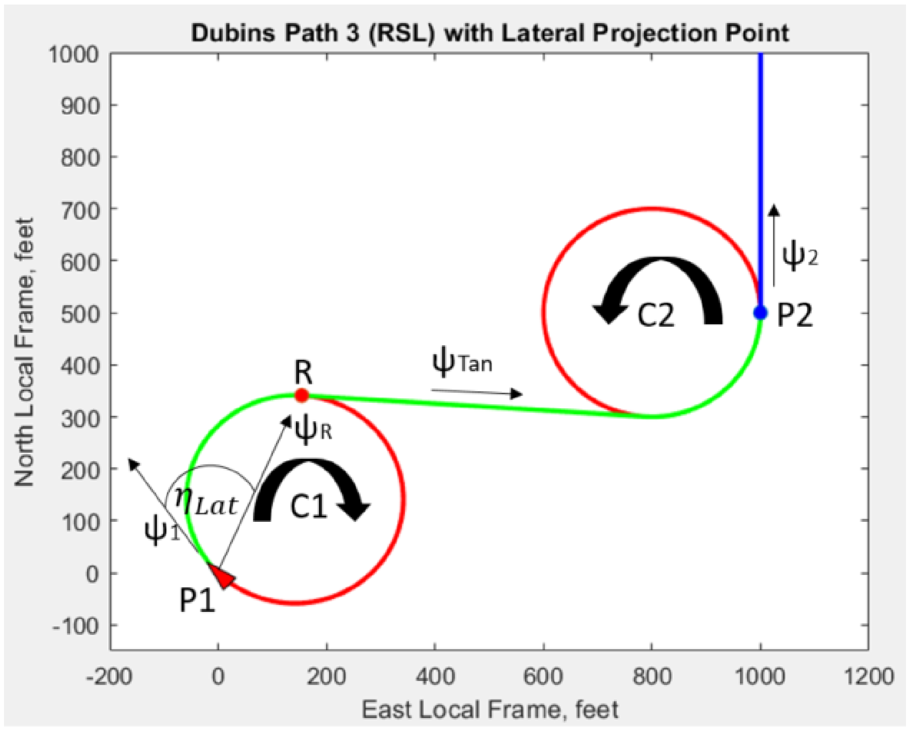


Figure 53: Dubins Path Guidance Example Graphic

3.3.4 Switching Logics

The switching logic between the various trajectory types are discussed in this section. Firstly, when the aircraft traverses to the home loiter circle, it initially follows the tracking of the home Dubins Paths described in Section 2.2.3. While following this Dubins Path, the aircraft utilizes the Dubins Path guidance methods outlined in Section 3.3.3. However, when the aircraft arrives within twice the radius of the home loiter circle, it transitions to the loiter guidance methods described in Section 3.3.1 in order to smoothly loiter around the home location. This criteria is shown in Equation 3.52, using terminology from Section 3.3.1 and Figure 51.

$$D_C < 2C_R \qquad \textit{Eq. 3.52}$$

The next switching logic that needs to be defined is when the aircraft transitions from following the Dubins Paths approaching the mission flight lines to surveying the flight lines themselves. For this, two criteria must be met: (1) The aircraft must be within a distance D_{SW} of the flight line start location, and (2) the summation of the angles comprising the arcs in the Dubins Path must be less than θ_{SW}. Once both of these criteria (shown in Equation 3.53) are satisfied, the aircraft beings to track the flight line using methods outlined in Section 3.3.2. Figure 54 depicts the distance criteria, while Figure 55 shows an example of why the angular criteria is also necessary to prevent premature tracking logic switches, as it is necessary for the aircraft to track the Dubins Path as long as possible in order to arrive at the beginning of the flight line with the required heading angle.

$$(D_{12} < D_{SW}) \quad \&\& \quad \left(\sum_{i=1}^{N_\theta} \left| C_{i_\theta} \right| < \theta_{SW} \right) \qquad \textit{Eq. 3.53}$$

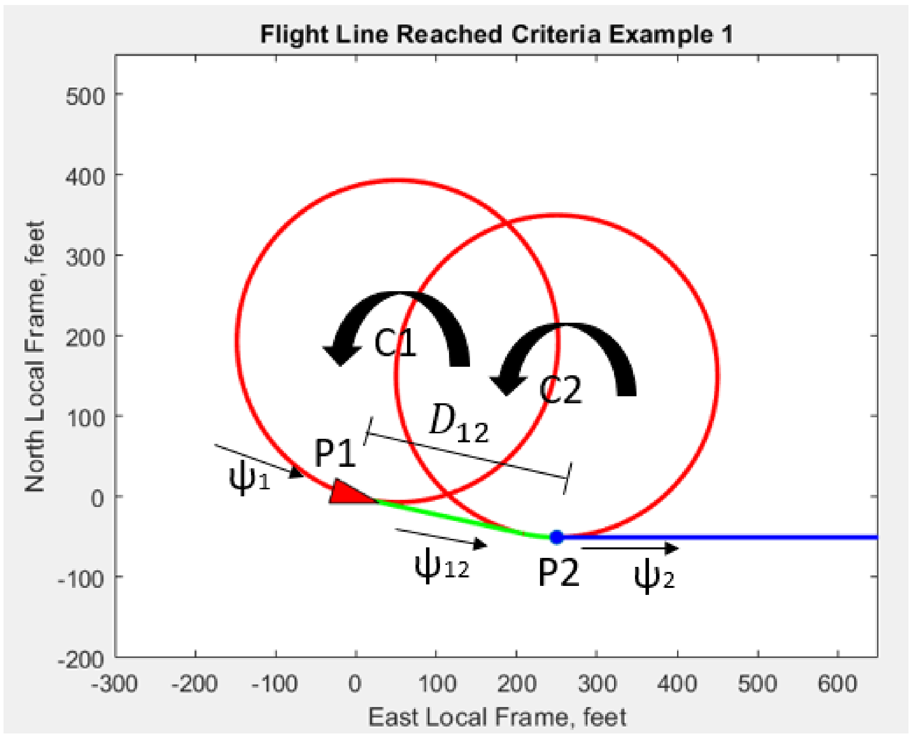


Figure 54: Flight Line Reached Criteria 1 Example

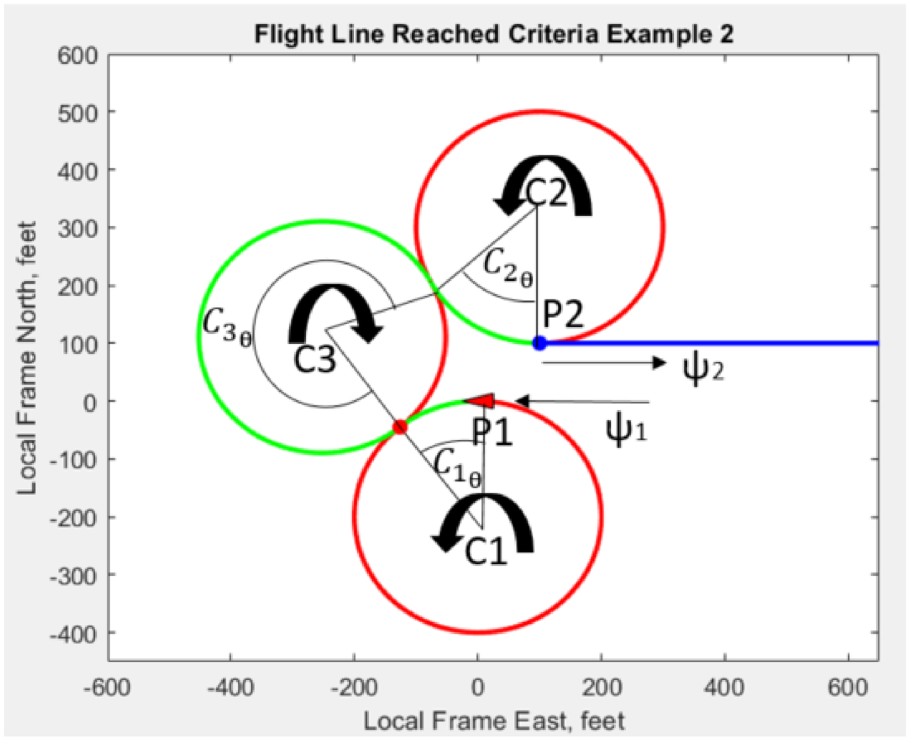


Figure 55: Flight Line Reached Criteria 2 Example

For end-to-end surveying of the mission flight, the aircraft should continue tracking the flight line until it has crossed the flight line endpoint. More specifically, the criteria for finishing a flight line is when the aircraft crosses the half-plane formed by the flight line endpoint. The satisfying criterion for finishing straight-line tracking logic is outlined below, for line segments formed by start and endpoints A = {A_N, A_E}, and B = {B_N, B_E}, respectively. First, the heading of the flight line is determined using a four-quadrant arctangent function (Equation 3.54), then used to determine the unit vector of the flight line (Equation 3.55). This unit vector is used to project the position of the aircraft P = {P_N, P_E} onto the flight line (Equation 3.56), and this projected position D and flight line heading ψ_{TL} is compared with the endpoint position to determine if it has crossed the half-plane (Equation 3.57). The half-plane criteria is shown in Figure 56.

$$\psi_{TL} = atan2\left(\frac{B_E - A_E}{B_N - A_N}\right) \qquad \textit{Eq. 3.54}$$

$$\hat{V}_{TL} = \langle \cos(\psi_{TL}), \sin(\psi_{TL}) \rangle \qquad \textit{Eq. 3.55}$$

$$D = \frac{\hat{V}_{TL}\hat{V}_{TL}^T}{\hat{V}_{TL}^T\hat{V}_{TL}} P + \left(I - \frac{\hat{V}_{TL}\hat{V}_{TL}^T}{\hat{V}_{TL}^T\hat{V}_{TL}}\right) A \qquad \textit{Eq. 3.56}$$

$$S_{TL} = \begin{cases} 1 & if \begin{cases} D_E \leq B_E & if(-\pi < \psi_{TL} < 0) \\ D_N \geq B_N & if(\psi_{TL} = 0) \\ D_E \geq B_E & if(0 < \psi_{TL} < \pi) \\ D_N \leq B_N & if(\psi_{TL} = \pi) \end{cases} \\ 0 & otherwise \end{cases} \qquad \textit{Eq. 3.57}$$

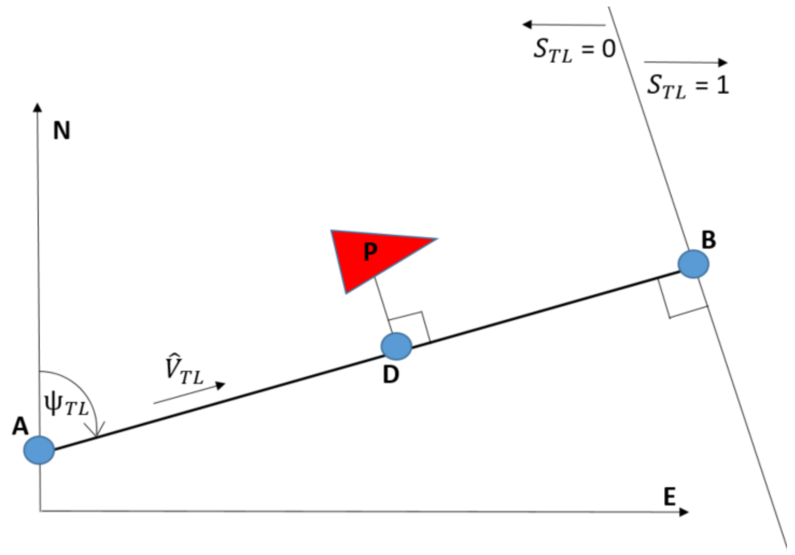


Figure 56: Flight Line Survey - End Condition Graphic

3.4 Longitudinal Guidance

The role of the longitudinal guidance in this work is to provide the desired total velocity (V_T) and pitch angle (θ) that the aircraft should obtain in order to precisely track the desired altitude and airspeed (Equation 3.58). In this work, the commanded total velocity is always set to the trim airspeed for the vehicle (Equation 3.59).

$$G_{Lon} = \begin{bmatrix} V_{T\,cmd} \\ \theta_{cmd} \end{bmatrix} \qquad \textit{Eq. 3.58}$$

$$V_{T\,cmd} = V_{T\,Trim} \qquad \textit{Eq. 3.59}$$

For the longitudinal plane, an altitude command is generated by the path planning either using standard waypoint altitudes or terrain tracking methods. A PD control scheme is applied to this desired altitude in order to smoothly minimize the altitude error. First, the sign and magnitude of the altitude error rate $\dot{h}_{error}$ is determined using the velocity in the Z axis of the inertial frame V_{ZI} and comparing the commanded altitude h_{cmd} with the current altitude h (Equation 3.60). Next, a proportional and derivative gain P$_G$ and D$_G$ are used to calculate a commanded climb rate (Equation 3.61), which is then saturated between a minimum and maximum allowable climb rate command bounds (Equation 3.62). Using the inertial ground speed V_G, the commanded flight path angle is calculated (Equation 3.63) [75], and the trim angle of attack α_{trim} is accounted for to calculate the resulting commanded aircraft pitch angle (Equation 3.64). Finally, the output pitch angle command is saturated within allowable bounds (Equation 3.65).

$$\dot{h}_{error} = \begin{cases} V_{ZI} & if(h_{cmd} \geq h) \\ -V_{ZI} & if(h_{cmd} < h) \end{cases} \qquad \textit{Eq. 3.60}$$

$$\dot{h}_{cmd} = (h_{cmd} - h)P_G + (\dot{h}_{error})D_G \qquad \textit{Eq. 3.61}$$

$$\dot{h}_{cmd} = sat_{\dot{h}_{cmd_{Min}}}^{\dot{h}_{cmd_{Max}}}\{\dot{h}_{cmd}\} \qquad \textit{Eq. 3.62}$$

$$\gamma_{cmd} = \tan^{-1}\left(\frac{\dot{h}_{cmd}}{V_G}\right) \qquad \textit{Eq. 3.63}$$

$$\theta_{cmd} = \gamma_{cmd} + \alpha_{trim} \qquad \textit{Eq. 3.64}$$

$$\theta_{cmd} = sat_{\theta_{cmd_{Min}}}^{\theta_{cmd_{Max}}}\{\theta_{cmd}\} \qquad \textit{Eq. 3.65}$$

3.5 Aircraft Control Architecture

The purpose of the aircraft control architecture is to determine the necessary control surface deflections required in order to maintain the desired aircraft state commands determined by the aircraft guidance logic. In this work the longitudinal and lateral-directional states are assumed to be decoupled, and a classical LQR longitudinal controller and a Non-Zero Set Point LQR (NZSP-LQR) lateral-directional control scheme are used to regulate the aircraft state about a trim condition (Equation 3.66 and Equation 3.67), where the augmented longitudinal states contain state errors and integral errors from the guidance commands [72,78,79]. The control matrices for the longitudinal and lateral-directional state spaces are shown in Equation 3.68 and Equation 3.69, representing throttle and elevator, and aileron and rudder, respectively. Optimal gain matrices K_{Lon} and K_{Lat} are calculated to minimize the cost function described by (Equation 3.70), given predefined values for the state costs Q and control costs R. The resulting perturbed LQR control outputs are then combined with the trim control deflections to calculate the total control outputs, shown in Equation 4.71 and 4.72, where a stability-augmented pitch rate feedback is added for improved longitudinal control. Finally, saturation bounds are applied to the resulting control outputs and their respective rates, shown in Equations 3.73 and 3.74.

$$x_{lon} = \begin{bmatrix} U \\ \alpha \\ \theta \\ q \\ \int U_{error} \\ \int \theta_{error} \end{bmatrix} \qquad \textit{Eq. 3.66}$$

$$x_{lat} = \begin{bmatrix} \beta \\ \varphi \\ p \\ r \end{bmatrix} \qquad \textit{Eq. 3.67}$$

$$u_{lon} = \begin{bmatrix} \delta_T \\ \delta_E \end{bmatrix} \qquad \textit{Eq. 3.68}$$

$$u_{lat} = \begin{bmatrix} \delta_A \\ \delta_R \end{bmatrix} \qquad \textit{Eq. 3.69}$$

$$J = \int (x^T Q x + u^T R u) dt \qquad \textit{Eq. 3.70}$$

$$\begin{bmatrix}\delta_T\\ \delta_E\end{bmatrix} = \begin{bmatrix}\delta_{T_{trim}}\\ \delta_{E_{trim}}\end{bmatrix} + K_{Lon}\begin{bmatrix}U_{cmd}-U\\ \alpha_{trim}-\alpha\\ \theta_{cmd}-\theta\\ Q_{trim}-Q\\ \int(U_{cmd}-U)\\ \int(\theta_{cmd}-\theta)\end{bmatrix} \qquad \text{Eq. 3.71}$$

$$\begin{bmatrix}\delta_A\\ \delta_R\end{bmatrix} = \begin{bmatrix}\delta_{A_{trim}}\\ \delta_{R_{trim}}\end{bmatrix} + K_{Lat}\begin{bmatrix}\beta_{cmd}-\beta\\ \varphi_{cmd}-\varphi\\ P_{trim}-P\\ R_{trim}-R\end{bmatrix} \qquad \text{Eq. 3.72}$$

$$\begin{bmatrix}\delta_T\\ \delta_E\end{bmatrix} = \begin{bmatrix}\begin{bmatrix}Sat_{\delta_{T_{Min}}}^{\delta_{T_{Max}}}\left(\delta_{T_{i-1}} + Sat_{\delta_{dT_{Min}}}^{\delta_{dT_{Max}}}\left(\delta_{T_i}-\delta_{T_{i-1}}\right)\right)\\ Sat_{\delta_{E_{Min}}}^{\delta_{E_{Max}}}\left(\delta_{E_{i-1}} + Sat_{\delta_{dT_{Min}}}^{\delta_{dT_{Max}}}\left(\delta_{E_i}-\delta_{E_{i-1}}\right)\right)\end{bmatrix}\end{bmatrix} \qquad \text{Eq. 3.73}$$

$$\begin{bmatrix}\delta_A\\ \delta_R\end{bmatrix} = \begin{bmatrix}\begin{bmatrix}Sat_{\delta_{A_{Min}}}^{\delta_{A_{Max}}}\left(\delta_{A_{i-1}} + Sat_{\delta_{dA_{Min}}}^{\delta_{dA_{Max}}}\left(\delta_{A_i}-\delta_{A_{i-1}}\right)\right)\\ Sat_{\delta_{R_{Min}}}^{\delta_{R_{Max}}}\left(\delta_{E_{i-1}} + Sat_{\delta_{dR_{Min}}}^{\delta_{dR_{Max}}}\left(\delta_{R_i}-\delta_{R_{i-1}}\right)\right)\end{bmatrix}\end{bmatrix} \qquad \text{Eq. 3.74}$$

3.6 Morphing Potential Collision Avoidance

The collision avoidance mechanism applied in this work are morphing potential fields, which are a modified form of artificial potential fields where the shape of the potential is morphed as a function of the relative position and velocity between the aircraft and obstacle. This approach was developed specifically for fixed-wing aircraft avoidance, where high velocity and high inertia properties require avoidance maneuvers to begin early, but are desired to minimize the deviation from the original flight path [80,81]. While standard artificial potential fields produce uniform spatial potential about the obstacle (as shown in Equation 3.75), MPFs shift the potential by a factor S in the direction of the relative positions (Equation 3.76), where the morphing term Γ is determined using the relative velocity angle η_V between the aircraft and obstacles (Equation 3.77), where $\Gamma_0 \in (0,1]$ and $\eta_V \in [-\pi, \pi]$.

$$pf = Aexp\left\{-\left(\frac{\|\vec{p}^{Obj}-\vec{p}^{0}\|}{\sigma}\right)^2\right\} \qquad \textit{Eq. 3.75}$$

$$mpf = exp\left\{-\Gamma\left(\frac{\|\vec{p}^{Obj}-\vec{p}^{0}+\vec{s}\|}{\sigma}\right)^2\right\} \qquad \textit{Eq. 3.76}$$

$$\Gamma = (1-\Gamma_0)\sin^2\eta_V + \Gamma_0; \qquad \textit{Eq. 3.77}$$

The resulting potential field generated from Equation 3.76 is then used to modify the guidance commands for the aircraft in order to safely avoid collision with other agents or fixed obstacles. Morphing potential fields were originally integrated and flight test validated using a virtual leader framework and moving point guidance [82-84]. However, in this work the resultant potential field will be used to modify the reference point R in the lateral-directional frame, as well as the commanded altitude in the longitudinal frame to obtain 3D avoidance of the obstacle. This is shown below in Equation 3.78, where χ_R and γ_R represent angles from the resultant potential gradients in the lateral and longitudinal frames, respectively. The resulting components of the potential is then multiplied by the aircraft inertial speed, V_I, and a gain factor, G_{MPF}. It should be noted that as potential fields can be commutatively added, potential fields created from multiple obstacles can be combined to generate appropriate avoidance maneuvers.

$$\begin{bmatrix} R_N \\ R_E \\ h_{cmd} \end{bmatrix} = \begin{bmatrix} R_{N_0} \\ R_{E_0} \\ h_{cmd_0} \end{bmatrix} + G_{MPF}V_I \begin{bmatrix} \cos\gamma_R \cos\chi_R \\ \cos\gamma_R \sin\chi_R \\ \sin\gamma_R \end{bmatrix} \qquad \textit{Eq. 3.78}$$

3.7 MATLAB Simulink Framework

The autonomy in this work has been developed and tested in a MATLAB-Simulink environment. MATLAB is a mathematical computing software popular among engineers and scientists for its syntax simplicity, matrix manipulation capabilities, plotting functionalities, and available toolboxes for various research applications. Simulink is a MATLAB simulation framework used for model-based design. The Simulink models developed in this work involve the use of the nonlinear 6-Degree-of-Freedom aircraft equations of motion outlined in Section 3.1, as well as the dynamic models developed in AAA modeling software for the vehicles described in Section 3.2. State propagation occurs through an RK-4 integration scheme applied at a rate of 20 Hz. The autonomy developed in this work was first extensively tested and validation through simulations in the Simulink environment prior to its transition onto an embedded system for flight test validation.

As can been seen in Figure 57 and Figure 58, the Simulink block diagrams are partitioned into three major components: The Aircraft Autonomy, State Propagation, and Telemetry Emulation blocks. The Aircraft Autonomy block contains all the functions created to execute the developed autonomy on the embedded system outlined in Section 3.8. These functions are called from external libraries, which allows the developers to troubleshoot individual function issues outside of the Simulink environment. These functions take sensor information, mission objectives, telemetry information from other agents, and parameter selection information from autonomy, guidance, and control buses in order to ultimately make decisions and develop commanded outputs to the aircraft control surfaces. These control surface commands are sent to the State Propagation block, which uses them along with the simulation initial conditions to propagate the aircraft states, and emulate sensor readings for the Aircraft Autonomy block. Each aircraft in the simulation environment has its own set of Aircraft Autonomy and State Propagation blocks. A single Telemetry Emulation block is utilized in multi-agent Simulink models to collect and distribute the telemetry signals from agent to agent.

The logging iteration for state information, control commands, and autonomy decisions are set to a different rate than the 20 Hz state propagation in order to reduce the necessary memory allocations required for long duration simulations. Additionally, the computationally expensive autonomy functions for multi-agent flight line allocation and for real-time routing algorithms have been set to a separate iteration rate to emulate the implementation on the embedded system (see Section 3.8).

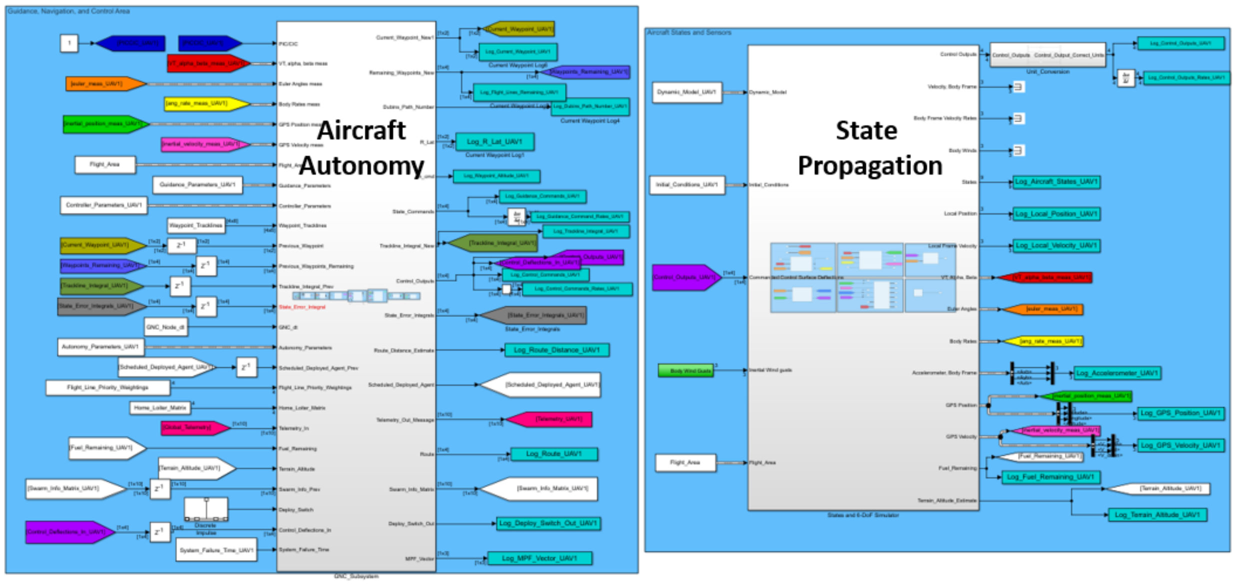


Figure 57: Single Agent Framework

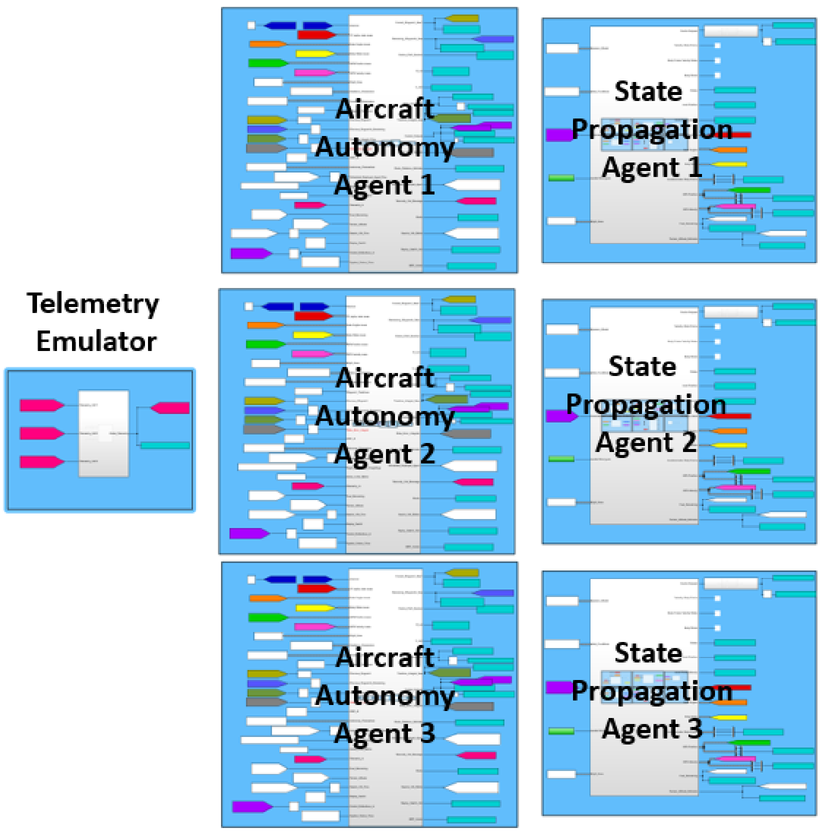


Figure 58: Multi-Agent Simulink Framework

3.8 ROS Integration

Following validation in the MATLAB-Simulink environment, the developed autonomy in this work was implemented onto a real-time system using an ROS (Robot Operating System) framework. ROS is a meta-operating system with a “system-of-systems” software architecture that incorporates numerous, unique processes, referred to as “nodes”, that execute asynchronously, parsing information through common databases, termed “topics”. Messaging protocols between nodes and topics occur through “subscribing” (reading) and “publishing” (writing) operations. ROS is an open-source robotics middleware that greatly benefits from code re-use, sensor integration compatibility, and hardware-agnostic nature, thus making it an ideal fit for research teams. The MATLAB codes developed for the autonomy in this work have been auto-coded into C++ libraries that the ROS nodes can call. In this work, ROS software is implemented onto an NVIDIA Jetson Nano unit for flight test validation of the developed autonomy onboard a SkyHunter UAS. A Pixhawk Cube unit is used as a data acquisition board for sensor readings, as well as a PWM generator for controlling the aircraft control surface actuators. The Pixhawk unit utilizes PX4 firmware, communicating to the companion computer through Mavros, and uses its Offboard functionality for executing autonomous control of the aircraft. In-flight system diagnostics are monitored through a modified Q_GCS ground station interface, through which the desired aircraft trajectory is plotted in real time. System parameter modifications and other mission commands are sent to the agents through parameters used in waypoint uploads from the ground station. The data types for various topics have been developed to be scalable sizes based on the number of flight lines, as well as the number of agents, in the survey operation. These sizing parameters can also be adjusted during operations. Finally, hardware-in-the-loop testing is conducted to validate the software and hardware integrity of the ROS package prior to flight testing operations.

Eight unique ROS nodes are created in this work in order to execute the necessary autopilot processes on a real-time platform. The names and roles of these nodes are outlined below.

1) Flight_Line_Manager
    - Reads input mission flight lines from ground station telemetry messages, populates into "Flight Lines" topic
    - Handles and updates the "Flight Lines Remaining" topic depending on "Deploy" and "Return to Home" commands from the ground station, as well as the "Flight Lines Finished" topic
2) Aircraft_States_Manager
    - Collects readings from various sensors through their respective topics and populates "Aircraft States" topic after converting values into the correct coordinate frames and units
3) Telemetry_In_Manager
    - Handles the collection of telemetric communications from other agents through respective telemetry nodes, and publishes messages to "Telemetry_In" topic
4) Swarm_Information_Manager
    - Reads in telemetry information from "Telemetry_In" topic and populates "Swarm_Information_Matrix" for flight line assignment and collision avoidance purposes
    - Creates estimates for other agent information in the event of intermittent communications
5) Autonomy_Node
    - Performs distributive real-time flight line allocation for multi-agent operations and performs TSP routing for surveying the assigned flight lines for the respective agent. The resulting order to survey the flight lines is published in the "Route" topic
6) GNC_Node

- Uses information from “Aircraft_States” and “Route” topics, the GNC_Node performs all guidance and control processes in order to output states and control surfaces commands necessary for the aircraft to track the desired trajectory

7) Telemetry_Out_Manager
   - Generates the telemetry message containing position, velocity, fuel and kinematic constraints, and mission status information that is broadcast to other agents
8) HIL_Node
   - Propagates aircraft states using an RK-4 integration scheme for hardware-in-the-loop testing, publishing emulated sensor information to the respective topics

In this framework, the core processes for intelligent surveying occur in the Autonomy_Node and GNC_Node, while all the other nodes are considered as peripherals. These peripheral nodes are responsible for simple tasks, and therefore are run at 40 Hz update rates in order to ensure information is available to the core nodes. The GNC_Node publishes control output commands at a 20 Hz rate for aircraft stability and precision tracking, while the Autonomy_Node runs at a much slower rate (1 Hz) due to the computational complexity of Hungarian Assignment and TSP heuristics. Note that the peripheral nodes are constructed in such a way that they can be modified for various aircraft, sensors, and telemetry units, while the core nodes can remain unmodified for transitioning the embedded systems across various aircraft and avionic configurations.

Chapter 4: Case Study Simulations

In this chapter, case studies will be performed on both past and future CReSIS unmanned polar research deployments in order to assess the potential benefit of the developed autonomy on operational capabilities.

4.1 Case Study – Russell Glacier 2016 Comparison

In this section, an analysis is conducted on the 2016 CReSIS UAS deployment to the Russell Glacier in order to demonstrate the improvements the developed autonomy could have on polar operations. Details about the timeline, objectives, and achievements of this deployment can be found in Section 1.2, while details about the G1X aircraft utilized in these operations can be found in Section 3.2.

4.1.1 Automated Flight Line Generation

During the 2016 Russell Glacier deployment, 8 over-the-horizon flights were conducted in which radar data was gathered along three unique areas of the outlet glacier. The first of these was an area where the collected radar imaging from the UAS could be validated by comparing with previously known bedrock topology of the area. The second area had gaps in the known bedrock topology, and the UAS radar imaging could be used to fill these gaps. Finally, the third area had little to no previously known information about the bedrock topology, and the UAS radar imaging could be used to map this area. These three areas were referred to as the "Good", "Medium", and "Bad" lines, respectively, by the operating crew to denote the level of confidence in the previously known bedrock topology from JPL WISE [25]. Radar imaging was collected along distinct flight lines over these areas during the first 7 over-the-horizon flights, while the 8th flight consisted of repeating previously flown lines to assess radar imaging consistency. Of these 7 flight operations, Flights 1-2 surveyed 6 flight lines along the "Good" area, Flights 3-4 surveyed 8 flight lines along the "Medium" area, and Flights 5-7 surveyed 10 flight lines along the "Bad" area. These flight lines are shown in Figure 59, where the “Good”, “Medium”, and “Bad” areas are the bottom, middle, and top line clusters, respectively.

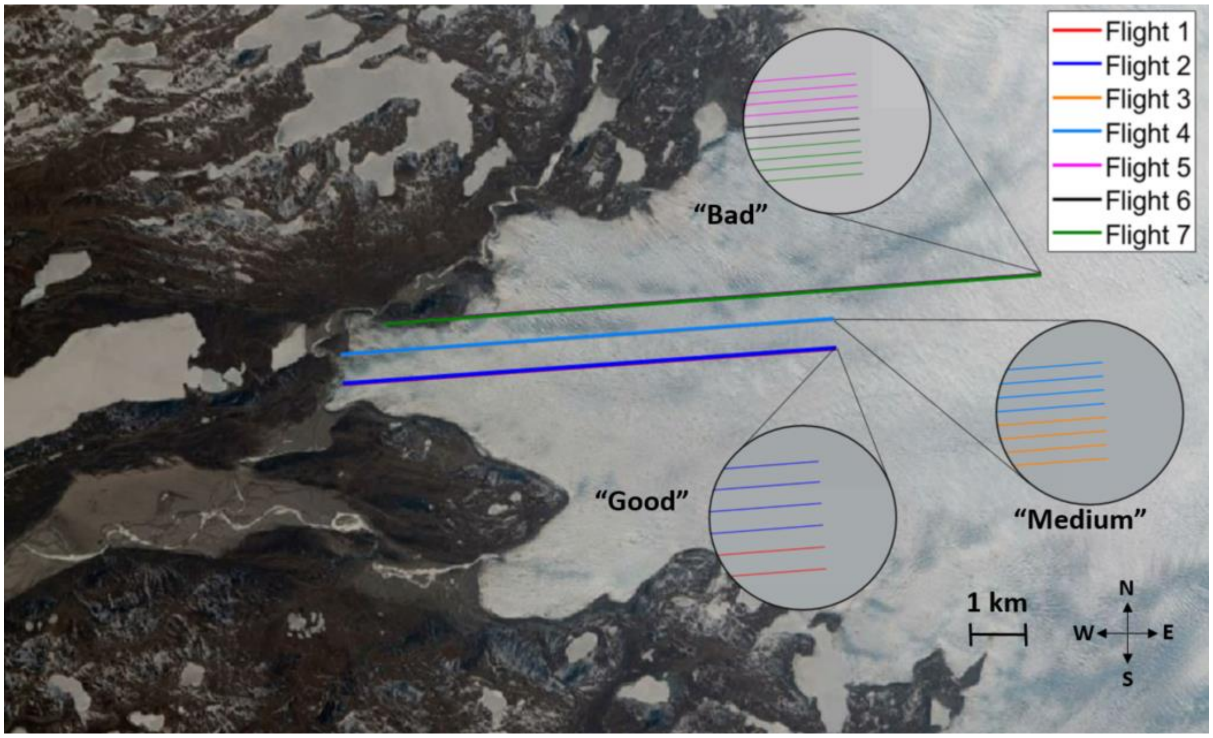


Figure 59: Russell Glacier 2016 – Mission Flight Lines - All Flights

Waypoint information for these flights were chosen by CReSIS and sent to the operating team as KML files, which consist of geographic coordinates in degree-minute-second format. These waypoints then had to be individually converted into decimal format and manually input into the flight controller’s ground station by the operating crew prior to each flight. By analyzing the waypoints from the aircraft log data, the centroids for these flight lines in each area were reconstructed, and are listed below in Table 4. Note that the average spacing between the flight lines slightly varies from the desired half-wavelength and quarter-wavelength spacing for the flight areas, and that the wavelength used in this analysis corresponds to the 35 MHz operating frequency of the radar system utilized in this deployment. Possible reason for these discrepancies in spacing include human error during waypoint positioning in the KML file, conversion errors from DMS to decimal formats, and human error during manual inputting the waypoints into the ground station. Additional effort was applied by the operating crew to ensure that the waypoint manually input were logistically correct, namely that none of the waypoint longitudes were positive (leading the aircraft to the Western hemisphere), and that the waypoint altitudes were suitable for the terrain (to avoid crashing into the surface of the glacier). These two factors were very important due to the fact that the aircraft would operate outside of the range of the telemetry units and that there was no feedback to the ground station for terrain clearance.

Table 4: Russell Glacier Flight Line Information

Flight Area	Good Lines	Medium Lines	Bad Lines
Distance (km)	8.89 km	8.89 km	11.86 km
Latitude 1 (degrees N)	67.0919855	67.0965979	67.1015807
Longitude 2 (degrees E)	-50.2327605	-50.2337748	-50.2159535
Altitude 1 (m AMSL)	620	630	630
Latitude 2 (degrees N)	67.0976477	67.102262	67.1093212
Longitude 2 (degrees E)	-50.0281193	-50.0290914	-49.9430035
Altitude 2 (m AMSL)	823	823	823
Number of Lines	6	8	10
Average Spacing as fraction of Wavelength	0.5792	0.2336	0.2333

For these reasons, an automated way of precisely generating flight lines along these centroids was developed for these operations and is outlined in Section 2.1. Using the information from Table 4 on centroid locations, desired spacing, and number of lines, the automated flight line generation algorithms were able to quickly recreate these flight lines with the appropriate spacing. The resulting lines from the automated flight line generation is shown in Figure 60.

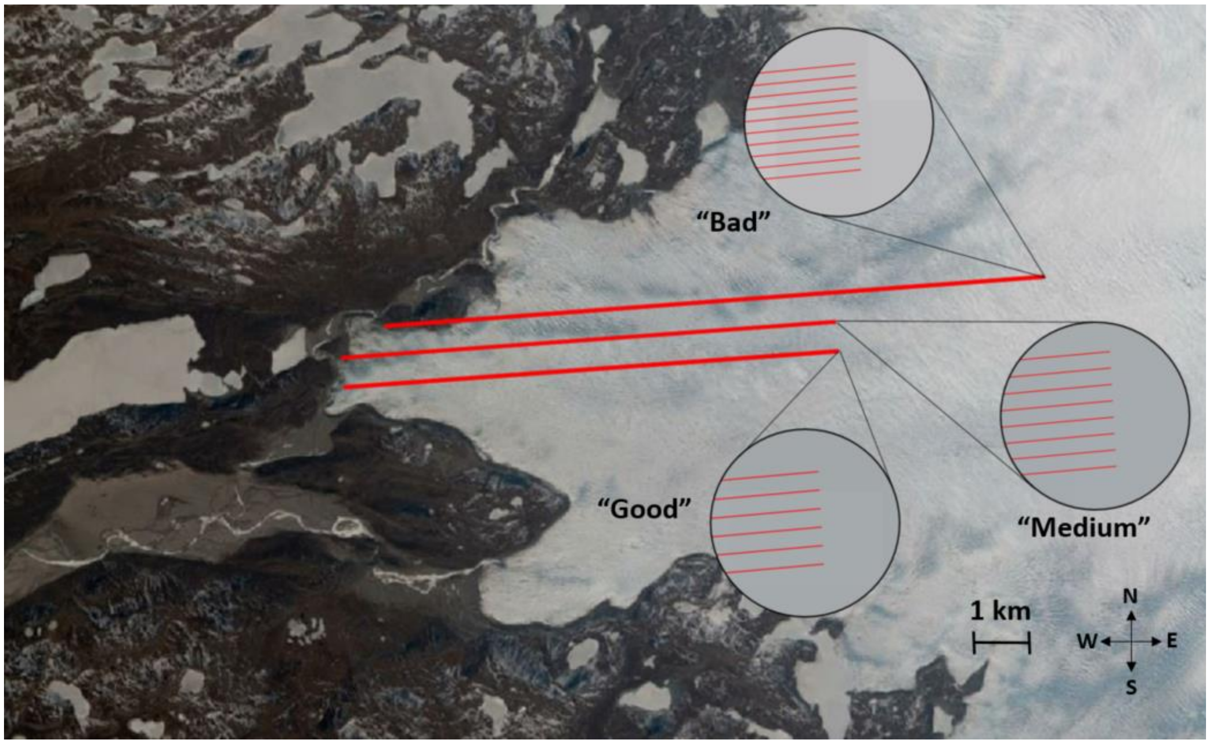


Figure 60: Russell Glacier 2016 - All Lines Automated Generation

4.1.2 Lateral Tracking Comparison

The tracking performance of the aircraft for the survey flight lines was assessed using data from Flight 3 of the 2016 Russell Glacier deployment in order to determine the potential benefits of the developed autonomy. The survey flight line objectives in Flight 3 involved four tightly-spaced flight lines in the "Medium" area, and a home loiter circle where the operating crew was located. This mission scenario is shown in Figure 61.

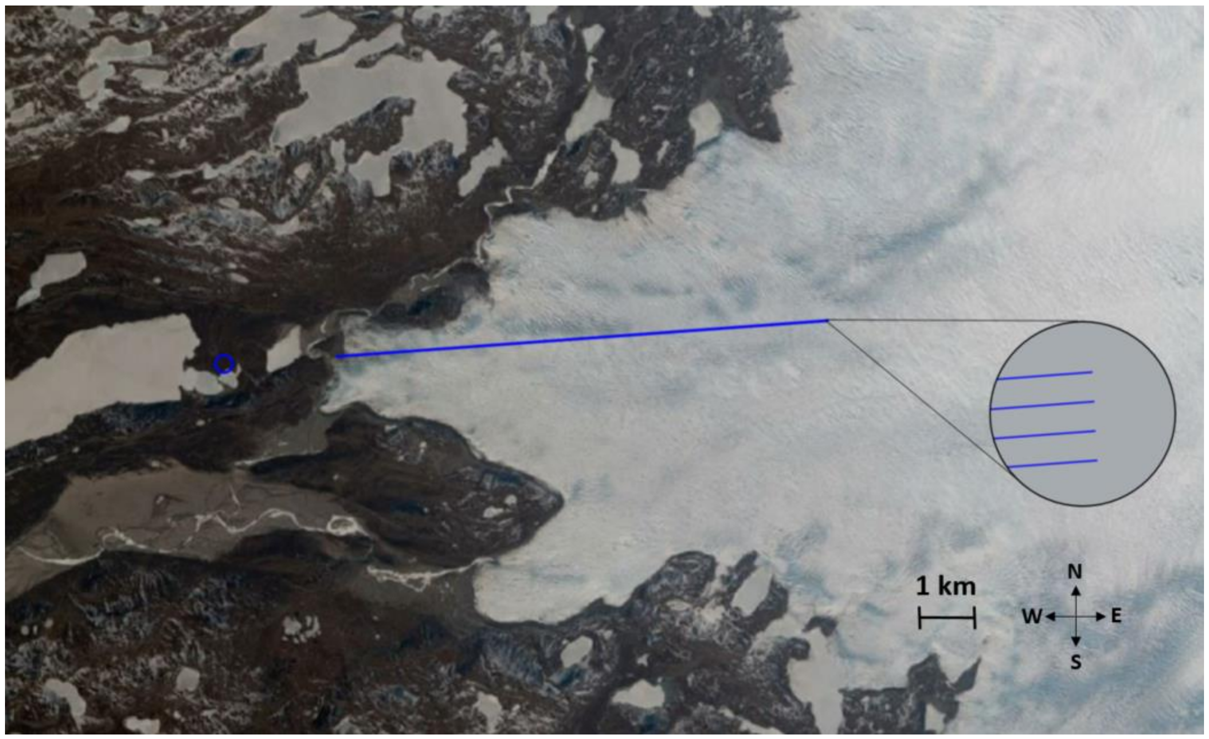


Figure 61: Flight 3 Mission Scenario

Using the flight data collected from the aircraft flight controller, the tracking performance of the system in terms of the end-to-end coverage of each of the survey flight lines was analyzed. The aircraft tracking for the commercial off the shelf (COTS) autopilot system used in the 2016 deployment is shown in Figure 62. Note that the standard waypoint tracking methods and guidance logic for general UAS applications used are not suitable for the tightly spaced flight lines, as the aircraft misses a significant portion of the beginning of each flight line following each turning maneuver. Additionally, a significant steady-state off-track positioning error occurs along the stretch of the flight line, as the aircraft never truly converges onto the desired path (possibly due to crosswind). The end-to-end coverage of each flight line was assessed in terms of distance from the desired survey line (Figure 63), heading error from the flight line heading (Figure 64), and the bank angle of the aircraft (Figure 65). Note that the data for this analysis is only collected between the two endpoints of each flight line (i.e. does not include turning maneuvers until the aircraft crosses the beginning of the flight line), and vertical partitions have been used to indicate the flight line number. Note that for ideal radar imaging, the aircraft would be directly over the desired survey line, flying with the proper heading, with low bank angles to maximize the directivity of the antenna in the nadir direction.

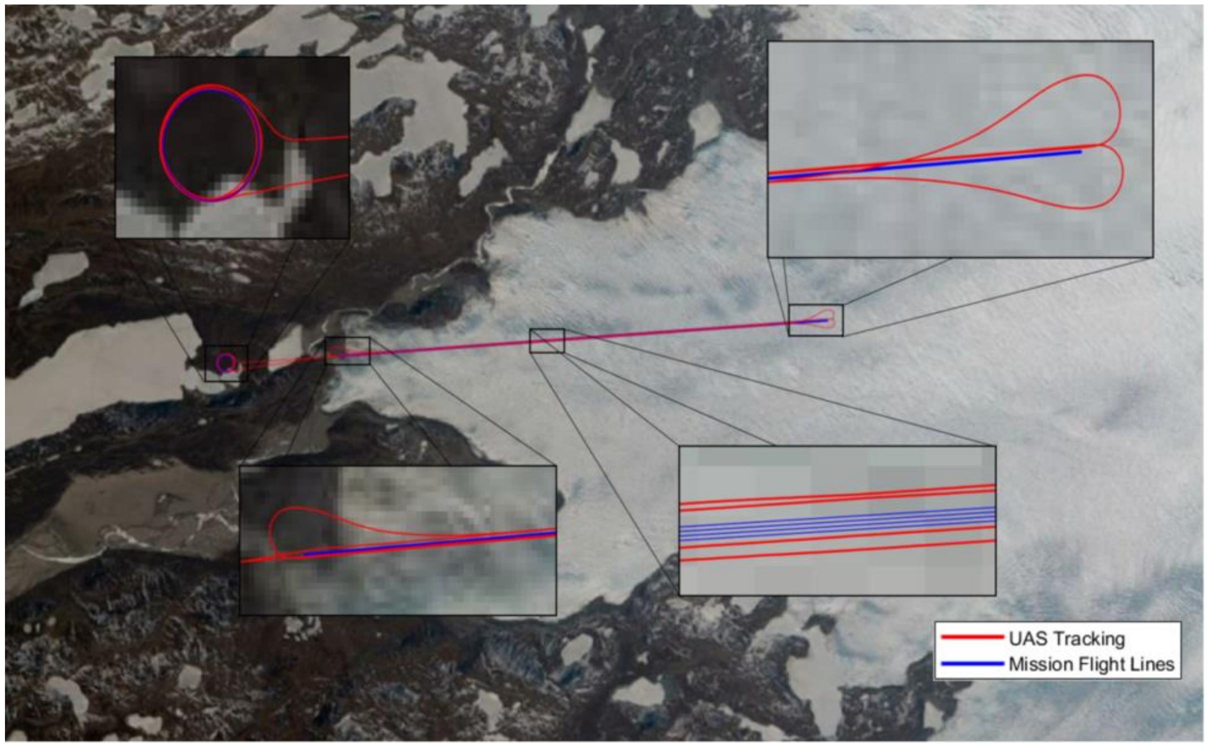


Figure 62: Flight 3 - COTS Tracking Analysis

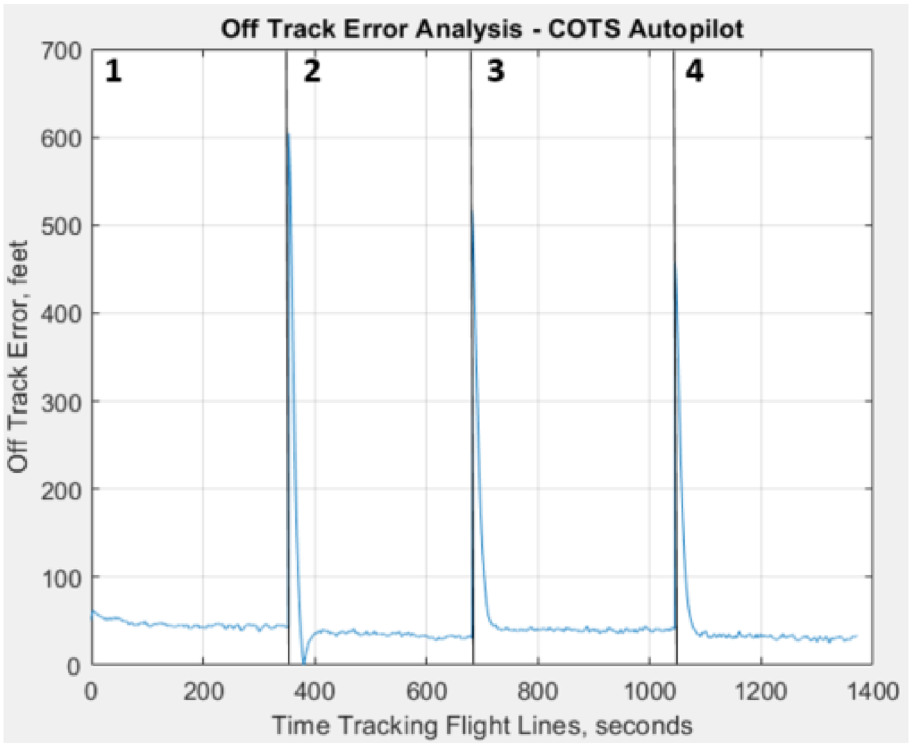


Figure 63: Distance from Flight Line – COTS Autopilot

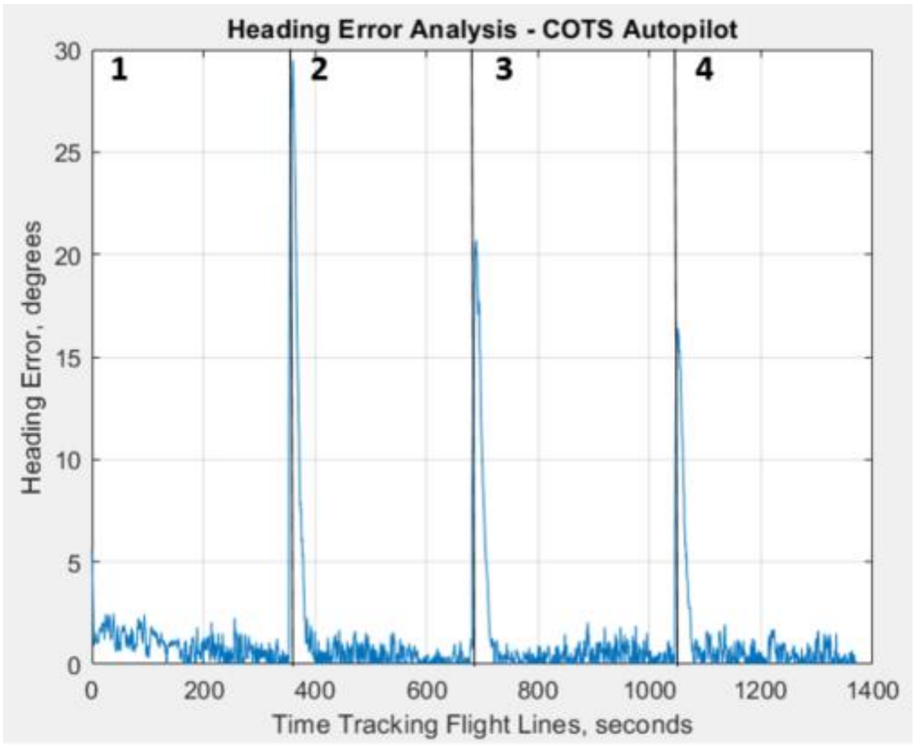


Figure 64: Heading Error Analysis – COTS Autopilot

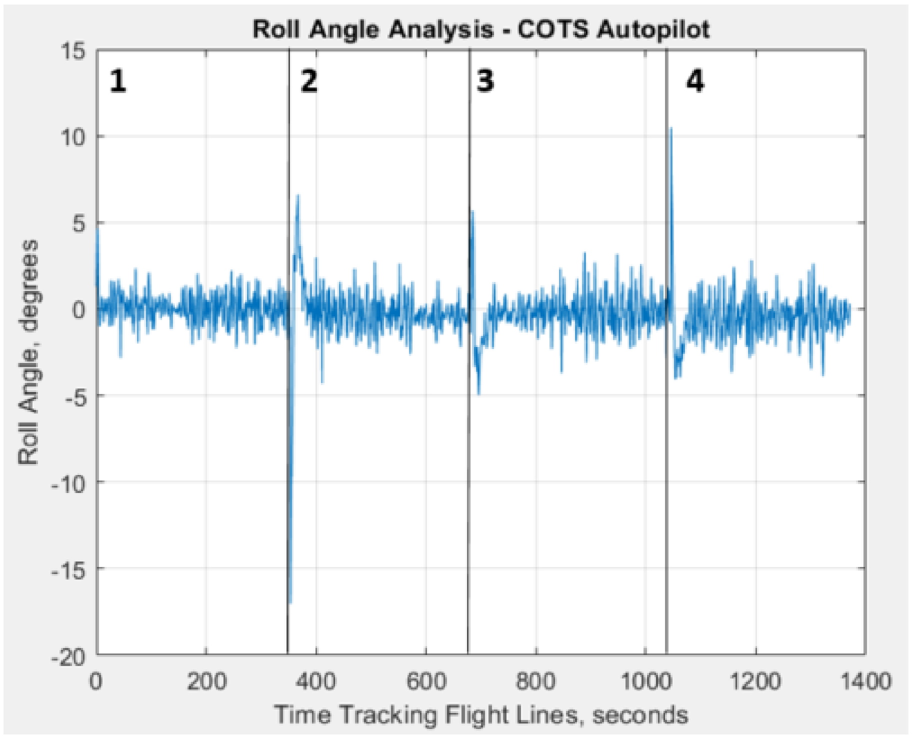


Figure 65: Roll Angle Analysis – COTS Autopilot

In the previous figures, it can be observed that the COTS autopilot has a significant off-track error at the beginning of each flight line, following its turning maneuver. This is a result of its traditional guidance methods for tracking lines between the waypoints constructing the mission flight lines. These turning maneuvers also introduce large heading errors and roll angles, which are not desirable for the along-track synthetic aperture imaging.

Using the 6-DoF equations of motion outlined in Section 3.1, the G1X dynamic model outlined in Section 3.2, and the developed MATLAB Simulink environment outlined in Section 3.7, the performance of the developed autonomy solution is evaluated for this mission scenario. The simulated tracking performance of the aircraft is shown in Figure 66, where it can be noted that the Dubins Path guidance methods outlined in Sections 2.2 and 3.3 enable the aircraft to utilize turning maneuvers that position the aircraft at the beginning of the flight line (Figure 67) with the proper heading (Figure 68), as well as a low bank angle (Figure 69). Additionally, the L2+ guidance method for flight line following allows the aircraft to fully converge onto the desired path and eliminate steady-state off-track positioning errors.

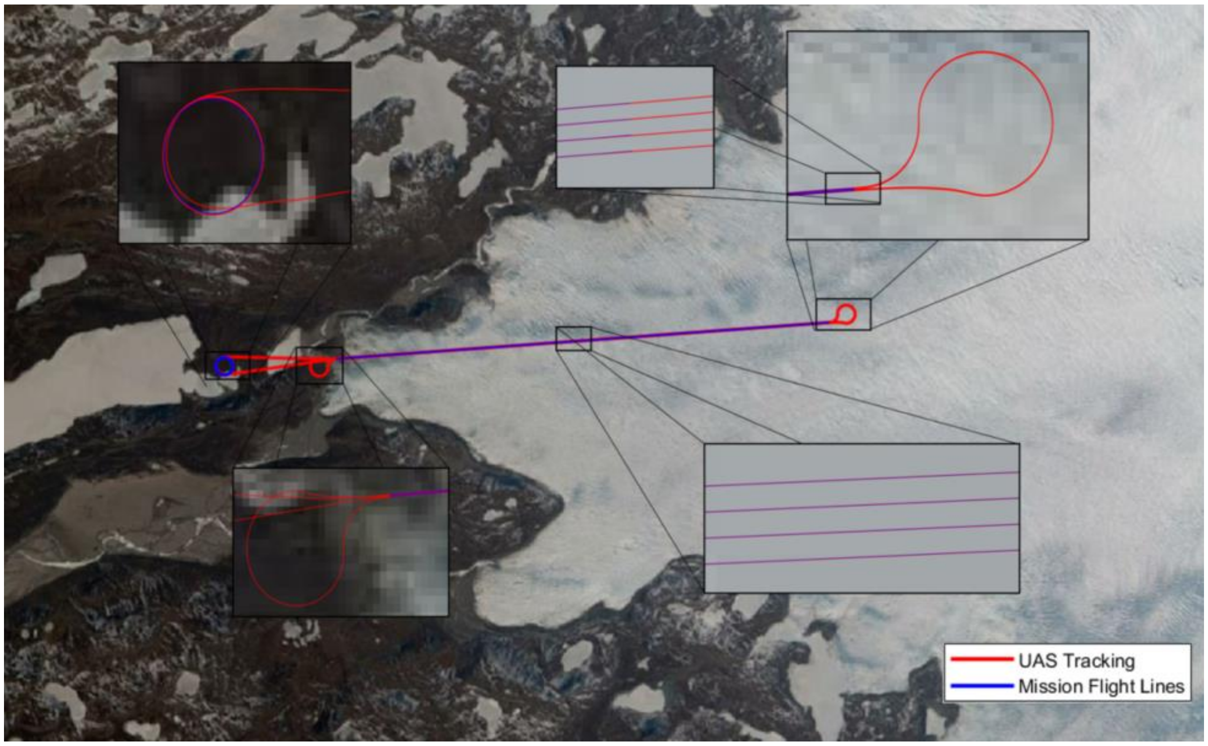


Figure 66: Simulation Tracking for Developed Autonomy

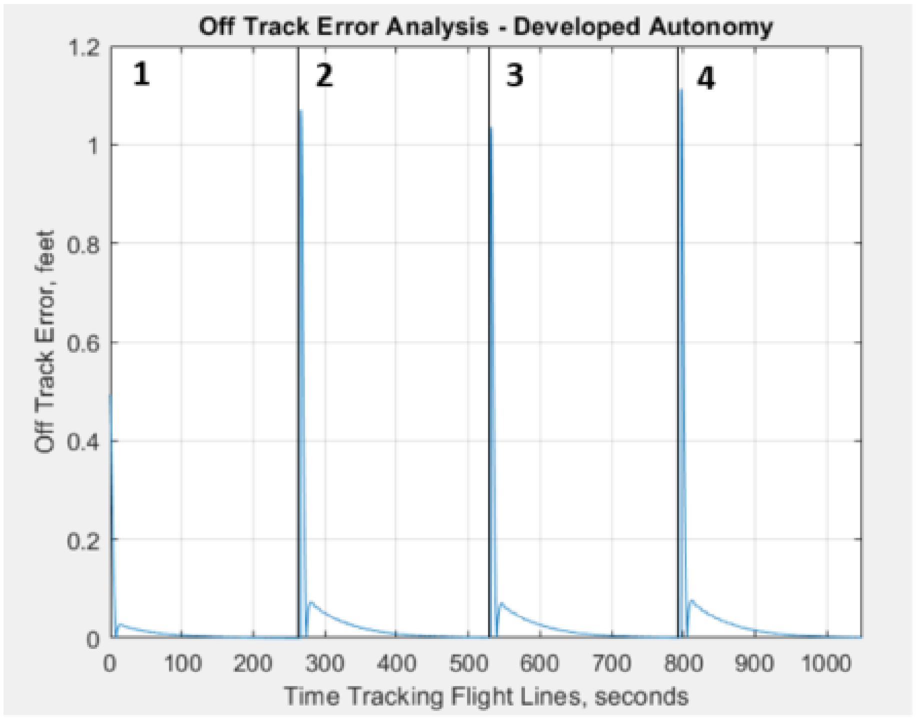


Figure 67: Off-Track Error Analysis - Developed Autonomy

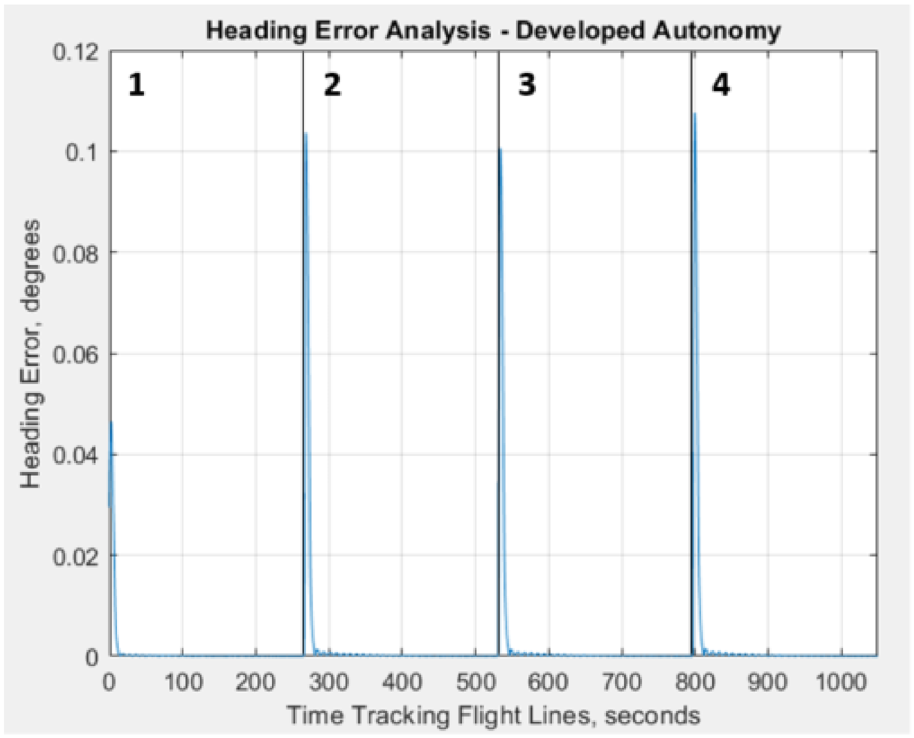


Figure 68: Heading Error Analysis - Developed Autonomy

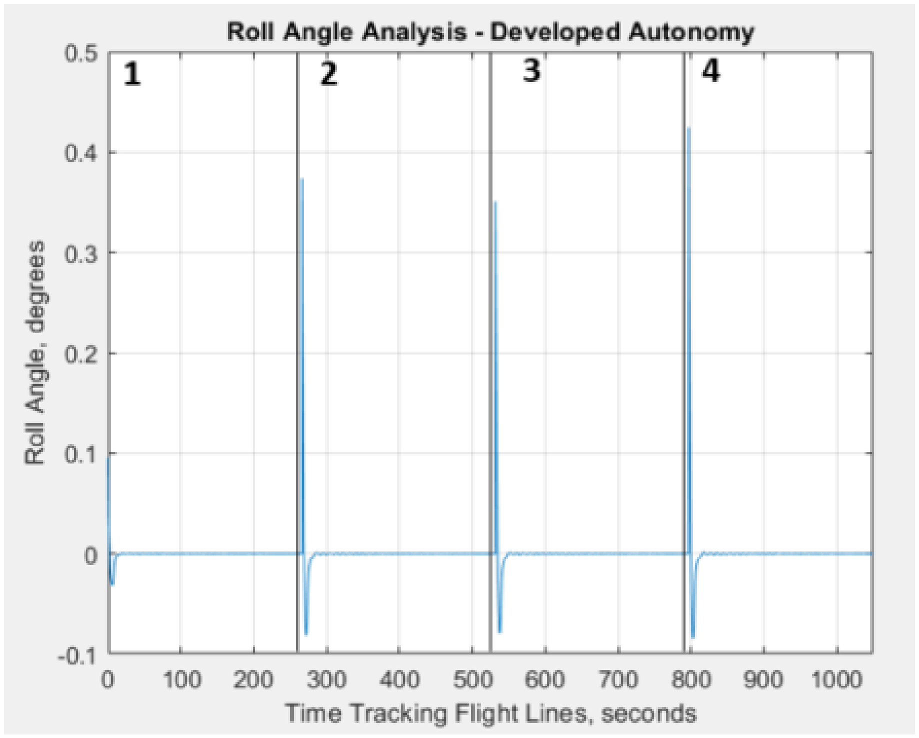


Figure 69: Roll Angle Analysis - Developed Autonomy

The performance of the two systems are compared in Table 5 in terms of their end-to-end coverage of the mission flight lines. Note that the criteria for convergence onto the flight lines is based on the rate of decrease of the off-track position error. In this scenario, the Dubins Path guidance methods were able to effectively eliminate the tracking errors following the required turning maneuvers, capturing full end-to-end coverage of the survey flight lines without significant off-track error, heading errors, or large roll angles. However, note that while this is an unfair tracking comparison between an actual flight and simulation data, the path planning methods for the developed autonomy are constructed in such a way as to more fully capture the flight line, while standard waypoint tracking methods on COTS systems would result in similar tracking performance as shown in the flight data for this case study. While the tracking performance of the developed autonomy would vary in actual flight due to external disturbances such as wind and gusts, the Dubins Path methods should still outperform standard methods if the flight controller is able to track the desired path to a satisfactory degree. Note that the integral term in the line tracking guidance (Section 3.2.2) would reduce the effect of steady-state off-track error in the event of steady-state wind conditions.

Table 5: Russell Glacier Tracking Comparison

Autopilot System	COTS Autopilot	Developed Autonomy
Max off-track error from Flight Line	604 feet	1.11 feet
Average Distance to Converge	2,688 feet	0 feet
RMS Off-Track Error	78 feet	0.1 feet
Max roll angle during line tracking	-17.02 degrees	0.42 degree
Max heading error during line tracking	29.5 degrees	0.11 degrees

4.2 Case Study: Single-Agent Helheim Glacier

In preparation for upcoming deployments to the Helheim Glacier by the CReSIS UAS team, the developed autonomy is tested in order to demonstrate its effectiveness for a mission planning tool as well as cognitive real-time path planning. The aircraft chosen by the CReSIS UAS team is a Mugin-2930, a VTOL platform detailed in Section 3.2. For incorporating the range constraints of the aircraft into the TSP analysis conducted in this section, the cruise velocity of 48 knots (81 feet/second) and the estimated aircraft endurance of 90 minutes results in a range restriction of 437,400 feet (~82 miles). Note that this range is disregarding fuel consumed during the climbing and descent phases, and should be reevaluated through preliminary flights at the site. A satellite view of the Helheim Glacier is shown in Figure 70, where the 5.5 km width of the main channel can be noted, along with the planned ground station area location on the ridge on the south side of this main channel. A view from this takeoff/landing area is shown in Figure 71, which was taken during a recent scouting trip by the CReSIS UAS team.

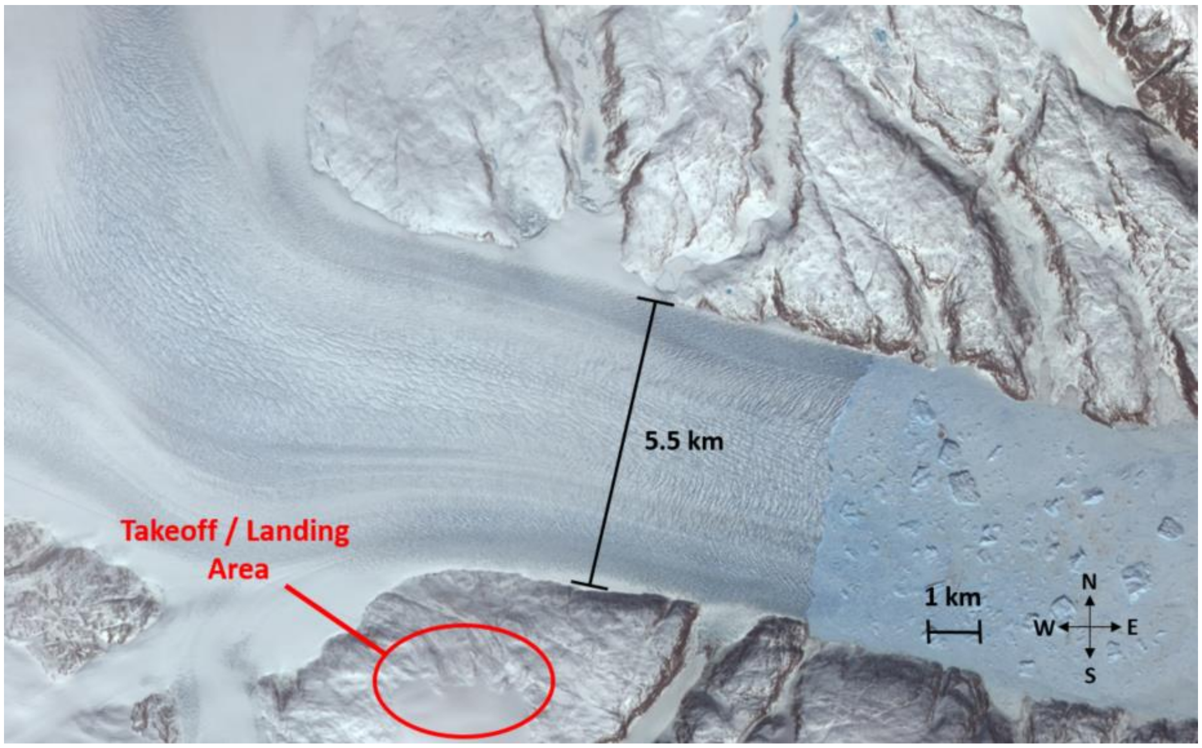


Figure 70: Helheim Glacier, Satellite View

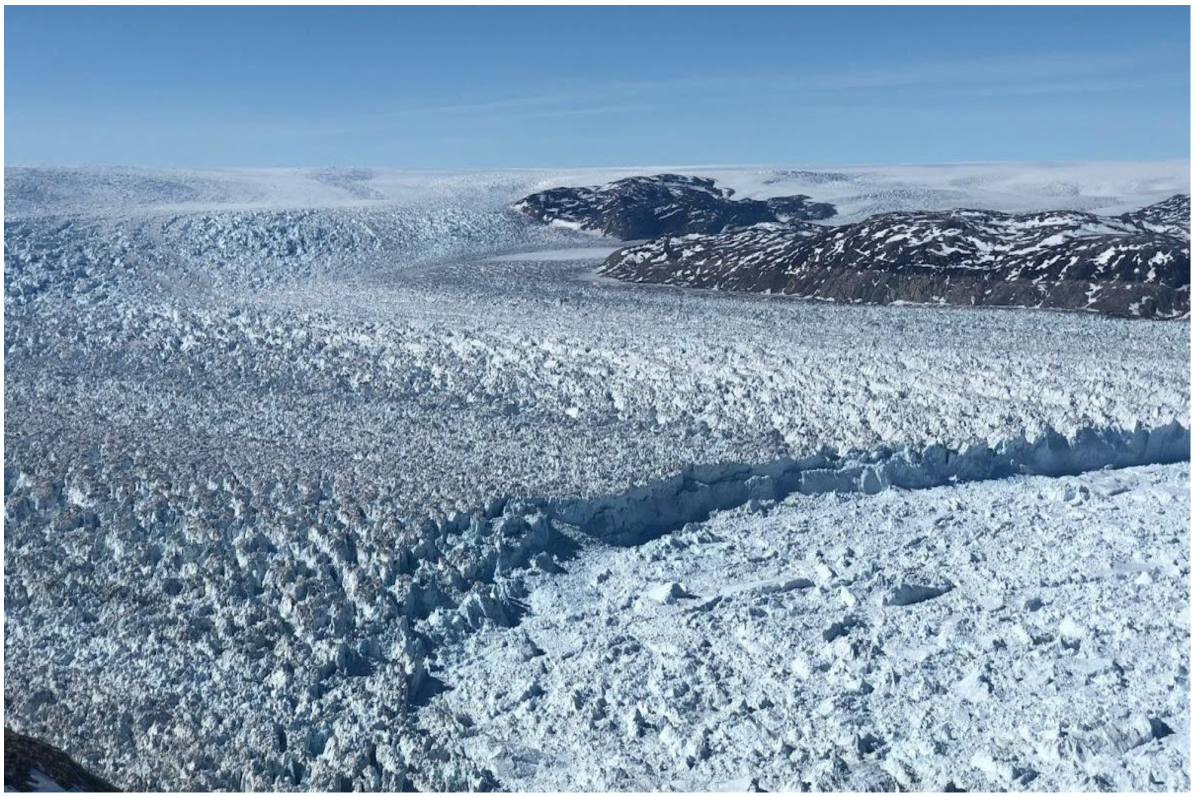


Figure 71: View from Helheim Ground Station Area (Photo Credit: KU Flight Research Lab)

In this section, the various TSP methods outlined in Sections 2.3 and 2.4 will be compared to assess their suitability for each mission scenario by analyzing the tradeoff in performance with computational runtime for the algorithm. Note that for these comparisons, Ant Colony Optimization was performed using 1000 iterations of 10 ants per generation, with pheromone reinforcement and decay rate factors of 0.1 and 0.001, respectively. Additionally, as it is challenging to quantify how a human operator would design a route through the mission flight lines, an assumption is made that the resulting route would be similar to the forward greedy method, for which the performance of the other heuristics will be compared against.

4.2.1 Broad Grid Survey

The first scenario for testing the developed TSP algorithms involves a broad grid survey operation involving 3 along-flow (East-West) flight lines and 7 cross-flow (North-South) flight lines, with respect to the outlet glacier. These flight lines are equally spaced at a distance of 1 km, and have been devised to broadly capture the along-flow and cross-flow bedrock topology of the outlet glacier. Mission operations involve the deployment of the aircraft from an area near the home loiter circle, where the operating crew and ground station will be located. Using the various TSP methods outlined in Section 2.3, different routes through the mission flight lines are developed. These routes are shown in Figure 72. The resulting lengths of the routes are shown in Table 6, where the transit distance does not include the lengths of the flight lines themselves, only the lengths onto, between, and returning from the flight lines. Also denoted is the percentage decrease of the transit route lengths compared to the forward greedy result. The computational runtime of each algorithm is also given to show the tradeoff of performance and computational cost between the various TSP methods. Note that all computational runtime analysis in this section was conducted on the same computer running the respective MATLAB scripts for the TSP algorithms.

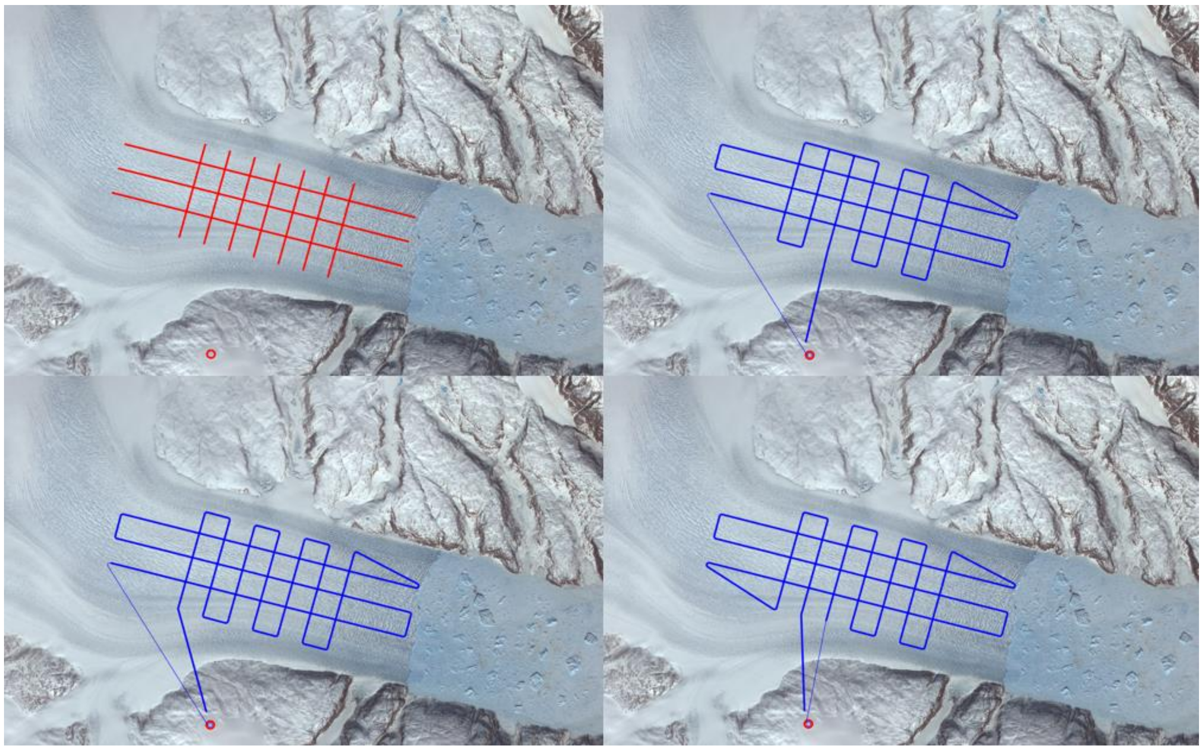


Figure 72: Scenario 1, Broad Grid Survey (Top Left: Mission Flight Lines; Top Right: Forward Greedy; Bottom Left: Global Greedy; Bottom Right: Ant Colony Optimization)

Table 6: Scenario 1 TSP Results

TSP Heuristic	Total Distance	Total Transit Distance	% Improvement Transit Distance from Forward Greedy	Computational Runtime
Forward Greedy	287,087 feet	82,777 feet	0%	0.194 seconds
Global Greedy	282,409 feet	78,099 feet	5.65%	0.375 seconds
Ant Colony Optimization	277,451 feet	73,141 feet	11.64%	5.689 seconds

The results from Scenario 1 indicate that the ACO route was able to find the best route through the mission flight lines, but took significantly more time to run. Note that the Forward Greedy route backtracks through the cross-flow lines in an inefficient way, while the Global Greedy route first completes all the cross-flow lines before transitioning to the along-flow lines. Conversely, the route produced by Ant Colony Optimization transitions back and forth between cross-flow and along-flow lines, resulting in a shorter overall route. The Ant Colony Optimization was able to find this result due to its stochastic nature, preventing it from being trapped in local optima during graph exploration.

4.2.2 Fine Grid Survey

In the second scenario, the various TSP methods will be compared for a fine grid survey operation, in which 7 tightly-spaced along-flow (East-West) lines are oriented perpendicular to 5 tightly-spaced cross-flow (North-South) lines. The tight spacing of approximately 7 feet between the flight lines is equal to the quarter-wavelength of the 35 MHz operating frequency of the planned radar system for the Helheim deployment. The resulting routes from the various TSP methods are shown in Figure 73, while their respective lengths and computational runtimes are shown in Table 7.

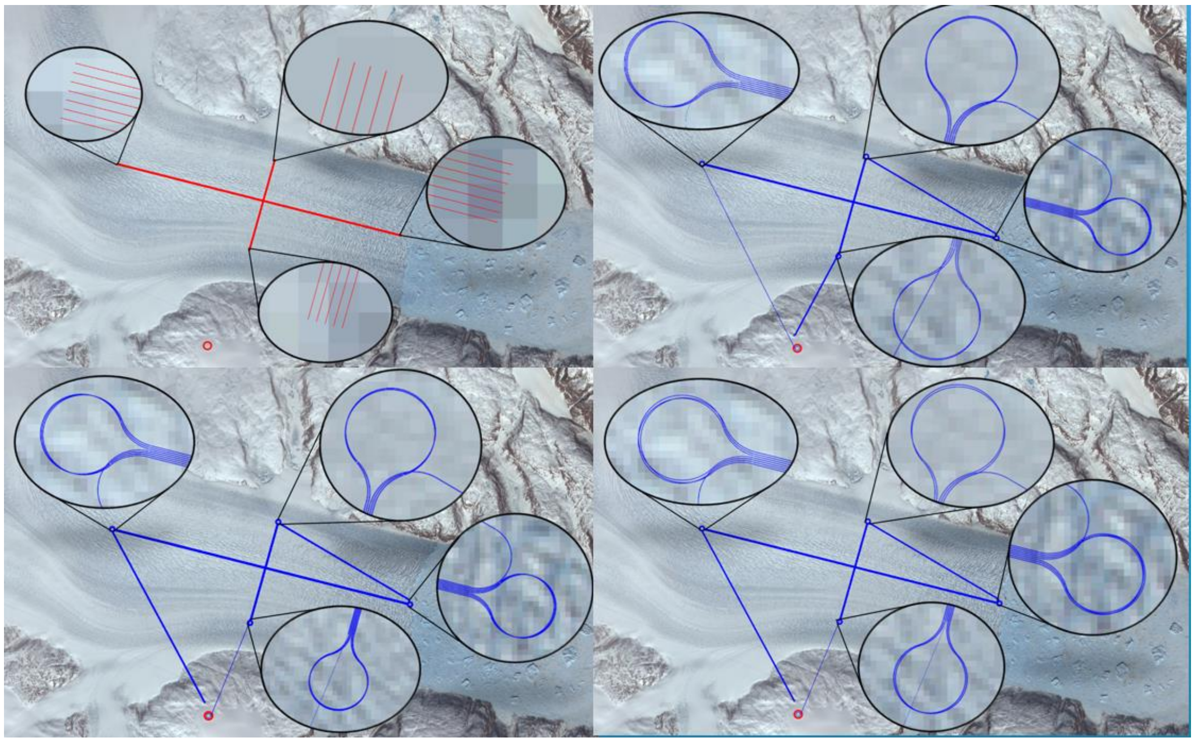


Figure 73: Scenario 2, Fine Grid Survey (Top Left: Mission Flight Lines; Top Right: Forward Greedy; Bottom Left: Global Greedy; Bottom Right: Ant Colony Optimization)

Table 7: Scenario 2 TSP Results

TSP Heuristic	Total Distance	Total Transit Distance	% Improvement Transit Distance from Forward Greedy	Computational Runtime
Forward Greedy	413,383 feet	81,123 feet	0%	0.203 seconds
Global Greedy	412,958 feet	80,698 feet	0.52%	0.728 seconds
Ant Colony Optimization	412,900 feet	80,640 feet	0.59%	8.746 seconds

In Scenario 2, the Ant Colony Optimization method was again able to find a shorter route through the lines than the greedy methods, while taking significantly more computational time. However, the difference between the lengths of the resulting routes is minimal. Though it may be hard to see in Figure 73, the main difference between the routes is that the Forward Greedy route heads to the cross-flow lines first, while the Global Greedy and Ant Colony Optimization routes head to the along-flow lines first. Between the latter two routes, the main differences exist in how the aircraft surveys the flight lines in each cluster, and how the resulting Dubins Path lengths between those ordered flight lines sum together. Overall, Scenario 2 is developed to demonstrate that the benefit of the various TSP methods decreases with the complexity of the mission flight lines (i.e. the complexity of the TSP graph developed by the flight lines in the mission). In this mission scenario, the Forward Greedy method could be used for real-time processing onboard the aircraft because the additional computational complexity of the Ant Colony Optimization isn't worth the tradeoff for a minimal reduction in overall route length.

4.2.3 Comprehensive Survey

In Scenario 3, a comprehensive survey is created by placing 15 flight lines along and across various flow lines on the glacier inlet and main channel in order to capture complex dynamics at these intersections. The various TSP methods were utilized for this scenario in the absence of the aircraft range constraints, and the resulting routes are shown in Figure 74. The respective lengths and computational runtimes are shown in Table 8.

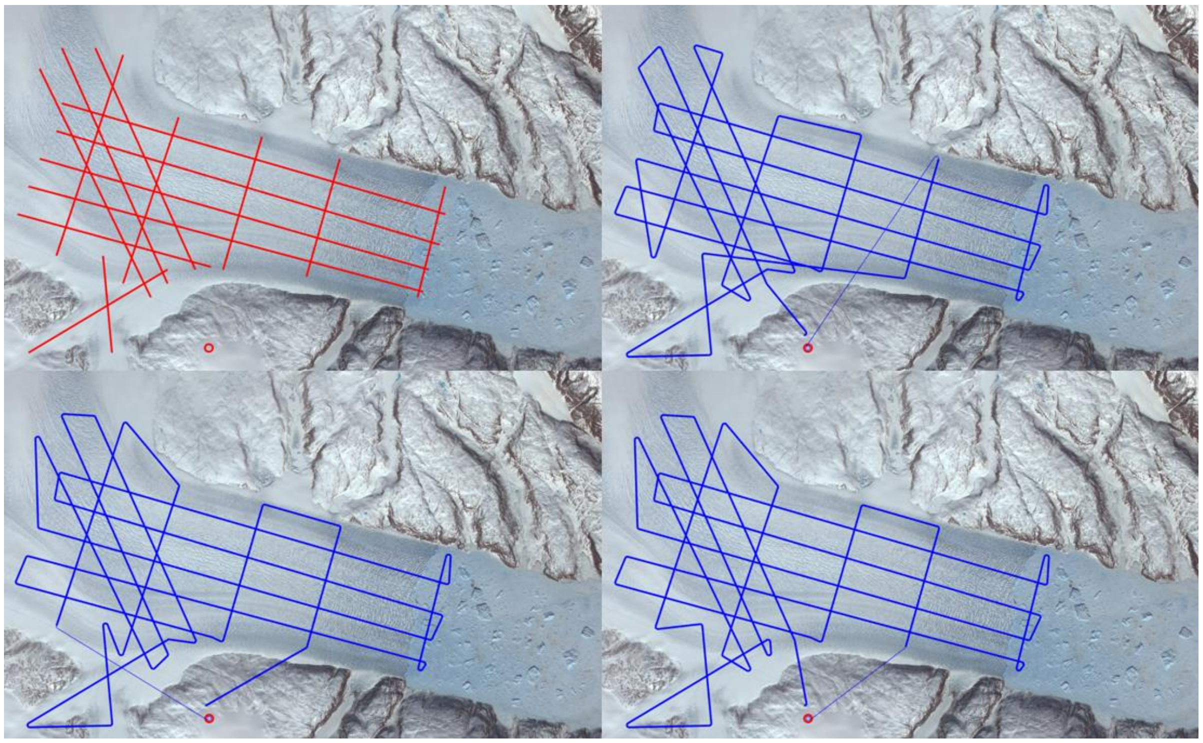


Figure 74: Scenario 3, Comprehensive Survey, No Fuel Constraints (Top Left: Mission Flight Lines; Top Right: Forward Greedy; Bottom Left: Global Greedy; Bottom Right: Ant Colony Optimization)

Table 8: Scenario 3 Results, No Fuel Constraints

TSP Heuristic	Total Distance	Total Transit Distance	% Improvement Transit Distance from Forward Greedy	Computational Runtime
Forward Greedy	606,147 feet	140,617 feet	0%	0.202 seconds
Global Greedy	587,882 feet	122,352 feet	12.99%	0.637 seconds
Ant Colony Optimization	575,545 feet	110,015 feet	21.76%	7.73 seconds

Due to the complex nature of the TSP graph created by the flight lines in this scenario, the Ant Colony Optimization method was able to produce a much shorter route that the greedy methods, although the differences in computational runtimes is similar to the previous scenarios. For this reason, the Ant Colony Optimization would be a more suitable choice for real-time route planning, as the added computational complexity will significantly improve mission efficiency. However, note that each TSP solutions produced a route longer than the aircraft's range constraint of 437,400 feet, and therefore it is not possible to survey all of the flight lines in a single flight operation. Scenarios 4 and 5 will assess how the fuel-constrained modifications to the TSP algorithms can be used to optimize the possible survey potential of the aircraft given the mission flight lines and fuel constraints.

4.2.4 Comprehensive Survey with Fuel Constraints and Uniform Weightings

In Scenario 4, the comprehensive survey flight lines are subjected to the fuel-constrained TSP algorithms outlined in Section 2.4, where each flight line in the mission has a uniform weighting of 1 utility. Note that this effectively reduces the problem statement to determining the maximum number of flight lines that can be practically surveyed given the fuel constraints. The resulting routes from the fuel-constrained TSP methods are shown in Figure 75, and the resulting route lengths, utility, and computational runtimes are shown in Table 9.

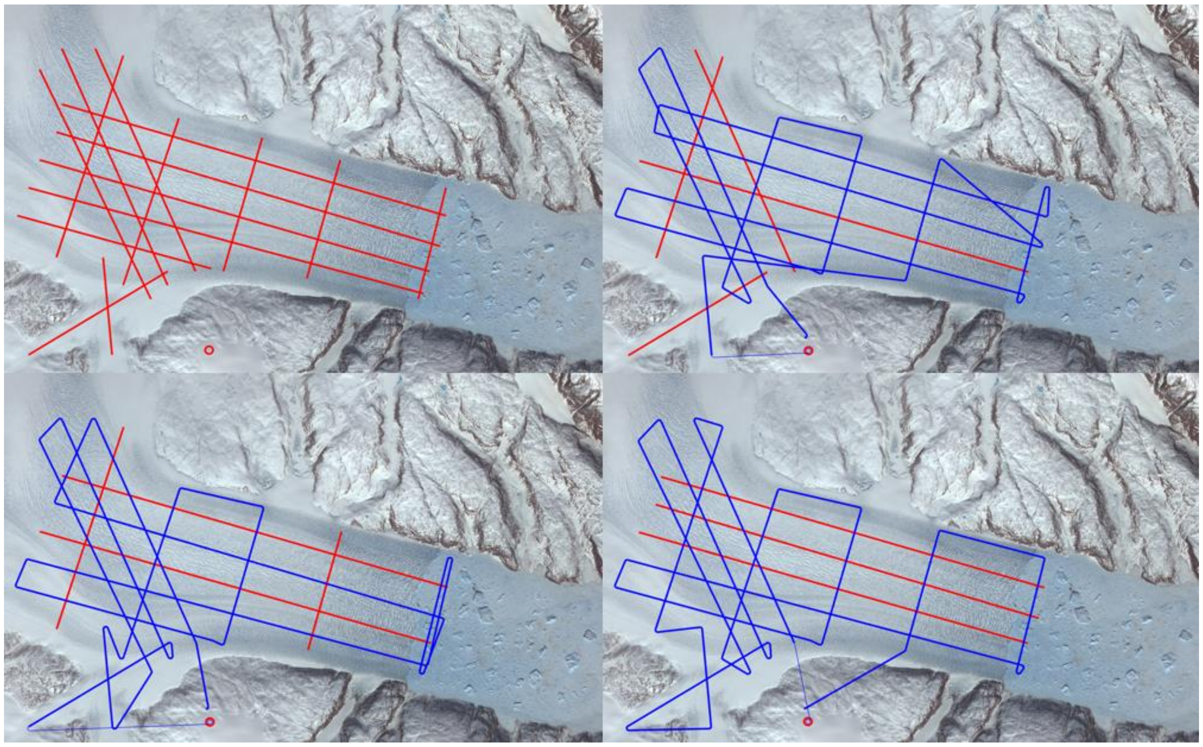


Figure 75: Scenario 3 With Fuel Constraints and Uniform Weightings (Top Left: Mission Flight Lines; Top Right: Forward Greedy; Bottom Left: Global Greedy; Bottom Right: Ant Colony Optimization)

Table 9: Scenario 3 Results With Fuel Constraints and Uniform Weightings

TSP Heuristic	Total Utility	Total Distance	<u>% Improvement Total Distance from Forward Greedy</u>	Computational Runtime
Forward Greedy	11	436,017 feet	0%	0.116 seconds
Global Greedy	11	427,214 feet	2.02%	1.504 seconds
Ant Colony Optimization	12	404,463 feet	7.24%	3.957 seconds

In this scenario, while the fuel-constrained Forward Greedy and the Global Greedy methods both were able to survey 11 flight lines, the fuel-constrained Global Greedy was able to do so in a shorter route. However, the Fuel-Constrained Ant Colony Optimization method was able to survey 12 flight lines, in an even shorter route. Note that this method accomplishes this by surveying the shorter flight lines, while the greedy methods survey more of the longer flight lines in the mission, due to their nature of being trapped in local optima. As each flight lines have a uniform weighting, the fuel-constrained Ant Colony Optimization method has found a more optimal solution through its stochastic searching, capturing a higher net utility while adhering to the aircraft range constraints.

4.2.5 Comprehensive Survey w/ Fuel Constraints, Non-Uniform Weightings

In Scenario 5, the fuel-constrained TSP methods will be assessed for a scenario where the 5 along-flow lines on the main channel of the glacier are highly prioritized, with a utility weighting of 100. The other flight lines in the mission retain their utility weighting of 1. The resulting routes from the fuel-constrained TSP methods are shown below in Figure 76, and their respective lengths, utilities, and computational runtimes are shown in Table 10. Note that the flight lines with higher utility weightings are marked in black in Figure 76.

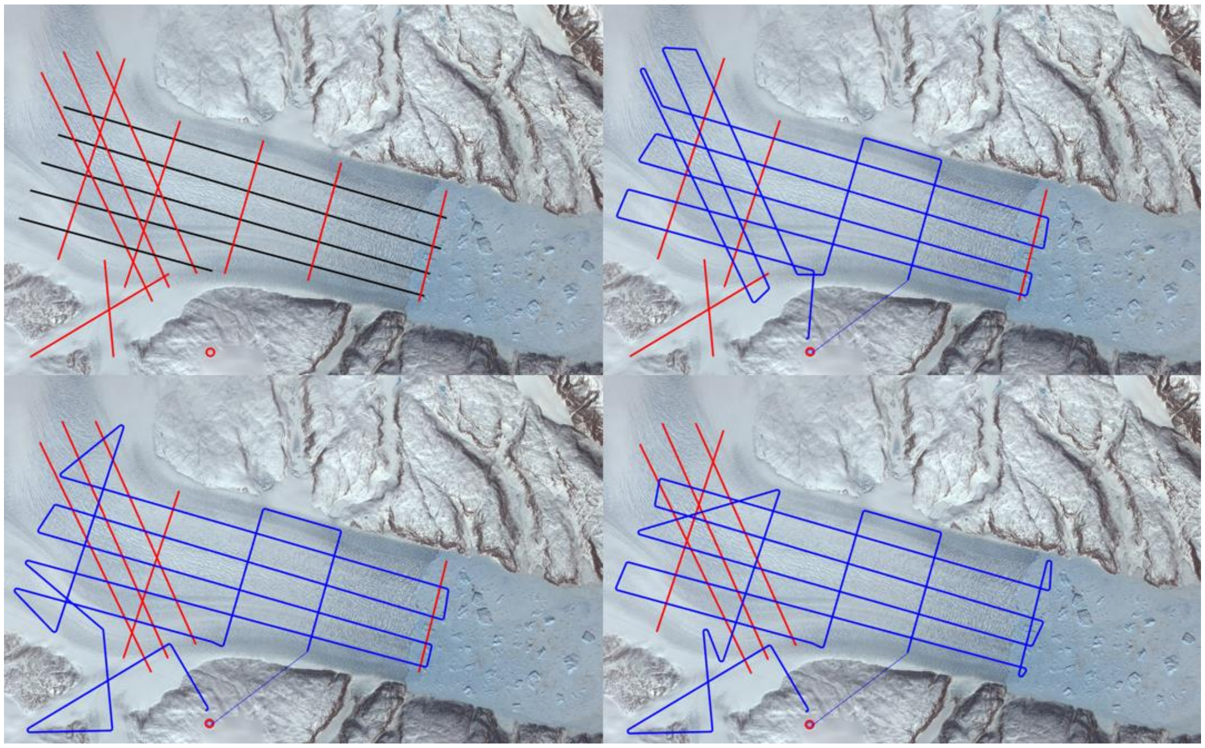


Figure 76: Scenario 3, With Fuel Constraints and Non-Uniform Weightings (Top Left: Mission Flight Lines; Top Right: Forward Greedy; Bottom Left: Global Greedy; Bottom Right: Ant Colony Optimization)

Table 10: Scenario 3 Results with Fuel Constraints and Non-Uniform Weightings

TSP Heuristic	Total Utility	Total Distance	% Improvement Total Distance from Forward Greedy	Computational Runtime
Forward Greedy	505	435,954 feet	0%	0.112 seconds
Global Greedy	505	423,978 feet	2.75%	1.412 seconds
Ant Colony Optimization	506	432,700 feet	0.75%	3.668 seconds

In this scenario, the Fuel-Constrained Forward Greedy and the Fuel-Constrained Global Greedy were able to capture a utility of 505 from the mission scenario (i.e. all 5 of the higher utility weighting flight lines as well as 5 of the other flight lines), but the latter was able to do so in a shorter route. The Ant Colony Optimization method was again able to outperform the greedy methods in terms of utility due to the greedy methods' susceptibility for being trapped in local optima, which effectively constrains their graph exploration immediately onto the higher weighted flight lines. Although the Fuel-Constrained Ant Colony Optimization produced a route longer than the Fuel-Constrained Global Greedy method, it was able to capture an additional flight line without violating the range constrain of the mission, making it the more optimal choice as it achieves a route higher net utility.

4.3 Case Study: Multi-Agent Helheim Glacier

In this section, simulations are conducted to showcase the multi-agent functionality of the developed autonomy for the upcoming Helheim deployments discussed in Section 4.2. The performance of the systems are then compared to the single-agent condition to assess the potential benefits in terms of operational capabilities. These simulations were conducted in the multi-agent MATLAB Simulink frameworks outlined in Section 3.7, utilizing the equations of motion described in Section 3.1 and the aircraft dynamic models outlined in Section 3.2. The methods for flight line allocation amongst the agents in the operation are described in Section 2.5-2.8. Note that following the flight line allocation, each agent utilizes the Forward Greedy TSP method for surveying its respective flight lines.

4.3.1 Collaborative Survey using Two Agents

In the first multi-agent scenario, the broad survey flight lines from Section 4.2.1 are flown utilizing Mugin-2930 UASs performing collaborative surveying. These agents utilize the Hungarian Assignment methods outlined in Section 2.5 in order to efficiently allocate the flight lines amongst themselves in real-time. Following the successful tracking of the mission flight lines, the agents return to their respective home loiter circles. The mission scenario is shown in Figure 77, while Figure 78 and Figure 79 show various portions of the mission operation. Figure 80 shows the final tracking of the survey operation. The total deployment times for each aircraft is shown in Table 11, which denotes the time between the aircraft deploying to survey the flight lines to the time that it returns to the home loiter circle. The total deployment time for a single aircraft in the same mission scenario is also given for comparison. Note that this single aircraft is using the optimal route developed from the TSP algorithms in Section 4.2.1.

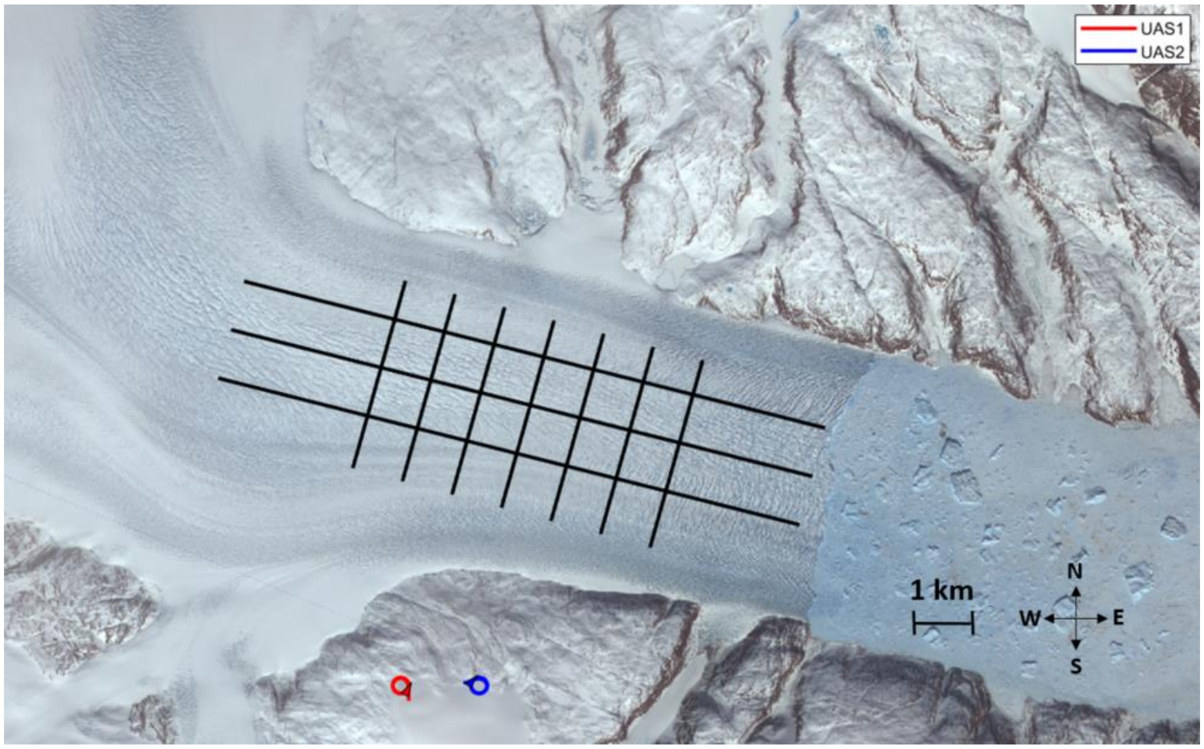


Figure 77: Multi-Agent Scenario 1, t = 0 seconds

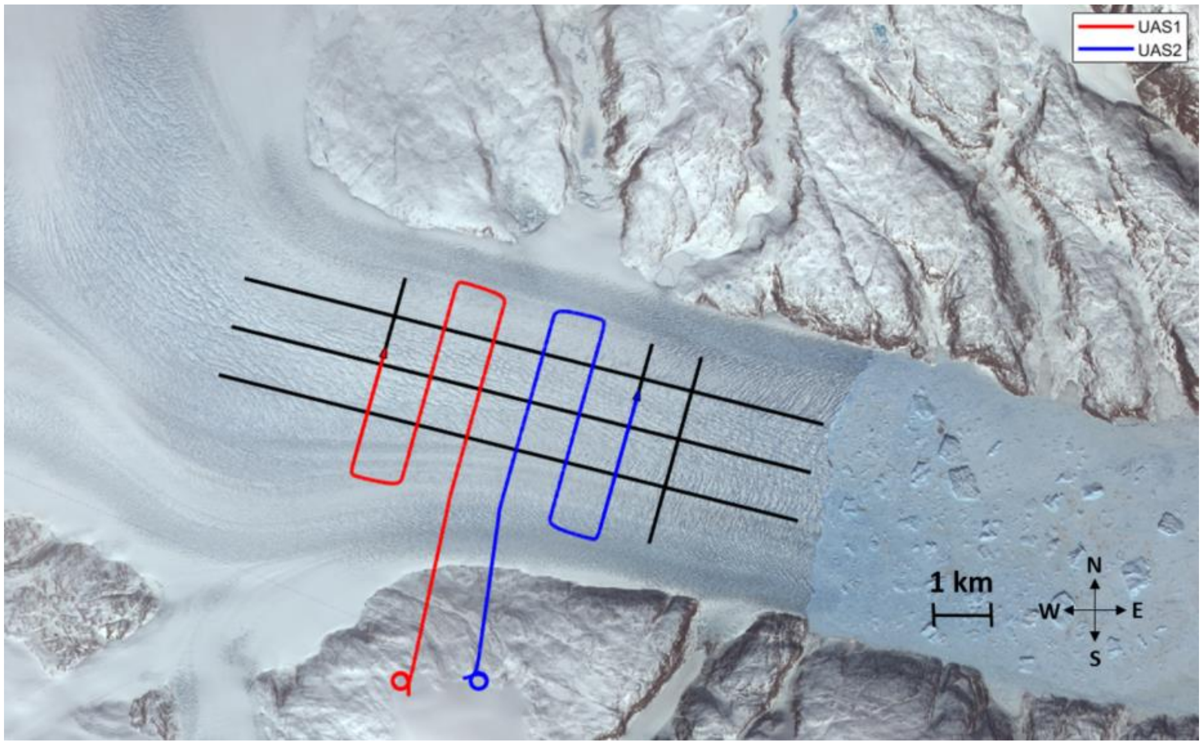


Figure 78: Multi-Agent Scenario 1, t = 675 seconds

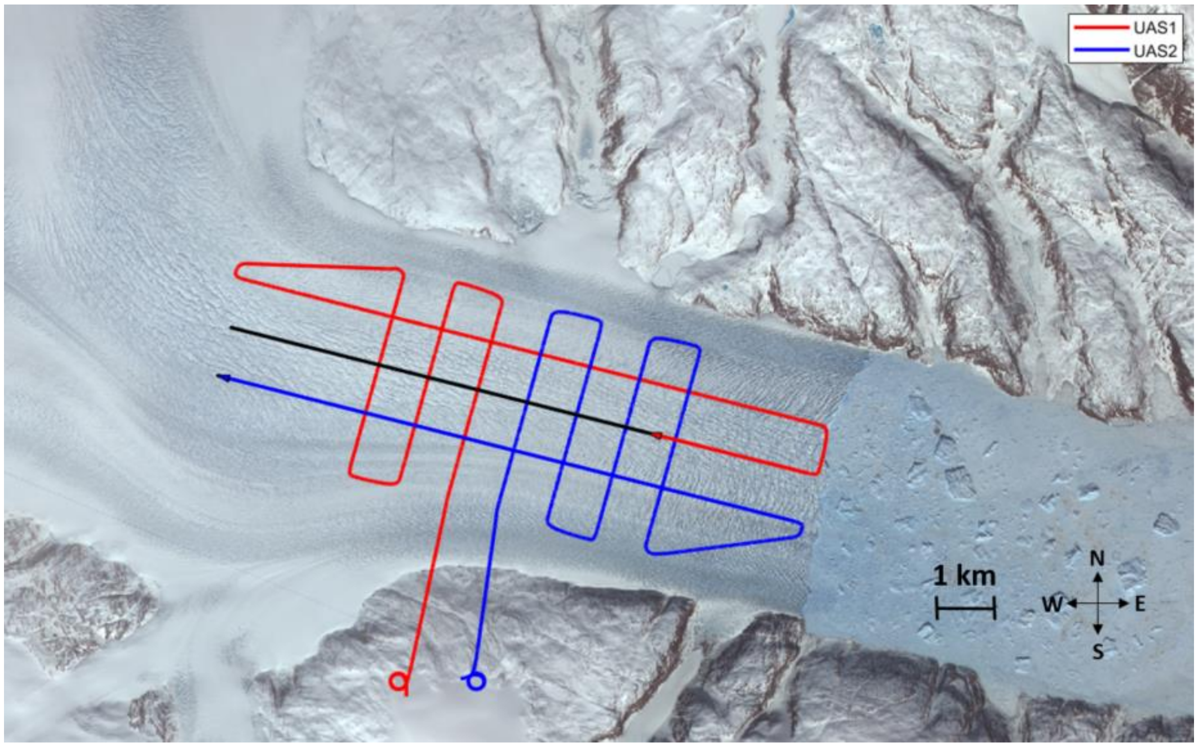


Figure 79: Multi-Agent Scenario 1, t = 1,540 seconds

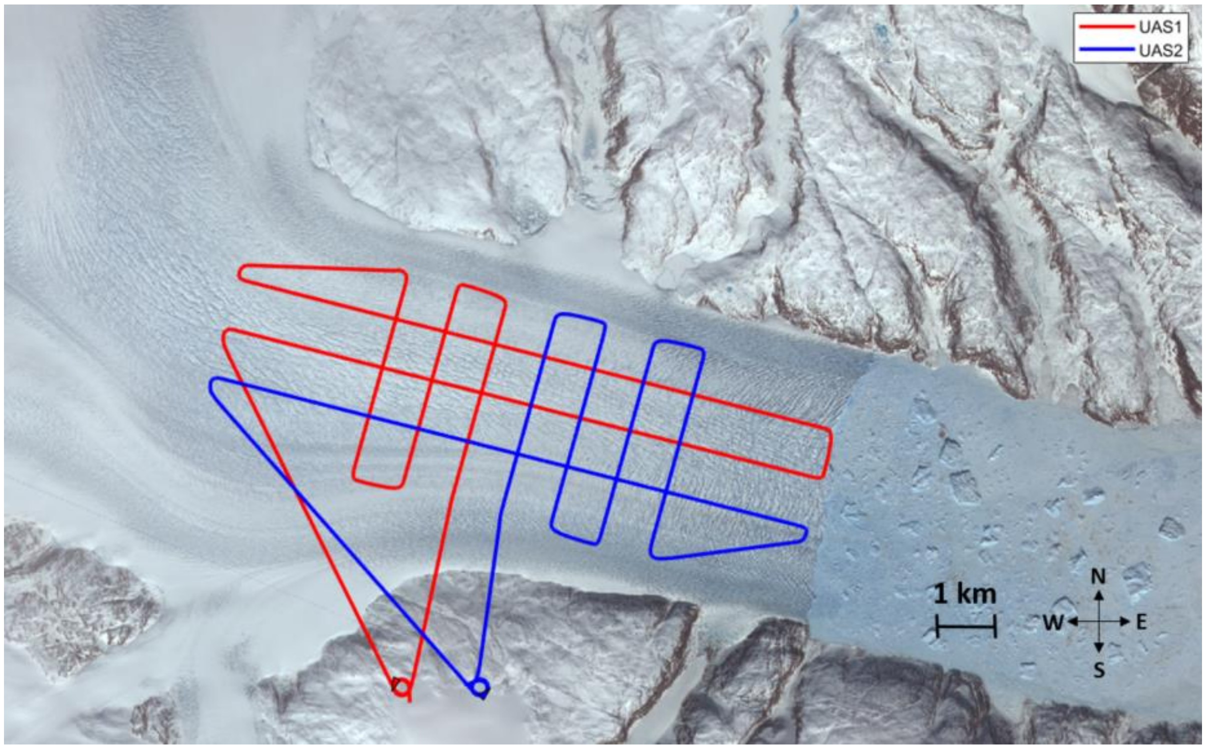


Figure 80: Multi-Agent Scenario 1, Final Tracking

Table 11: Multi-Agent Scenario 1, Deployment Times

UAS1 Total Time Deployed	36 minutes, 49 seconds
UAS2 Total Time Deployed	31 minutes, 14 seconds
Total Multi-Agent Operational Time	36 minutes, 49 seconds
Total Operational Time for Single Agent Condition	58 minutes, 36 seconds
Percent Reduction in Operational Time	37.17%

From Table 11, it can be seen that the two agents can collaboratively survey the flight lines in a shorter amount of time than the single agent. By utilizing Hungarian Assignment iteratively onboard each of the agents, the mission flight lines are efficiently allocated between the two aircraft. Note that they are allocating all of the remaining flight lines amongst the agents in real-time, not simply the nearest immediate two flight lines. Also note that this Hungarian Assignment process is conducted in a distributed manner onboard each of the aircraft, using their current position and the position of the other agents communicated over the telemetry modems. While the two aircraft were able to collaboratively survey the flight lines in a shorter amount of time than the single agent condition, this total operational time is greater than half of the single agent result due to the time spent transiting from the home loiter location to the mission flight lines in the outlet glacier channel.

4.3.2 Collaborative Survey using Five Agents

In the second multi-agent scenario, the comprehensive survey mission will be conducted with five Mugin-2930 UASs performing collaborative surveying. Similar to the previous scenario, the agents utilize the Hungarian Assignment algorithms outlined in Section 2.5 in order to allocate flight lines, and utilize a Forward Greedy TSP method in order to develop flight routes through their respective flight lines. Figure 81 shows the mission scenario, while Figure 82, Figure 83, and Figure 84 show various portions of the mission operation. Figure 85 shows the final tracking of the survey operation. The total deployment times for each aircraft is shown in Table 12, where the total deployment time for a single aircraft in the same mission scenario is also given for comparison. Note that this single aircraft is using the optimal route developed from the TSP algorithms in Section 4.2.3.

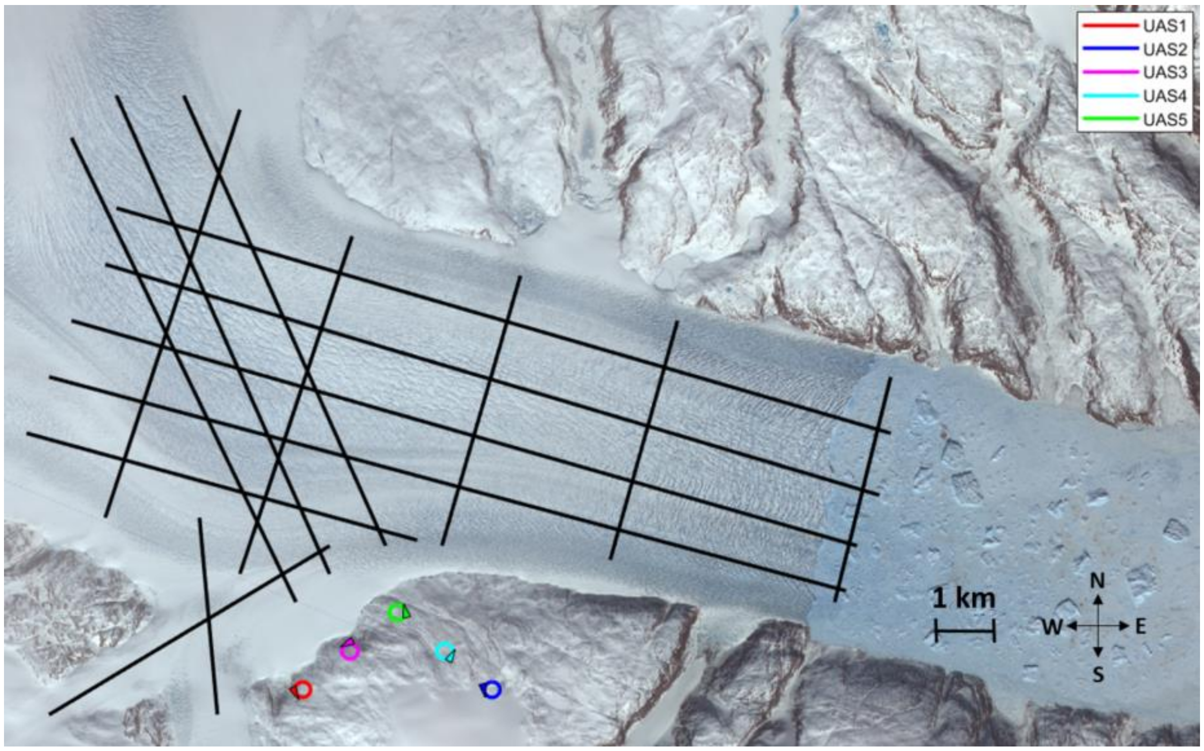


Figure 81: Multi-Agent Scenario 2, t = 0 seconds

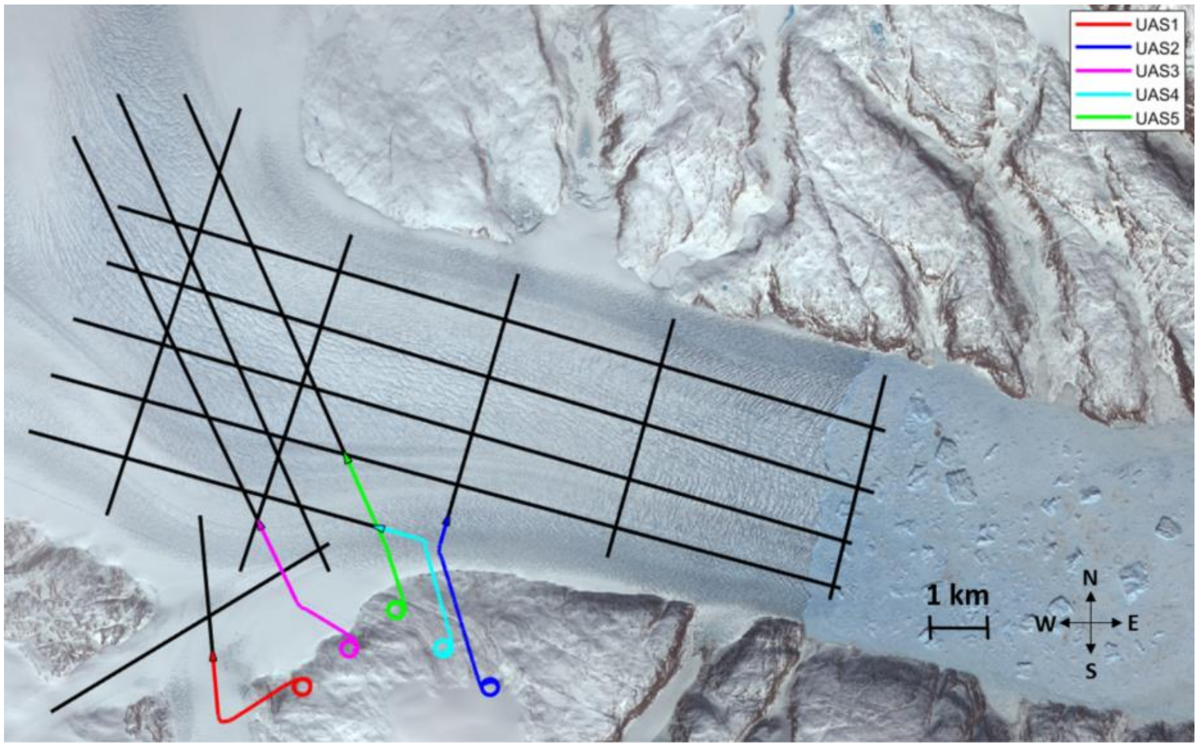


Figure 82: Multi-Agent Scenario 2, t = 130 seconds

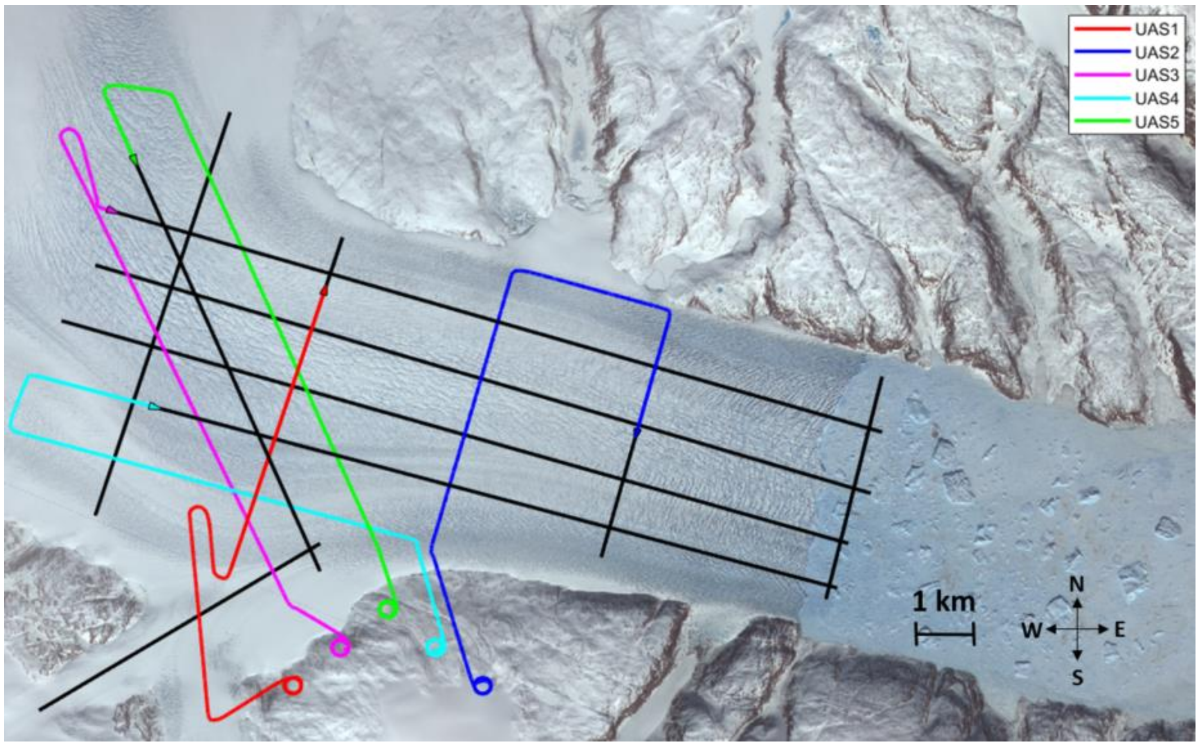


Figure 83: Multi-Agent Scenario 2, t = 570 seconds

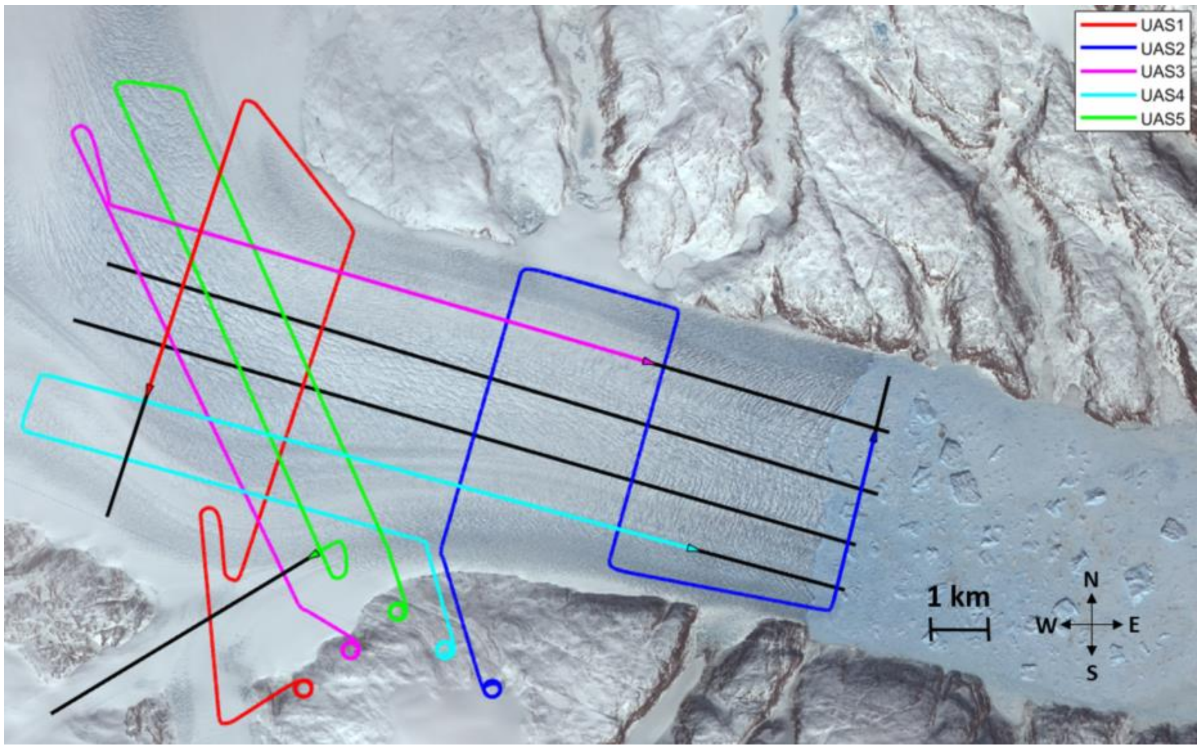


Figure 84: Multi-Agent Scenario 2, t = 1,000 seconds

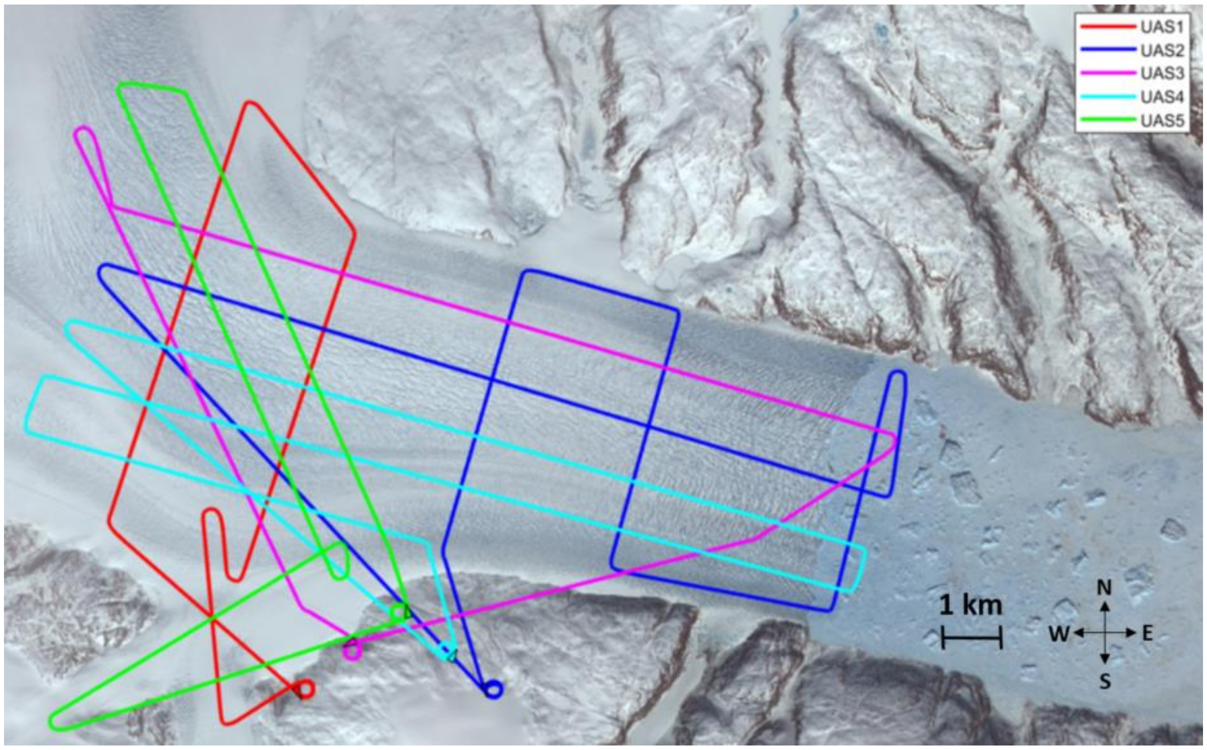


Figure 85: Multi-Agent Scenario 2, Final Tracking

Table 12: Multi-Agent Scenario 2, Deployment Times

UAS1 Total Time Deployed	22 minutes, 3 seconds
UAS2 Total Time Deployed	38 minutes, 4 seconds
UAS3 Total Time Deployed	27 minutes, 59 seconds
UAS4 Total Time Deployed	37 minutes, 8 seconds
UAS5 Total Time Deployed	25 minutes, 51 seconds
Total Multi-Agent Operational Time	38 minutes, 4 seconds
Total Operational Time for Single Agent Condition	2 hours, 1 minute, 22 seconds
Percent Reduction in Operational Time	68.64%

From Table 12 it can be observed that the five agents can survey the flight lines much quicker than the single agent condition. Through the Hungarian Assignment process, all of the mission flight lines are efficiently allocated amongst the five agents at each iteration of the autonomy functions in the simulation. Note that the computational complexity of this allocation process decreases as the number of remaining flight lines decreases. Additionally, as discussed in Sections 4.2.4 and 4.2.5, note that the Mugin-2930 aircraft has a 90 minute endurance and that a single aircraft would realistically have to perform multiple flight operations in order to survey all of the flight lines in this mission scenario. The total mission time for these iterative operations would be much greater, and would include time spend in takeoff and landing phases, as well as refueling and recharging/swapping batteries.

4.3.3 Five Agents featuring Robustness to System Failures

In the third multi-agent scenario, the comprehensive survey operation using five Mugin-2930 UASs from the previous section will be repeated, but UAS2 and UAS4 will experience onboard system failures at t = 570 seconds into the operation. This scenario is introduced to showcase the operational robustness of the multi-agent autonomy towards single-agent system failures. When these agents detect their onboard system failure, they will cease surveying their flight lines, notify neighboring agents of their failure status, and immediately return to their respective home loiter circle. As noted in Section 2.8, since the flight lines that these agents were tracking were not finished, they are then available for allocation amongst the remaining agents in the operation. Figure 86 shows the initial mission scenario, while Figure 87 shows the initial collaborative surveying operation similar to the previous scenario. However, Figure 88 shows the reaction of the systems following the system failures aboard UAS2 and UAS4, and Figure 89 shows UAS3 and UAS5 beginning to survey the unfinished flight lines that were left by the agents that experienced the system failures. Figure 90 shows the final tracking of the collaborative survey operation, where it can be observed that all of the mission flight lines were successfully surveyed, and Table 13 shows the total deployment time for each agent in the operation.

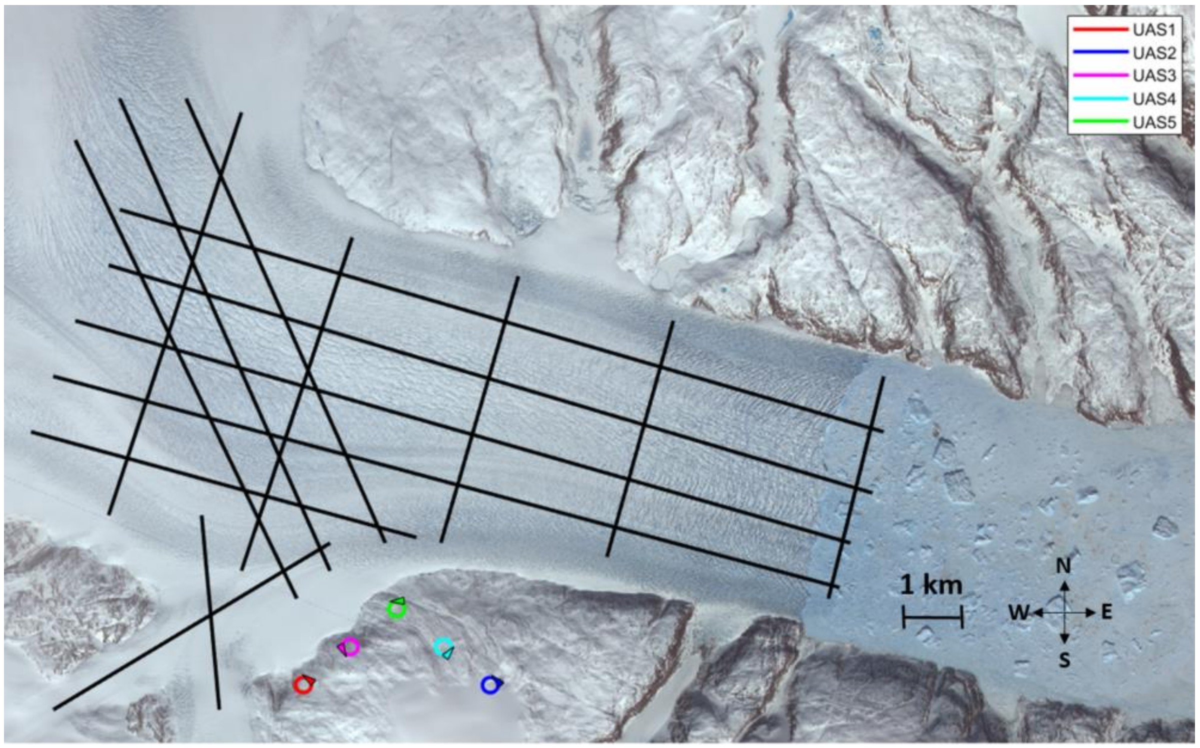


Figure 86: Multi-Agent Scenario 3, t = 0 seconds

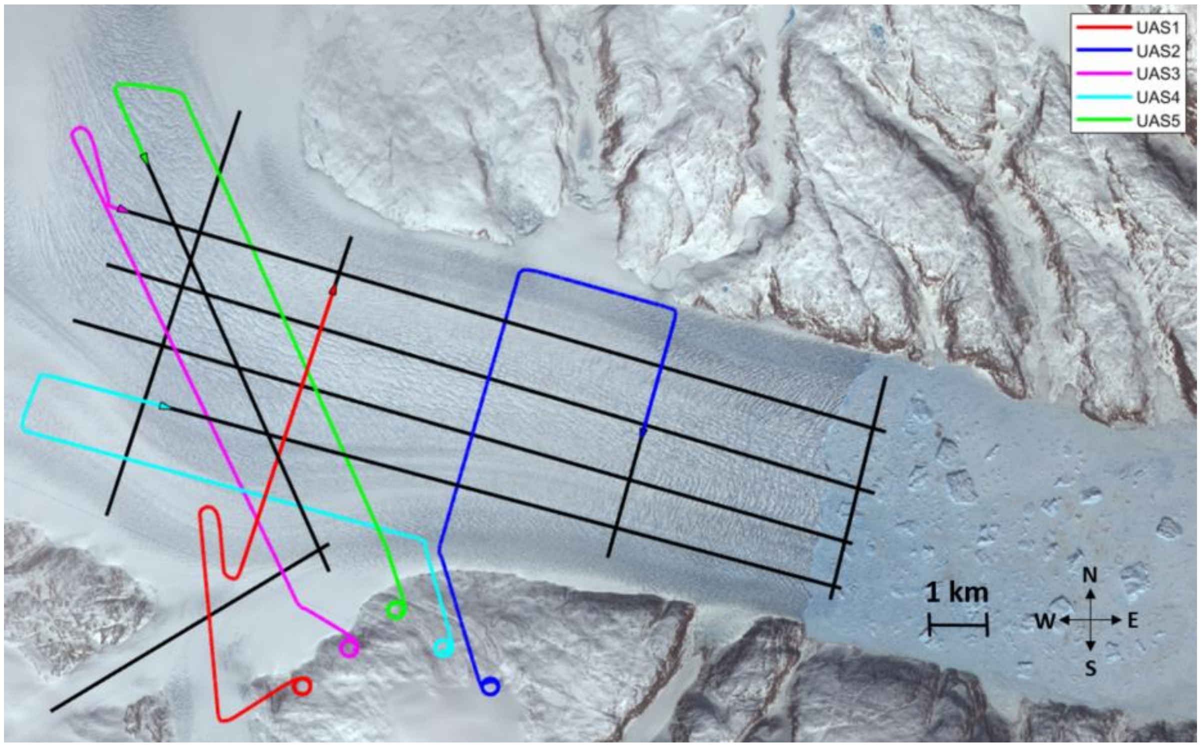


Figure 87: Multi-Agent Scenario 3, t = 570 seconds

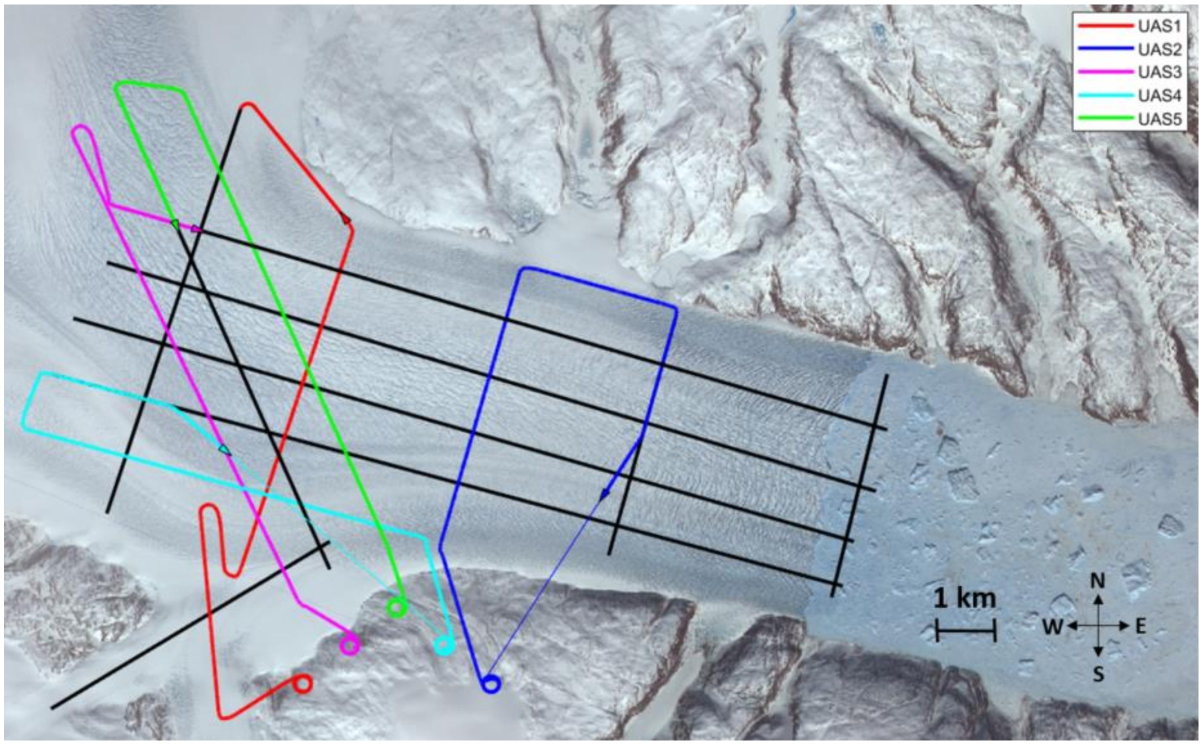


Figure 88: Multi-Agent Scenario 3, t = 630 seconds

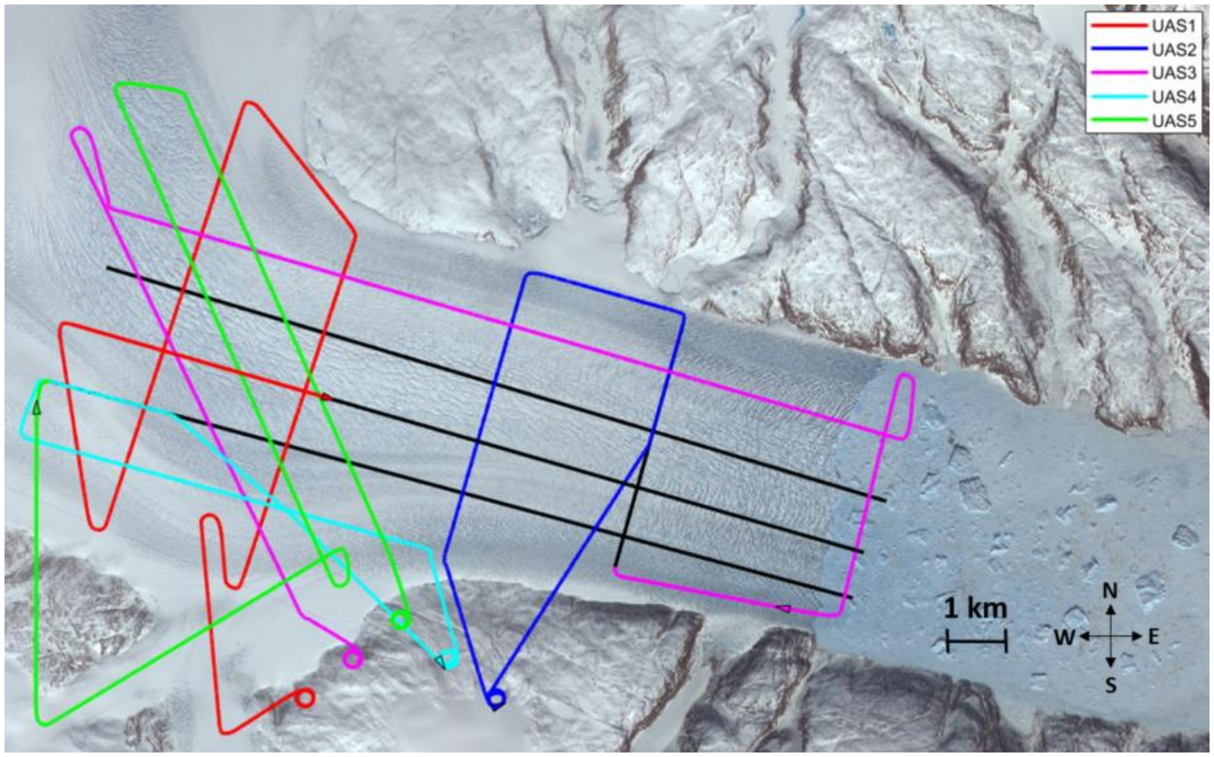


Figure 89: Multi-Agent Scenario 3, t = 1,500 seconds

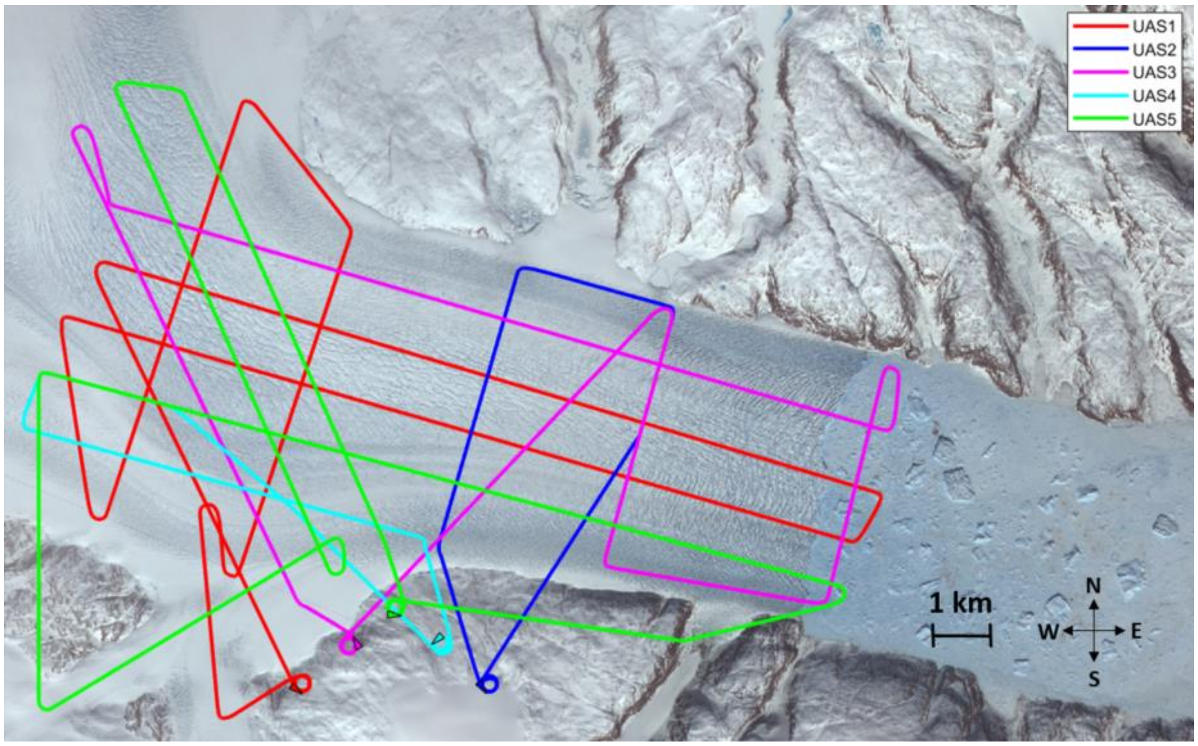


Figure 90: Multi-Agent Scenario 3, Final Tracking

Table 13: Multi-Agent Scenario 3, Deployment Times

UAS1 Total Time Deployed	50 minutes, 16 seconds
UAS2 Total Time Deployed	13 minutes, 33 seconds
UAS3 Total Time Deployed	37 minutes, 12 seconds
UAS4 Total Time Deployed	14 minutes, 30 seconds
UAS5 Total Time Deployed	42 minutes, 39 seconds
Total Multi-Agent Operational Time	50 minutes, 16 seconds
Total Operational Time for Single Agent Condition	2 hours, 1 minute, 22 seconds
Percent Reduction in Operational Time	58.58%

In this scenario, the agents were able to adapt to the single system failures of Agents 2 and 4, and were able to dynamically reallocate the flight lines amongst the remaining functional agents in the collaborative survey operation. Additionally, the agents were still able to successfully survey all of the mission flight lines in a significantly reduced time as compared to the single agent condition, though in a slightly longer amount of time than in Section 4.3.3 where all of the agents were functional throughout the operation.

4.3.4 Scheduling Survey using Three Agents

In a fourth multi-agent scenario, the comprehensive flight lines will be covered by three Mugin-2930 UASs utilizing the scheduling algorithm outlined in Section 2.6, so that only one agent will be surveying the flight lines at any given time. Note that this scheduling algorithm was introduced in order to reduce possible inter-agent interference from the onboard radar systems. The fuel constraint for each agent corresponded to a deployment time of 1 hour. Figure 91 shows the mission scenario, while Figure 92, Figure 93, and Figure 94 show the operations following the surveying for UAS1, UAS2, and UAS3, respectively. Note that each subsequent agent is deployed as soon as the previous agent begins its return to their home loiter, reducing the operational time while ensuring only 1 agent is radiating at any given instance. Figure 95 shows the final tracking of the survey operation. Table 14 details the total deployment time of the scheduling survey operation, where the total time is compared with the single agent condition, using the optimal route developed from the TSP algorithms in Section 4.2.3.

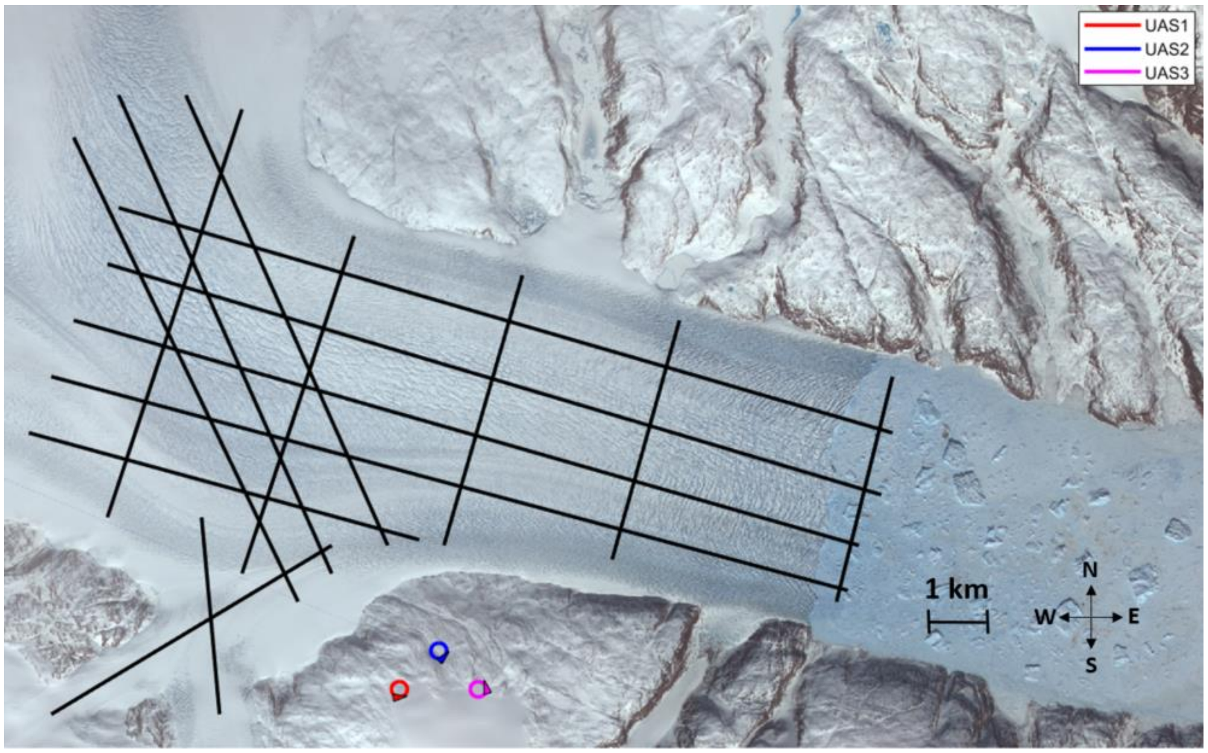


Figure 91: Multi-Agent Scenario 4, t = 0 seconds

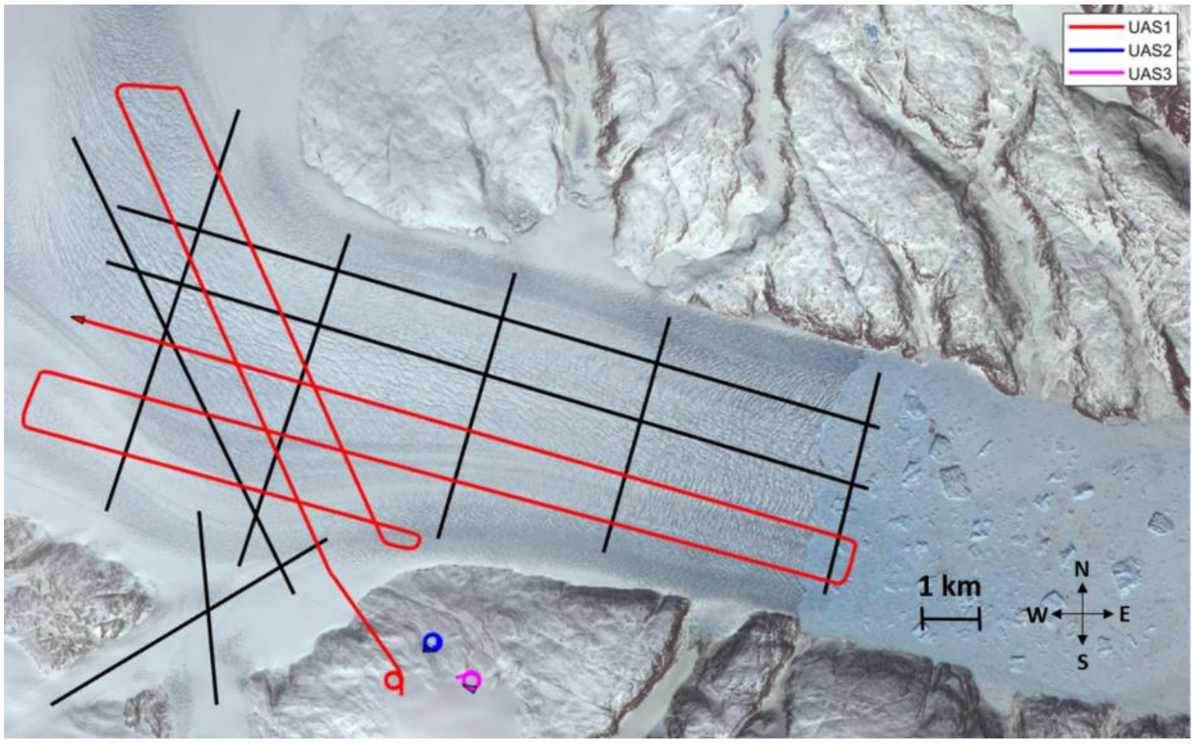


Figure 92: Multi-Agent Scenario 4, t = 2,750 seconds

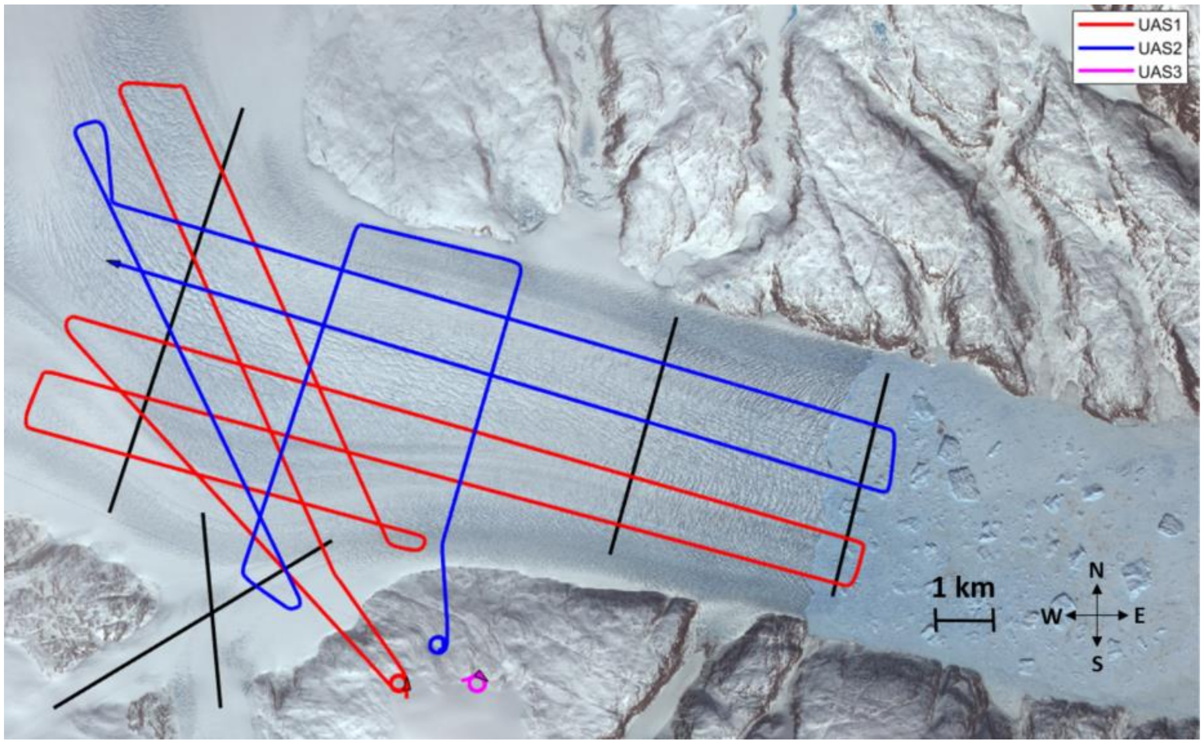


Figure 93: Multi-Agent Scenario 4, t = 5,400 seconds

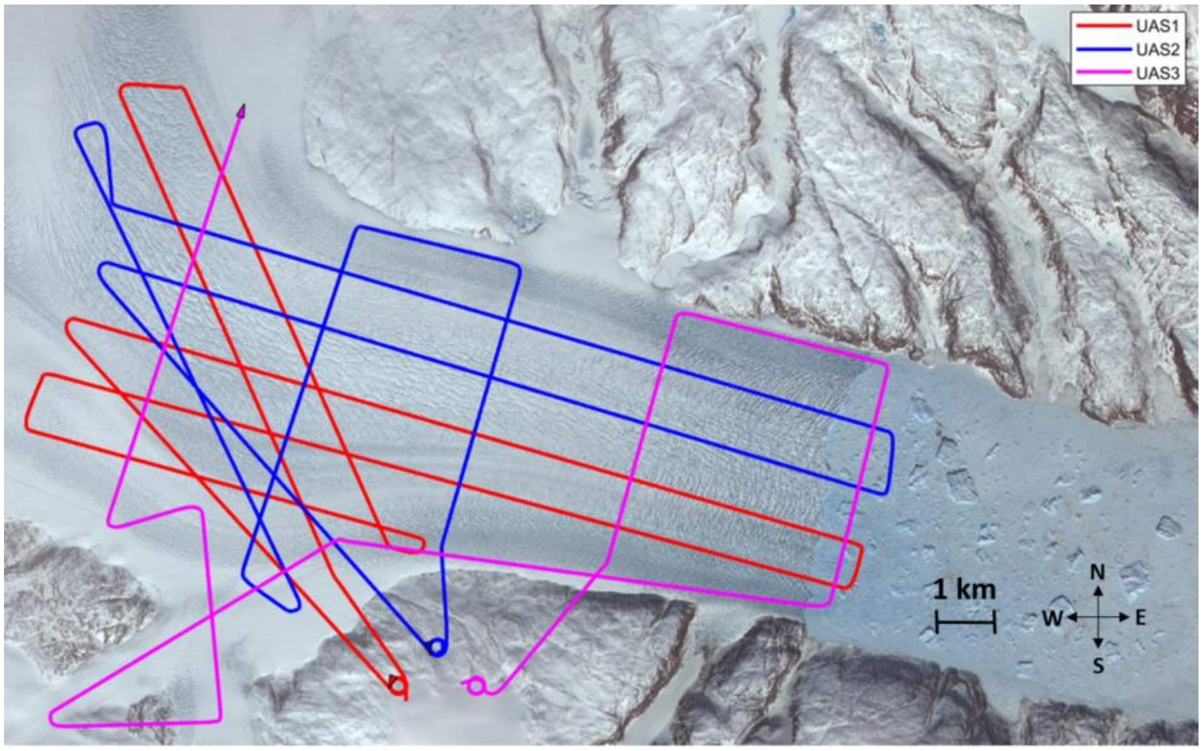


Figure 94: Multi-Agent Scenario 4, t = 7,580 seconds

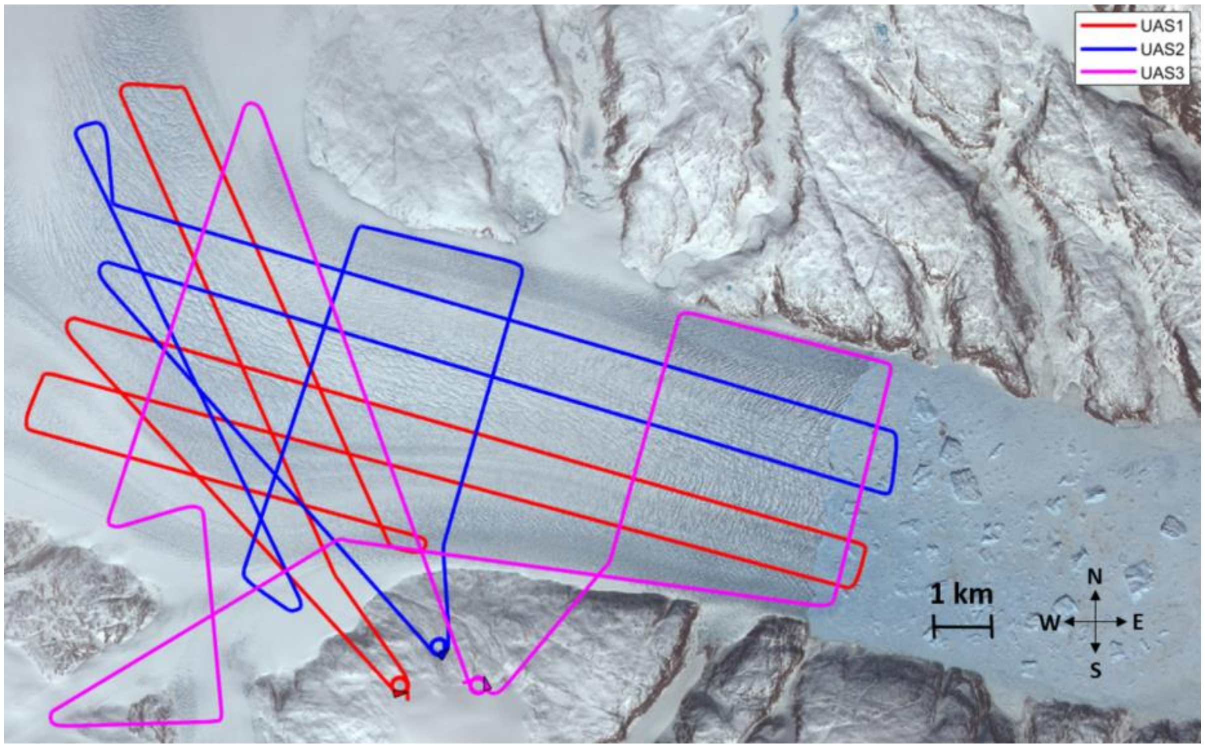


Figure 95: Multi-Agent Scenario 4, Final Tracking

Table 14: Multi-Agent Scenario 4, Deployment Times

UAV1 Total Time Deployed	52 minutes, 25 seconds
UAV2 Total Time Deployed	51 minutes, 9 seconds
UAV3 Total Time Deployed	43 minutes, 57 seconds
Total Multi-Agent Operational Time	2 hours, 14 minutes, 48 seconds
Total Operational Time for Single Agent Condition	2 hours, 1 minute, 22 seconds
Percent Reduction in Operational Time	-11.07%

From Table 14 it can be observed that while the scheduling operation took slightly longer than the single-agent condition from Section 4.2.3, this single-agent condition is actually infeasible due to the endurance constraints for the aircraft (~90 minutes). However, by utilizing the scheduling autonomy in this scenario, the three agents were able to feasibly survey the flight lines in a relatively similar amount of time. This scenario shows promising results for the rapid deployment and recovery of systems into and out of the mission flight area in deployment scheduling operations.

4.3.5 Heterogeneous Survey using Three Agents

In the final multi-agent scenario, a heterogeneous collaborative survey operation will be conducted using two Mugin-2930 UASs alongside one Mugin-4450 UAS. Characteristics of these platforms are detailed in Section 3.2, where it should be noted that the trim airspeed for the Mugin-2930 platform is 48 knots, while the Mugin-4450 has a trim airspeed of 61 knots. In this scenario shown in Figure 96, three clusters of five tightly-spaced flight lines are oriented in the along-flow direction of the glacial channel. The approximate 7 foot spacing between these flight lines corresponds to the quarter-wavelength of a 35 MHz operating radar system, for use in cross-track synthetic aperture processing methods described in Section 1.4.2. Note the respective home loiter locations for the three agents in Figure 96, where the Mugin-4450 system has a larger loiter radius due to its increased trim airspeed. Once deployed, the developed autonomy will utilize the space partitioning methods described in Section 2.5.4 in order to group the 15 total flight lines into three distinct clusters, and utilize the heterogeneous modifications described in Section 2.7 in order to perform the Hungarian Assignment between these clusters and the agents. Figure 97 shows the initial deployment of the systems onto the survey flight lines, and Figure 98 shows how the agents continue to survey the flight lines in their assigned cluster. Figure 99 shows the instant that UAS3 has finished surveying its respective cluster, and since all the remaining clusters have an assigned agent to survey, UAS3 will begin to return to its home loiter. Figure 100 shows the final tracking for the survey operation, while Table 15 shows the total deployment times for the respective vehicles, as well as a comparison to the same scenario without the heterogeneous modifications to the Hungarian Assignment cost functions.

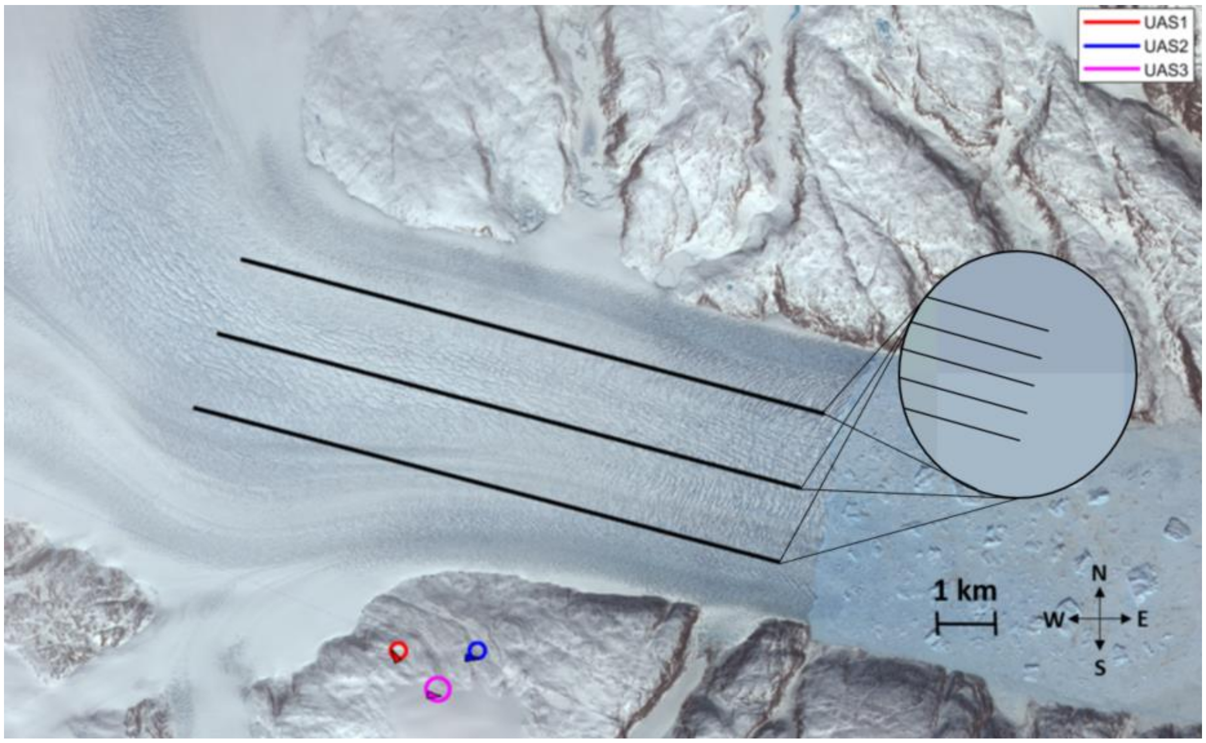


Figure 96: Multi-Agent Scenario 5, t = 0 seconds

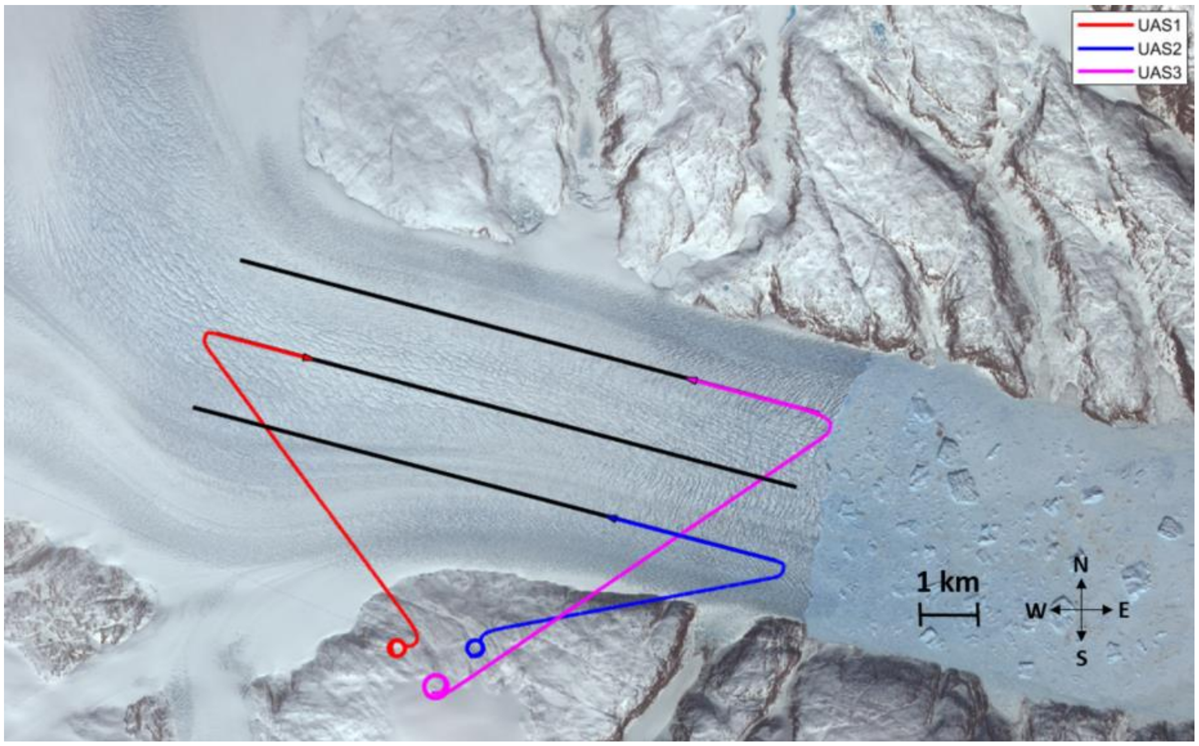


Figure 97: Multi-Agent Scenario 5, t = 880 seconds

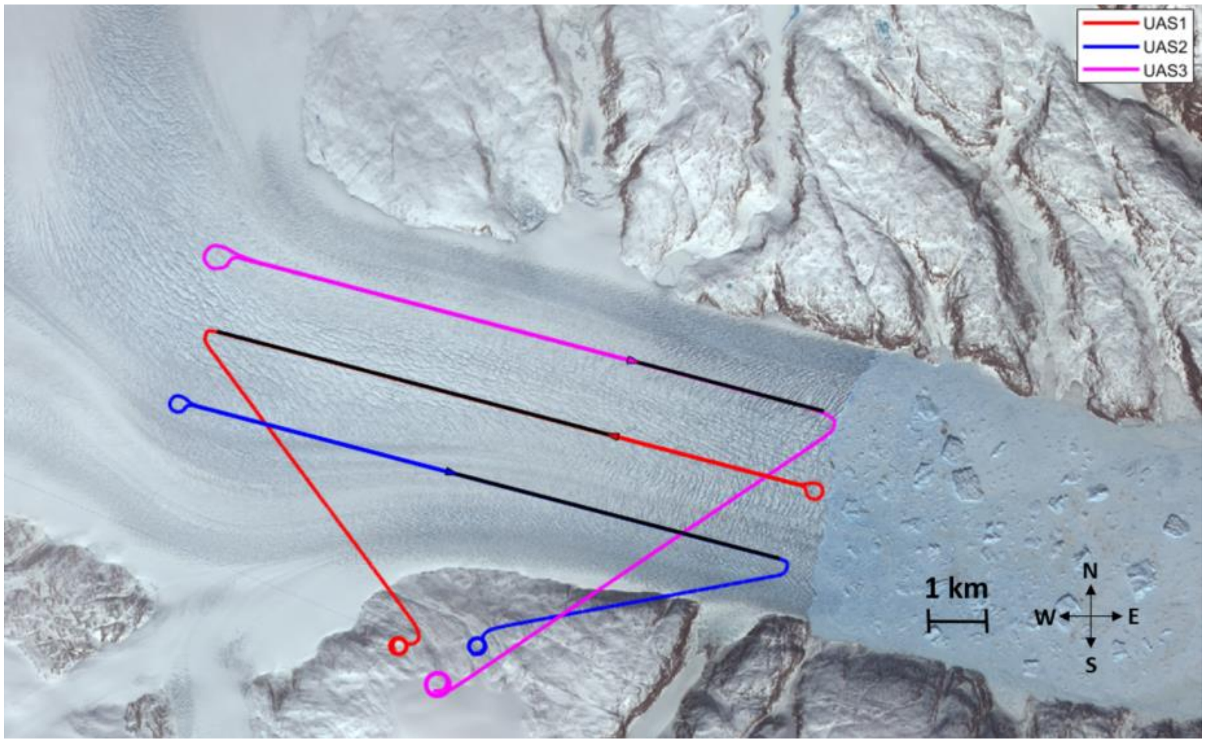


Figure 98: Multi-Agent Scenario 5, t = 1,480 seconds

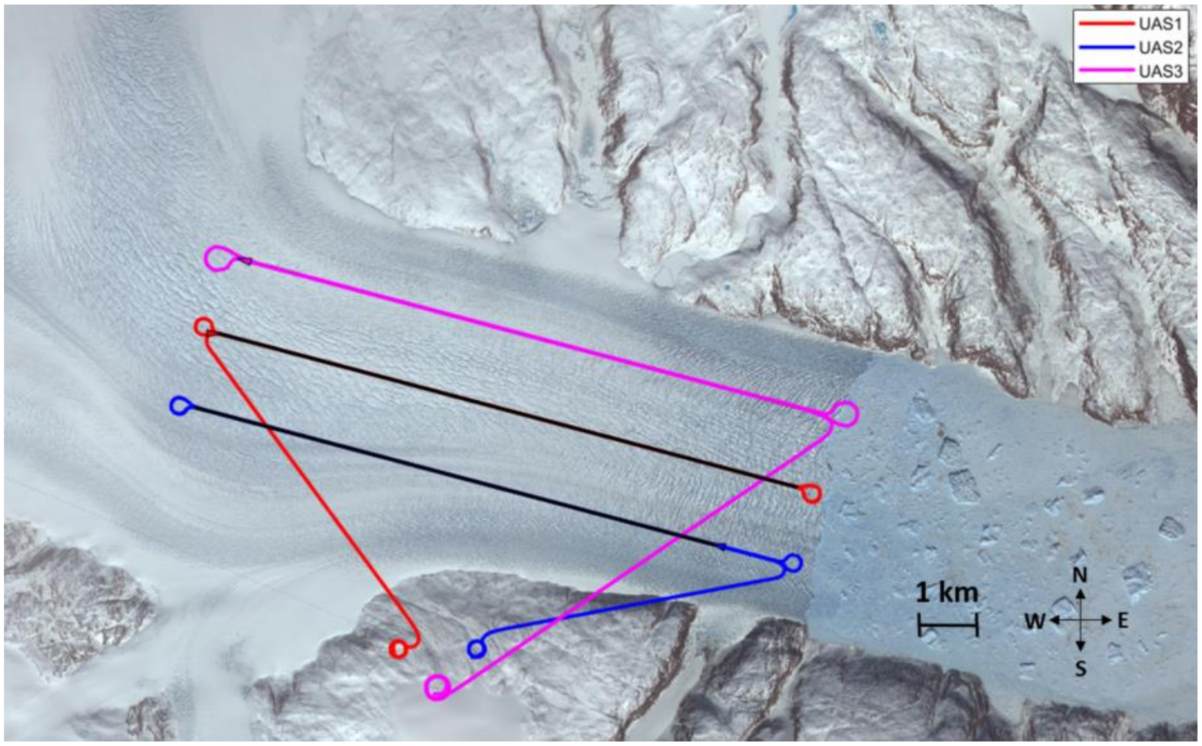


Figure 99: Multi-Agent Scenario 5, t = 2,930 seconds

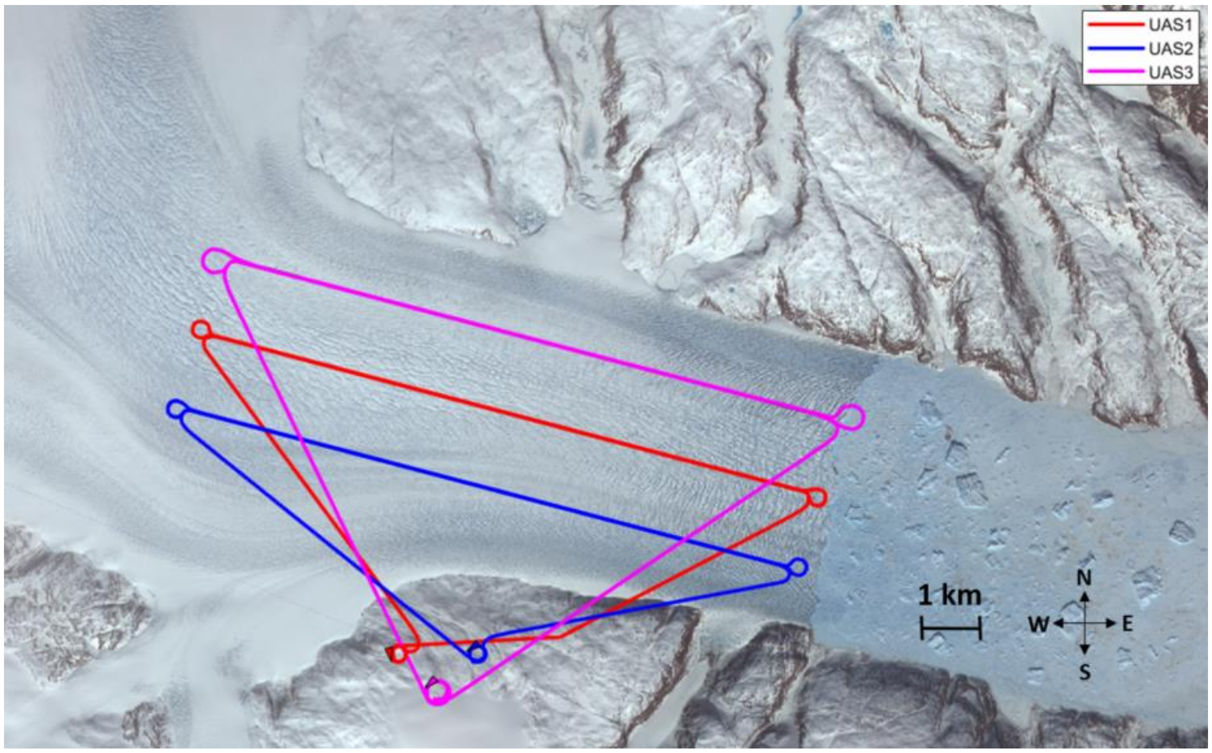


Figure 100: Multi-Agent Scenario 5, Final Tracking

Table 15: Multi-Agent Scenario 5, Deployment Times

UAV1 Total Time Deployed	54 minutes, 5 seconds
UAV2 Total Time Deployed	52 minutes, 26 seconds
UAV3 Total Time Deployed	45 minutes, 16 seconds
Total Multi-Agent Operational Time	54 minutes, 5 seconds
Total Operational Time without Heterogeneous Modifications	55 minutes, 27 seconds
Percent Reduction in Operational Time	2.46%

In this heterogeneous survey operation, although UAS3 was deployed from a home loiter location furthest (i.e. southern-most) from the flight lines, it was assigned to the furthest flight line cluster (i.e. northern-most) due to the heterogeneous modifications to the Hungarian Assignment cost functions, which consider not the distance, but the time required to survey the cluster. As UAS3 could transit onto and survey the flight line cluster faster than UAS1 and UAS2, this was the optimal assignment of the agents to clusters in terms of survey time. From Table 15, it can be observed that the heterogeneous modifications to the Hungarian Assignment cost functions were able to reduce the total flight time as compared to a similar scenario without the heterogeneous modifications (i.e. using distance as the Hungarian Assignment cost function), where UAS3 was assigned to the southern-most flight line cluster.

Chapter 5: Flight Test Validation

In this chapter, flight test validation for the developed autonomy will be presented, both as an onboard autonomy for unmanned system operations and as an off-board mission planner for manned operations.

5.1 Flight Test Procedure

Preliminary flight testing of the developed autonomy was conducted by the KU Flight Research Lab (KU FRL) at the Clinton International Model Airfield in Lawrence, Kansas. Shown in Figure 101, this field features a 450 ft North-South runway, as well as a 650 ft East-West runway, and has been extensively utilized by the KU FRL team for flight testing various systems. For these flight testing operations, the ground station is positioned under the shelter area in the center of the field, with the ground station telemetry antenna positioned on the roof of the shelter for increased directivity with the aircraft during flight testing activities.

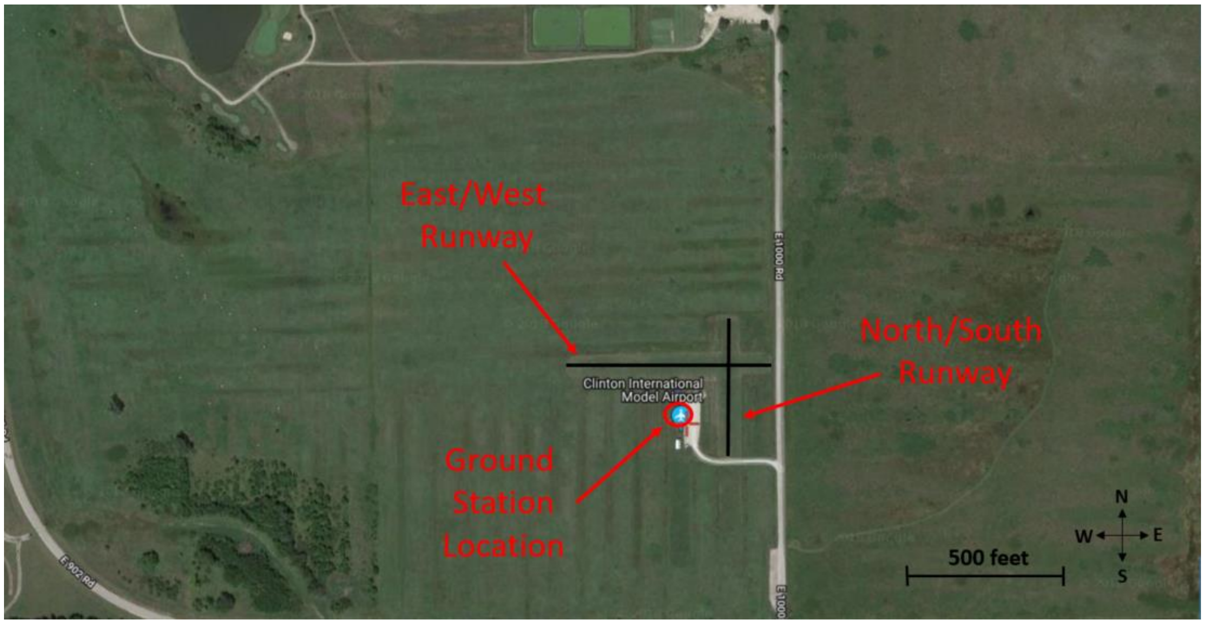


Figure 101: Flight Field

The aircraft utilized in this work for validation flight testing is the SkyHunter platform described in Section 3.2.2. A Pixhawk Cube flight control unit is utilized for aircraft sensor measurements and as a PWM generator, while the developed autonomy resides in a ROS framework (see Section 3.8) on an onboard Tegra Nano unit communicating to and from the Pixhawk using MAVROS protocols. In this architecture, the custom autopilot system onboard the Tegra Nano is capable of controlling the aircraft by sending control servo PWM commands to the Pixhawk using the Offboard mode in the PX4 firmware. Both the Pixhawk and the Tegra units have distinct telemetry modules and ground stations in order to view aircraft diagnostics in real-time.

Flight test activities are conducted through constant communication between the RC Pilot, the Ground Station Operators, and the Flight Test Lead. The RC Pilot uses an RC transmitter in order to manually control the aircraft during the takeoff and climb phases. Once at the desired altitude and in a trimmed condition about a target airspeed, the RC Pilot transitions control to the autopilot system running on the Tegra Nano board. The "base autopilot" developed by the KU FRL then controls the aircraft as it tracks four waypoints, creating a large racetrack pattern around the flight field. Once stable control of the aircraft is verified, the developed autonomy is engaged using a flag delivered in a waypoint upload from the ground station. Once engaged, the developed autonomy then commands a roll angle for the base autopilot's controller to track, and develops the appropriate commands to lead the aircraft into a loiter around the home location. The aircraft will continue to track this loiter circle until another flag is sent from the ground station, commanding the deployment of the system to survey the mission flight lines. During this preliminary flight testing, a 10 mph wind from the East was present, which was a significant factor in the tracking performance of the small UAS.

5.2 Racetrack Survey Mission

In the first flight test mission scenario, shown in Figure 102, six parallel survey flight lines of length 860 ft are positioned along the center of the field in order to create a "racetrack" pattern for surveying. This mission was designed in order to test the Dubins Path guidance methods for approaching the survey lines outlined in Section 2.2, as well as the corresponding lateral guidance outlined in Section 3.3, while keeping the aircraft inside the designated flight testing area. Note that for this scenario, the aircraft is utilizing the Forward Greedy TSP method in real-time for cognitive path planning the order to survey the flight lines. This process is occurring inside the Autonomy_Node (see Section 3.8), at a 1 Hz update rate, while the Dubins Paths and required roll commands are being generated at a 20 Hz rate by the GNC_Node.

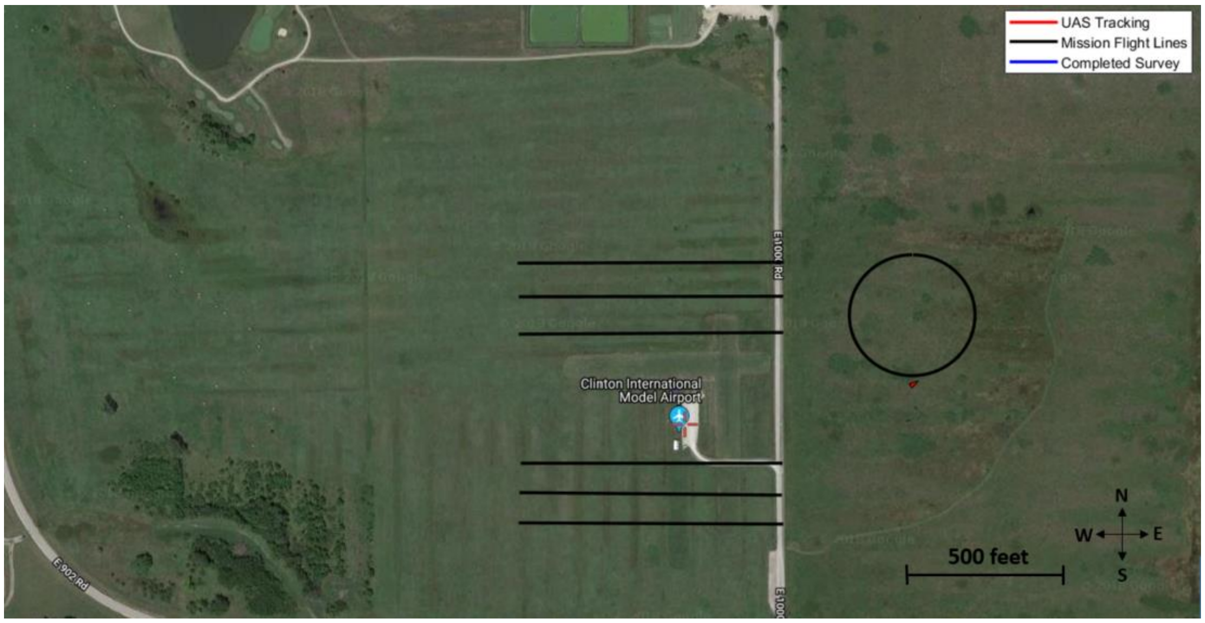


Figure 102: Flight 1 Mission Scenario

Figure 103 shows the moment that the aircraft deploy signal is transmitted, as well as the tracking of the aircraft in a counter-clockwise direction about the home loiter circle prior to the deployment signal, where it can be noted that the 10 mph eastern wind significantly affects the tracking performance on the western side of the loiter. This overshoot is due to the increased ground speed of the aircraft during the northern portion of the loiter, where the aircraft is experiencing a tailwind. Figure 104 shows the tracking of the aircraft as it enters onto the first mission flight line, which was chosen based on the minimum Dubins Path length among the flight lines in the mission. An overshoot occurs along this line due to the eastern wind, increasing the ground speed of the aircraft as it turns onto the flight line. In Figure 105, the aircraft has finished the first flight line and has followed a Dubins Path onto a second flight line. The tracking performance of this second flight line is improved due to the reduced ground speed of the aircraft due to the headwind during the approach stage onto the flight line. Note that the criteria for switching to the next flight line is that the aircraft must cross the half-plane of the flight line endpoint, as outlined in Section 3.3.4. Figure 106 to Figure 109 show the progression of the aircraft through the mission flight lines, using the Forward Greedy TSP method for flight line selection, and the Dubins Path guidance methods for approaching the selected flight lines. Note that the symmetric tracking performance of the northern flight lines compared to the southern, which can be explained by the eastern wind present during the flight operation. Figure 110 shows the final tracking of the aircraft through the mission flight lines and back to the home loiter circle for this scenario.

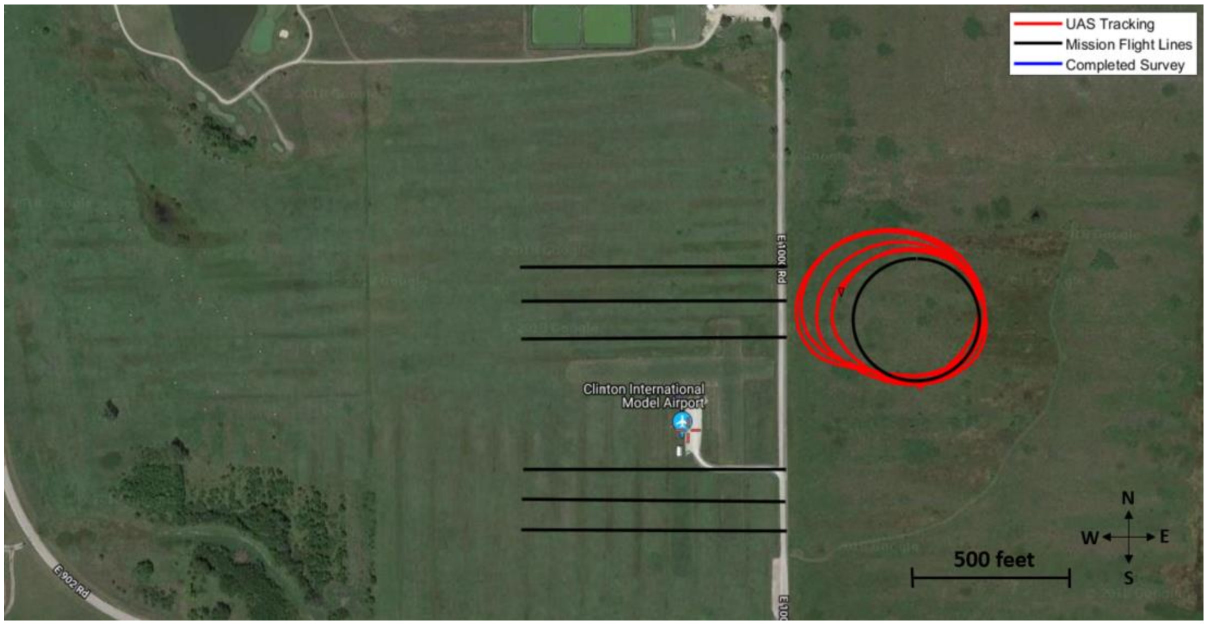


Figure 103: Flight 1 Tracking, t = 0 seconds after deployment

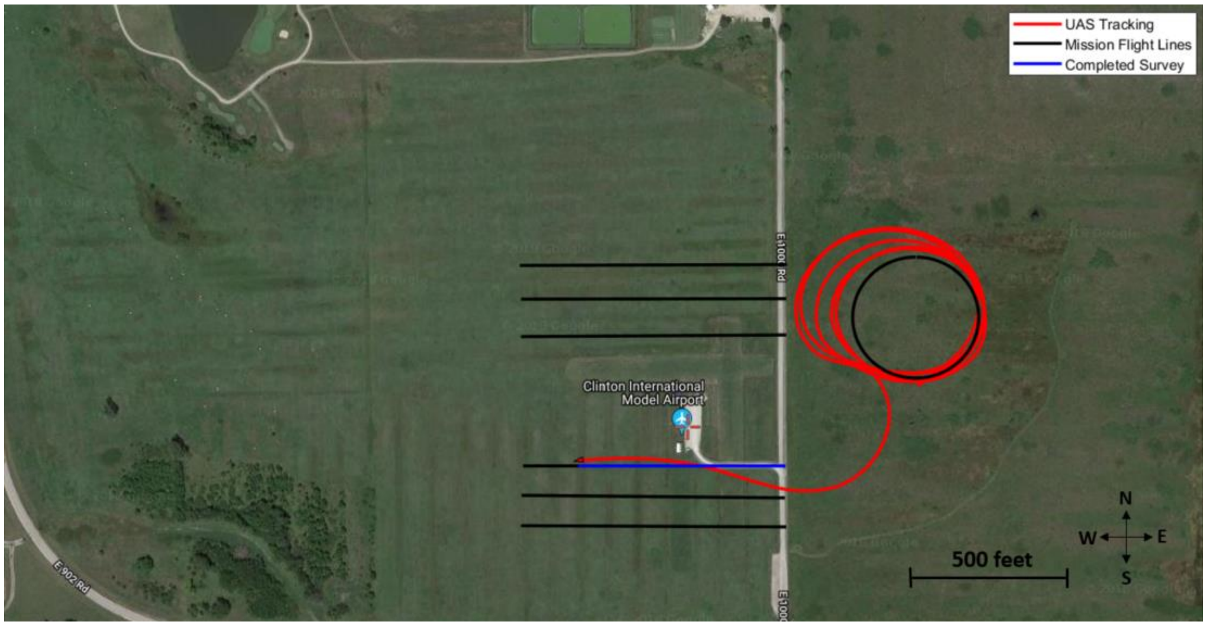


Figure 104: Flight 1 Tracking, t = 22 seconds

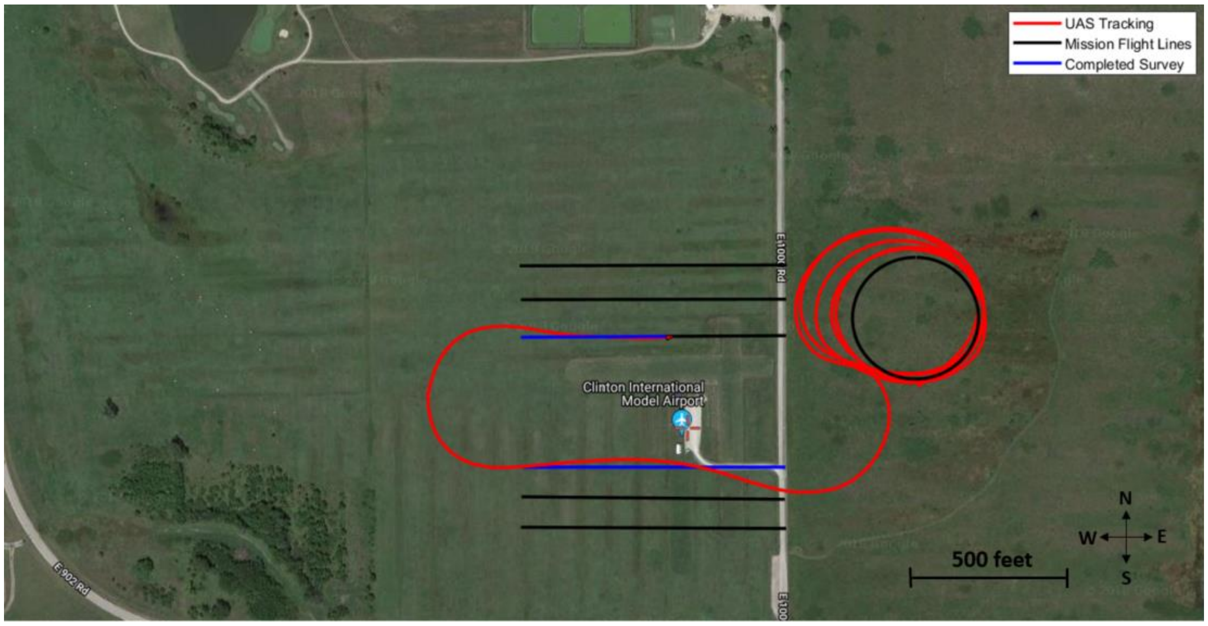


Figure 105: Flight 1 Tracking, t = 50 seconds

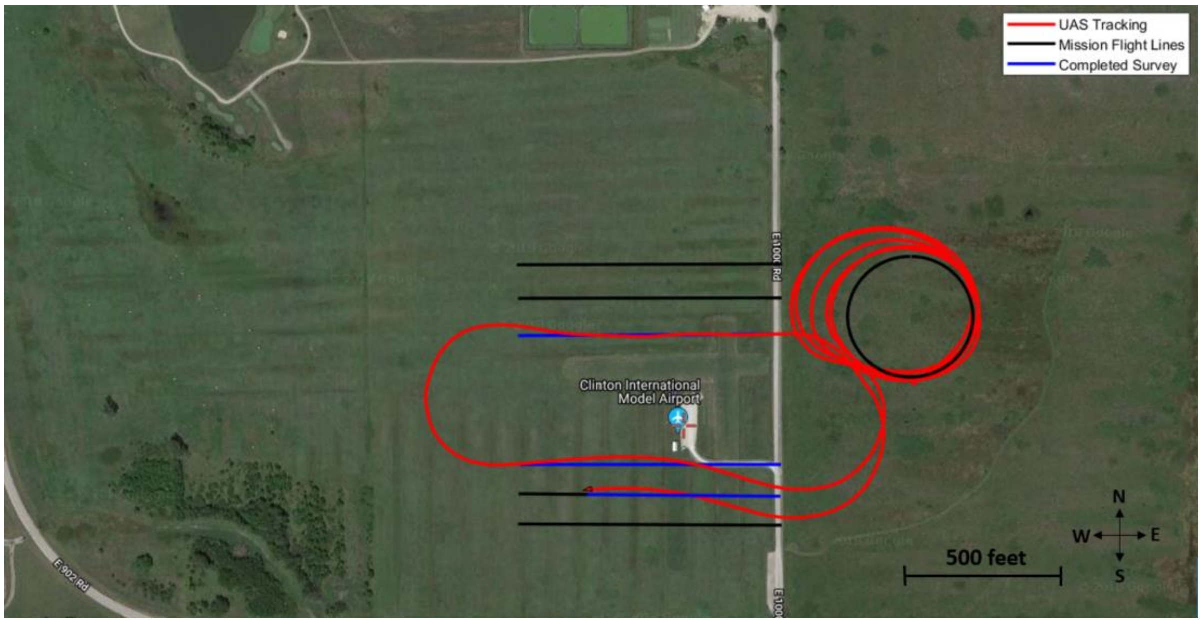


Figure 106: Flight 1 Tracking, t = 82 seconds

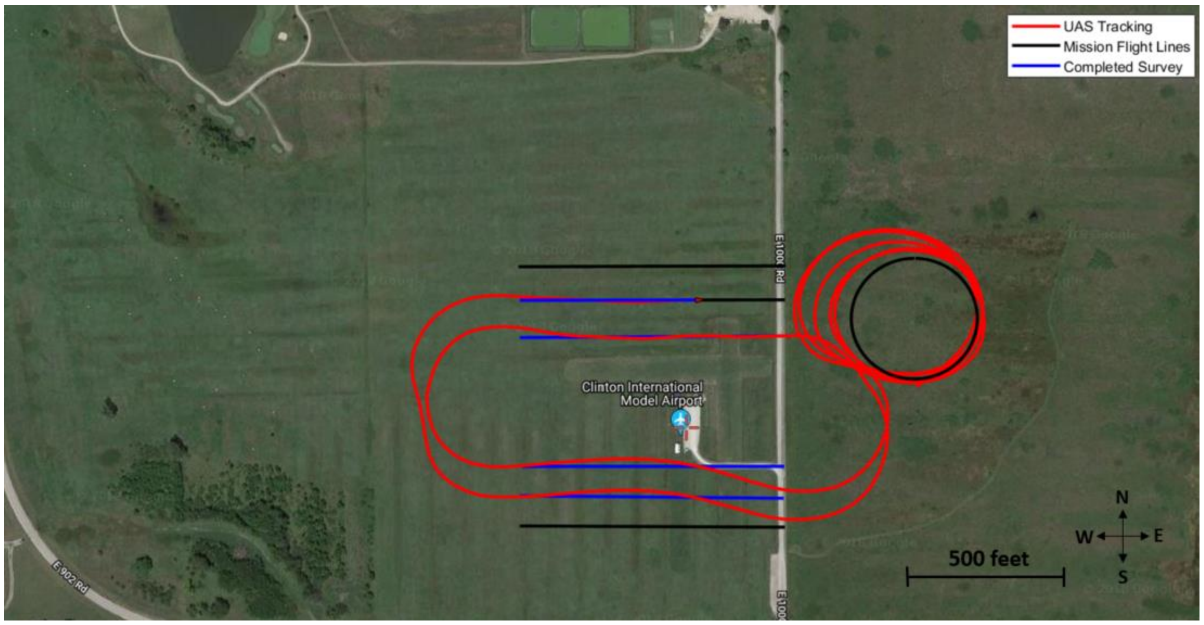


Figure 107: Flight 1 Tracking, t = 117 seconds

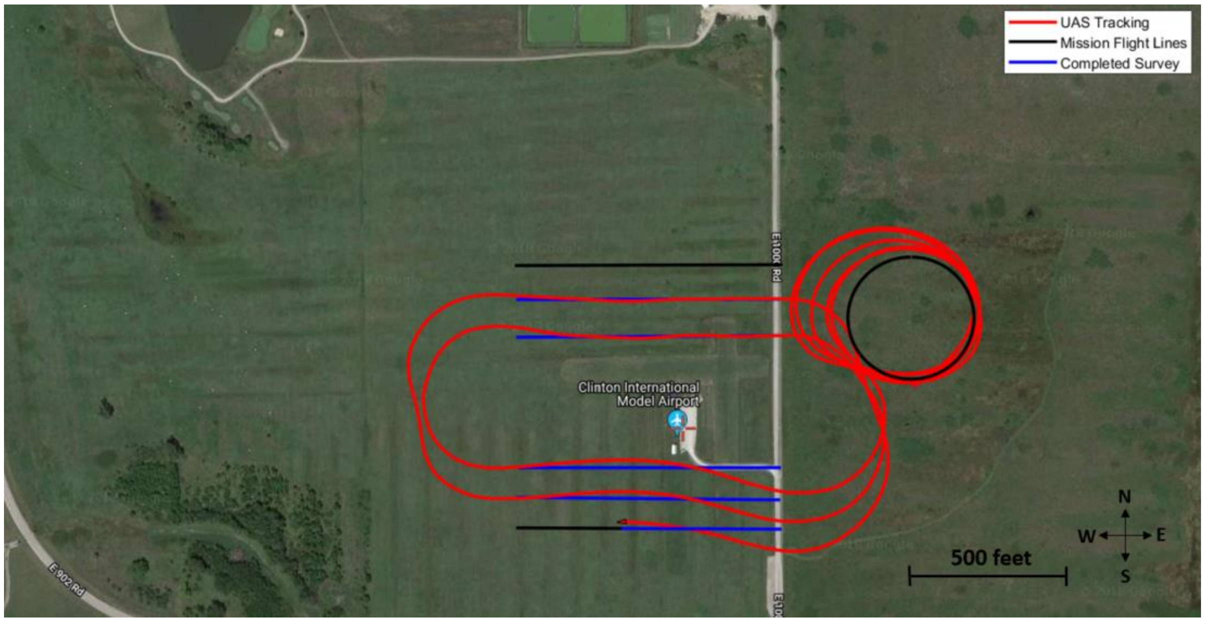


Figure 108: Flight 1 Tracking, t = 150 seconds

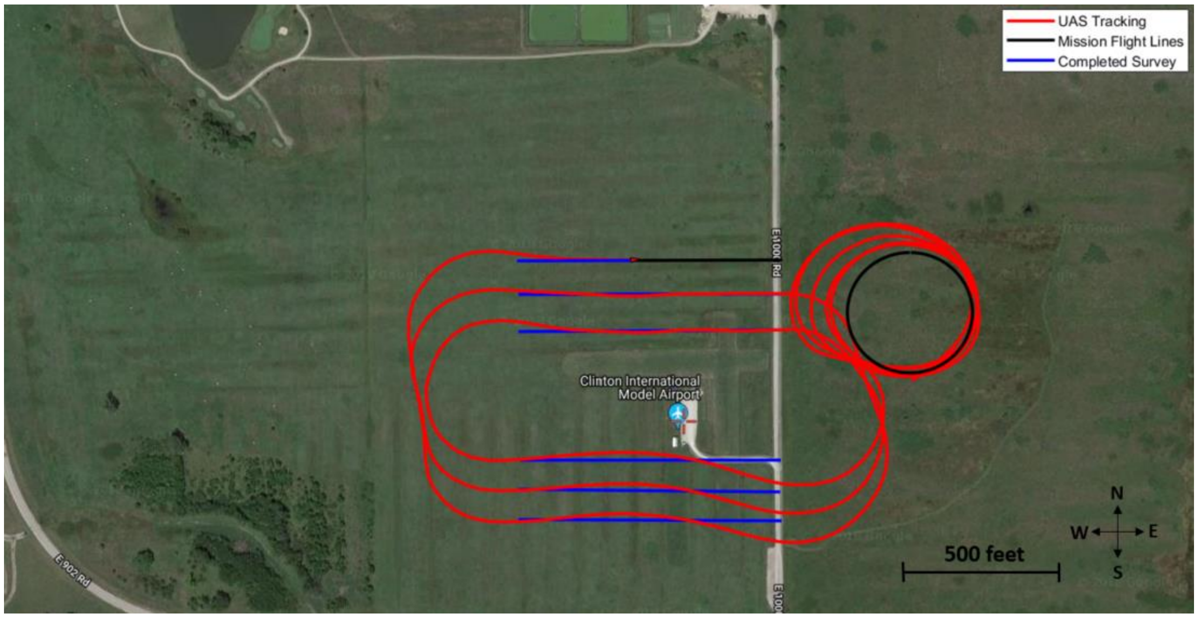


Figure 109: Flight 1 Tracking, t = 185 seconds

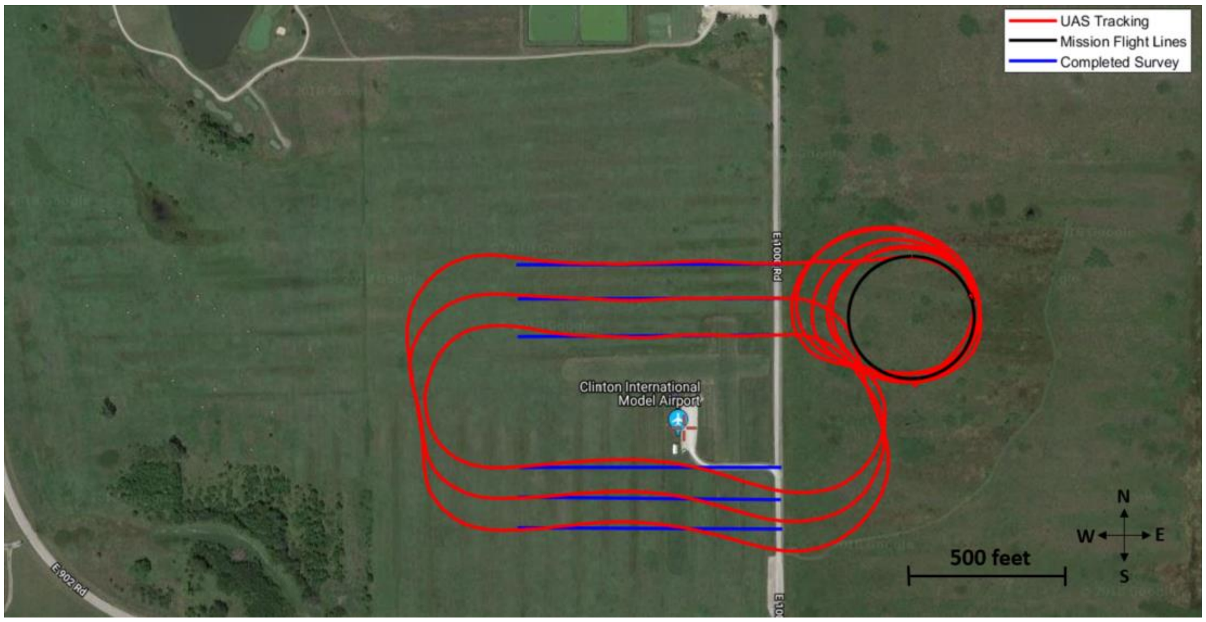


Figure 110: Flight 1 Final Tracking

The overall performance of the flight controller with respect to the commanded roll angle by the developed autonomy is shown in Figure 111, where it can be observed that while the controller follows the command trends well, noise and steady state errors exist, which can result in degraded tracking performance. Note that these errors can be the result of a variety of factors other than the flight controller, such as external disturbances and un-modeled dynamics such as aeroelastic effects on the foam airframe.

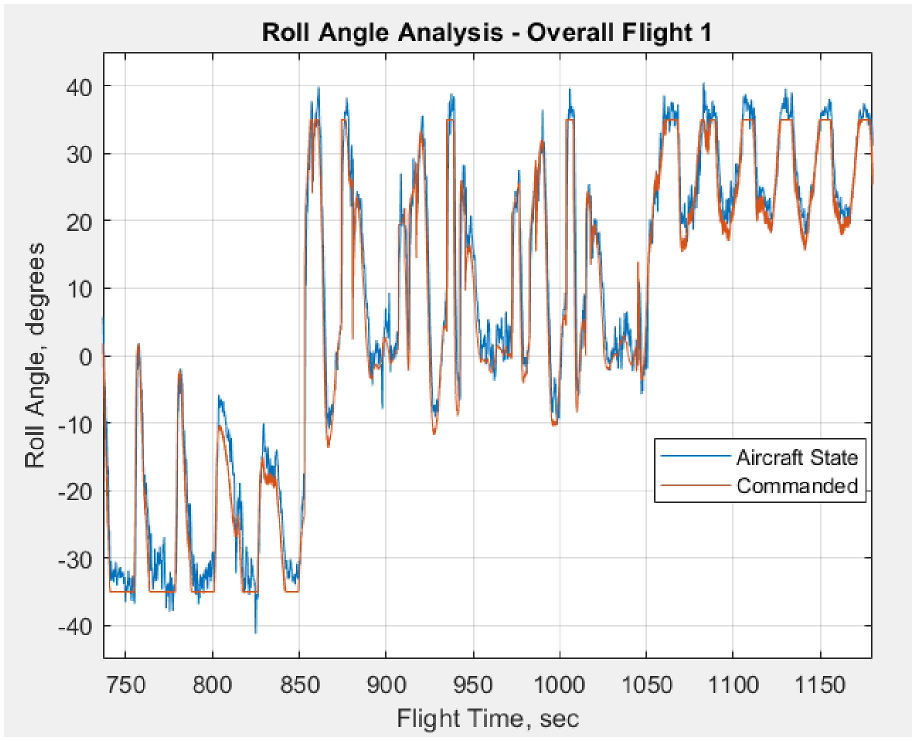


Figure 111: Flight 1, Overall Roll Angle Tracking

In Figure 112, the tracking performance of the aircraft with respect to the survey flight lines is shown, with the start of new flight lines depicted as vertical partitions on the graph. Note that the difference in time to survey for flight lines 1, 3, and 5 compared to line 2, 4, and 6 are directly a function of the tailwinds and headwinds, respectively. Also note the data in this figure corresponds only to the off-track position error of the aircraft when it is between the two endpoints for each respective flight line, and does not include the portions of flight where the aircraft is approaching the flight lines. Figure 113 shows the aircraft roll angle during the flight line surveying, while Figure 114 shows the heading error with respect to the flight line. Note that from Section 1.4.2, the ideal tracking of mission flight lines involve low off-track positional errors, along with low aircraft roll angles and low heading errors from the survey direction. Also note that the errors between the commanded roll angles from the developed autonomy and the achieved roll angels from the flight controller shown in Figure 113 directly contribute to positional tracking errors with respect to the flight line.

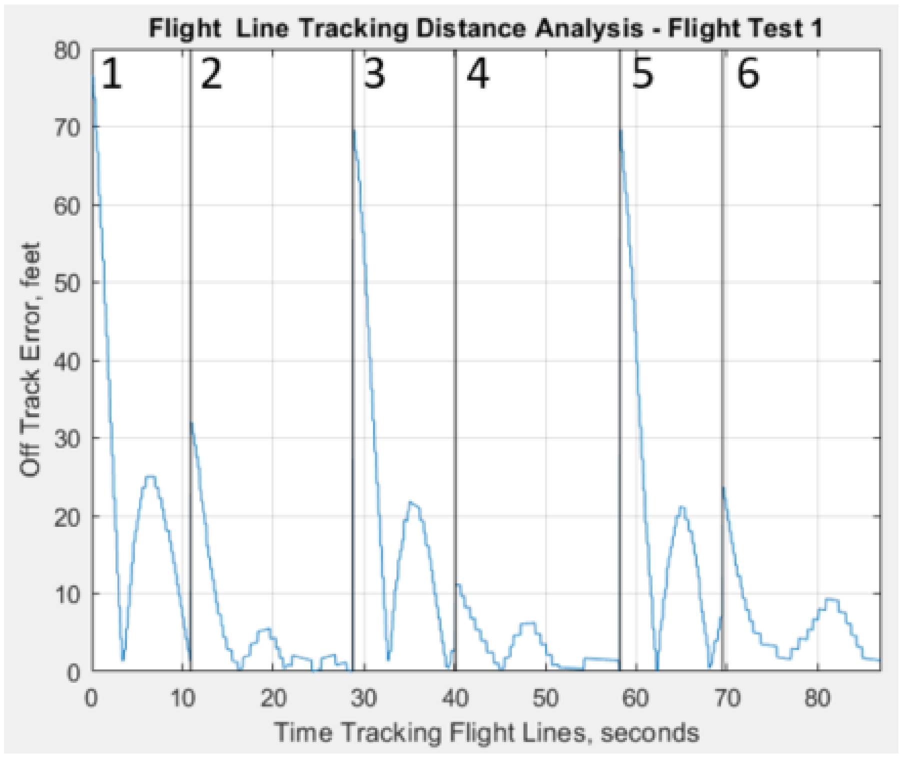


Figure 112: Flight 1, Off-Track Distance Analysis during Flight Line Tracking

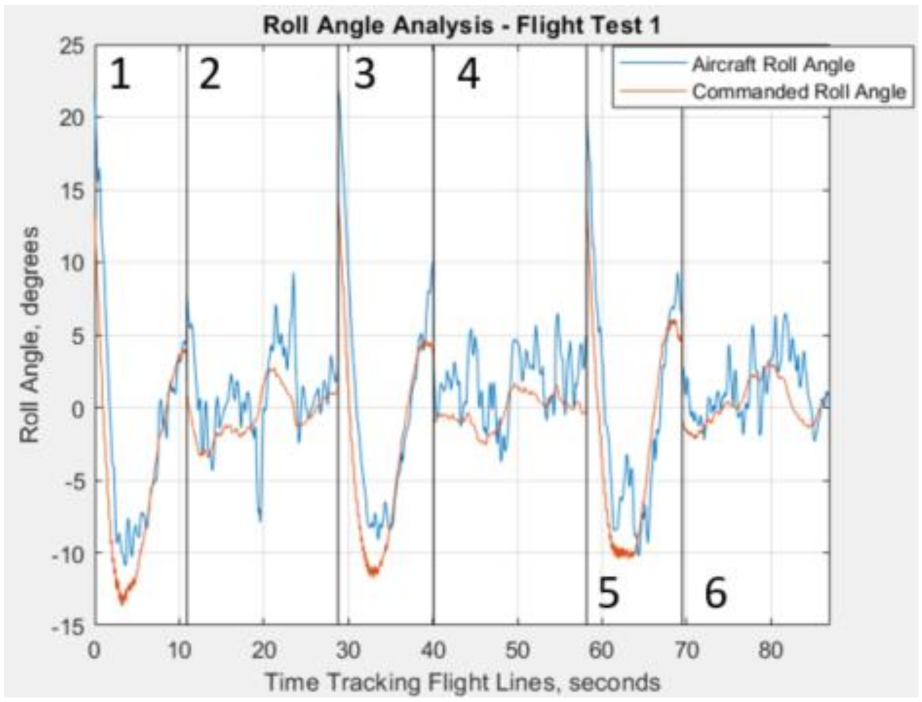


Figure 113: Flight 1, Roll Angle Tracking Analysis during Flight Line Tracking

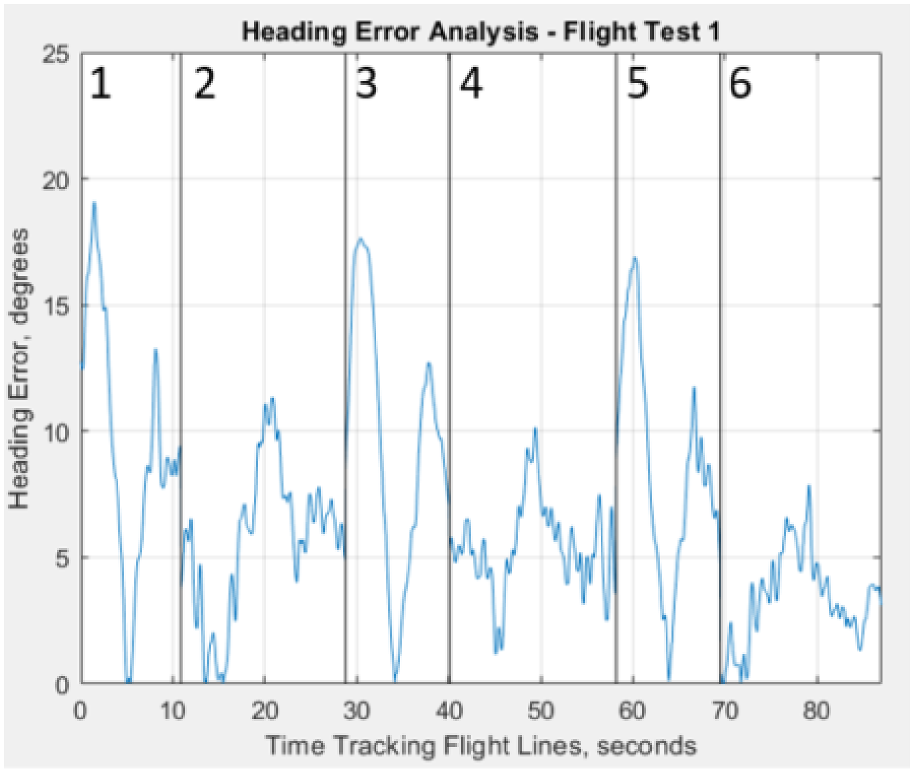


Figure 114: Flight 1, Heading Error Analysis during Flight Line Tracking

Table 16 depicts the overall survey results for the racetrack mission, while Table 17 depicts the tracking performance with respect to each survey flight line. In this scenario, while the developed autonomy was successfully able to create an efficient route through the mission flight lines and generate optimal approaches using Dubins Path guidance methods, significant improvement can be made in the tracking performance for the aircraft with respect to the mission flight lines. However, the performance of the northern lines varied dramatically from the southern lines, indicating the effect of the wind on the aircraft tracking. The aircraft's best survey was for Flight Line 4, with a maximum off-track error of 11.11 feet for the entire end-to-end coverage of the flight line.

Table 16: Flight 1, Overall Survey Results

Total Deployment Time	Max Distance from Flight Line	RMS Tracking Error from Flight Line	Max Roll Angle	Max Heading Error
198 seconds	76.49 feet	18.75 feet	21.88 degrees	19.09 degrees

Table 17: Flight 1, Flight Line Survey Analysis

Flight Line Number	Max Distance from Line (feet)	RMS Tracking Error from Flight Line (feet)	Max Roll Angle (degrees)	Max Heading Error (degrees)
1	76.49	23.98	21.58	19.09
2	31.96	9.34	9.27	11.34
3	69.55	22.66	21.88	17.64
4	11.11	4.35	6.50	10.15
5	69.57	22.14	20.36	16.89
6	23.64	7.97	6.45	7.89

5.3 Tightly-Spaced Surveying Mission

In a second flight testing scenario, four parallel flight lines are positioned with 7 ft spacing in order to assess the capabilities of the system for survey operations required for cross-track synthetic aperture processes, as described in Section 1.4.2. Similar to the previous scenario, the Forward Greedy TSP method will be used for real-time route construction through the mission flight lines, while Dubins Paths will be utilized for generating optimal approaches onto the flight lines. This flight testing scenario is shown in Figure 115.

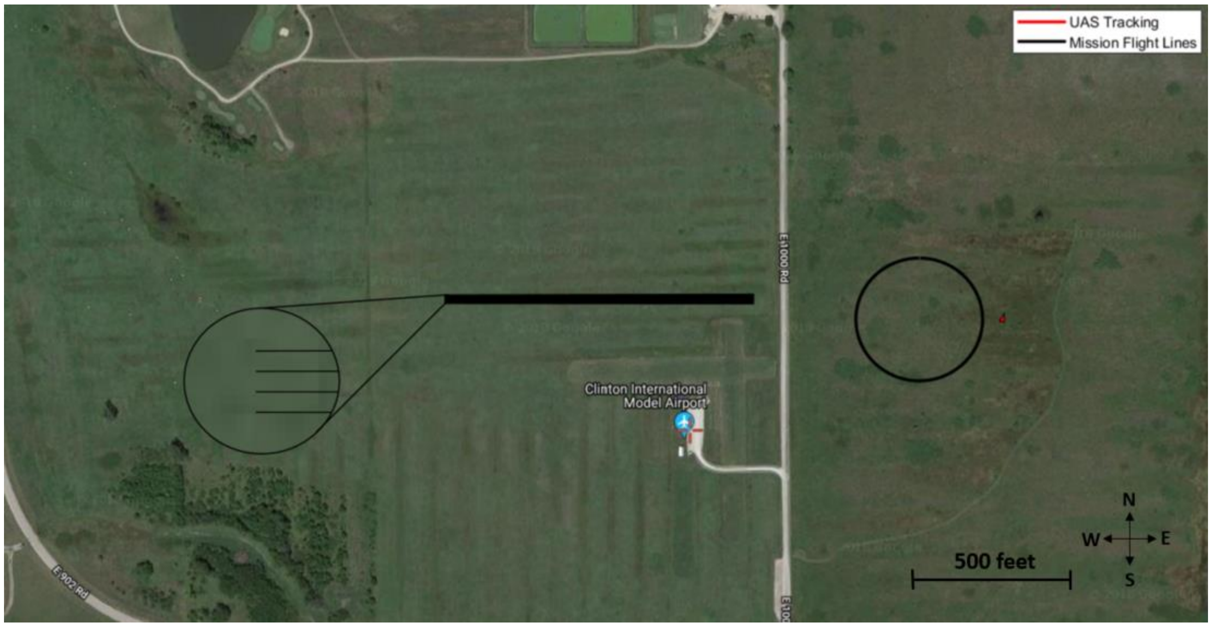


Figure 115: Flight 2, Mission Scenario

Figure 116 depicts the instant at which the deploy signal is sent to the aircraft, where the tracking of the aircraft about the home loiter can be noted. Figure 117 shows the tracking of the aircraft onto the first flight line, while Figure 118 shows the tracking of the aircraft onto the second flight line in the mission. Note that the aircraft utilizes the Dubins Path “Type 5” and “Type 6” architectures outlined in Section 2.2.2, which are the optimal path when the aircraft needs to approach a nearby position but with a 180 degree heading change. Figure 119 and Figure 120 show the tracking of the aircraft onto the third and fourth flight line, respectively, while Figure 121 shows the aircraft return to a loiter about the home position, as well as the overall tracking in the survey mission.

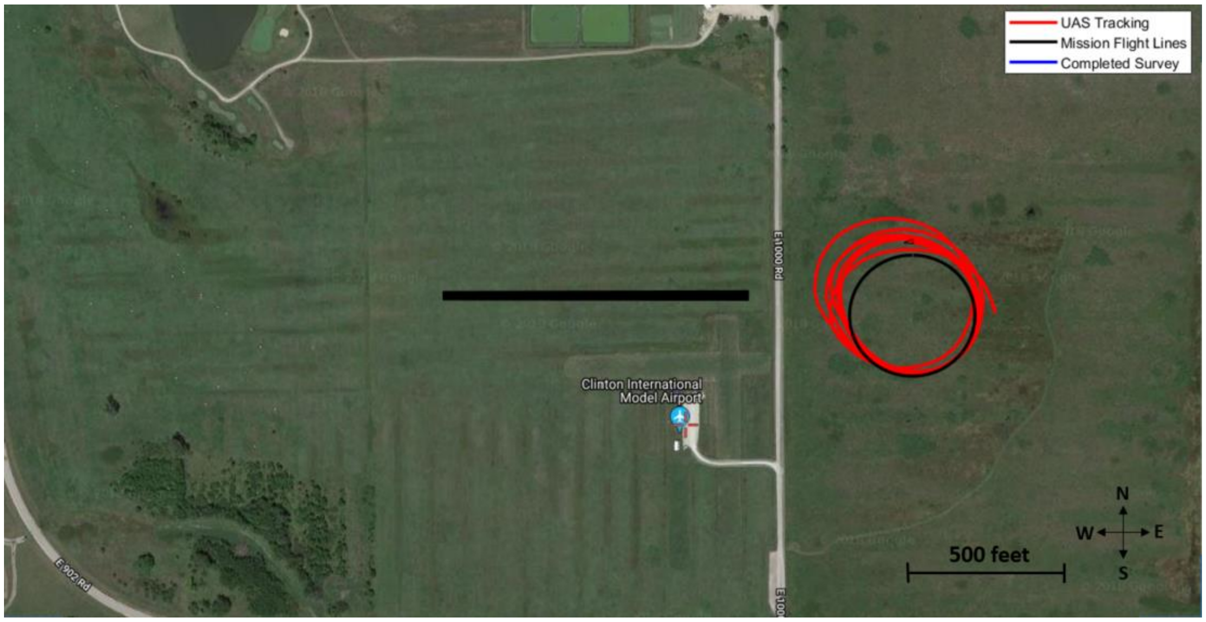


Figure 116: Flight 2 Tracking, t = 0 seconds after deployment

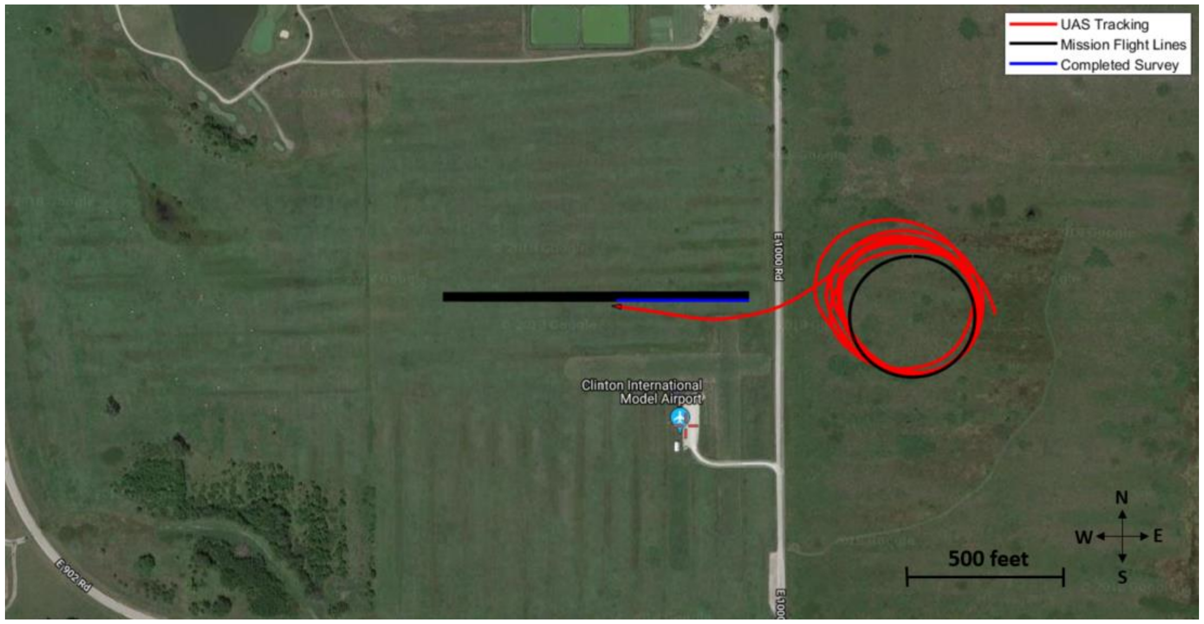


Figure 117: Flight 2 Tracking, t = 14 seconds after deployment

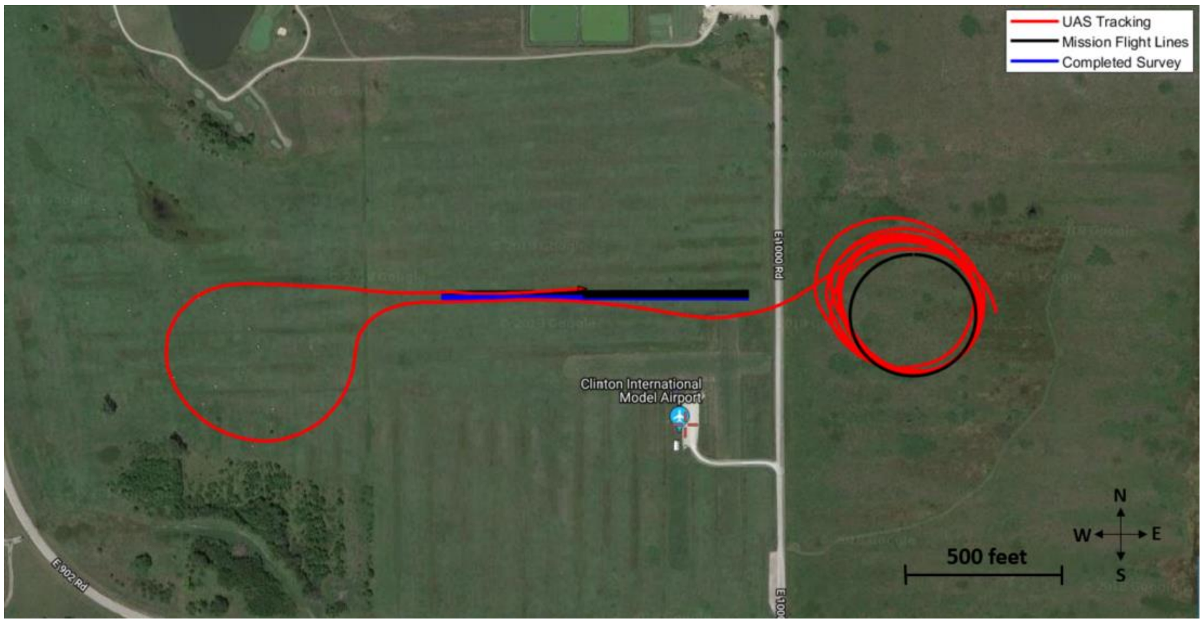


Figure 118: Flight 2 Tracking, t = 68 seconds after deployment

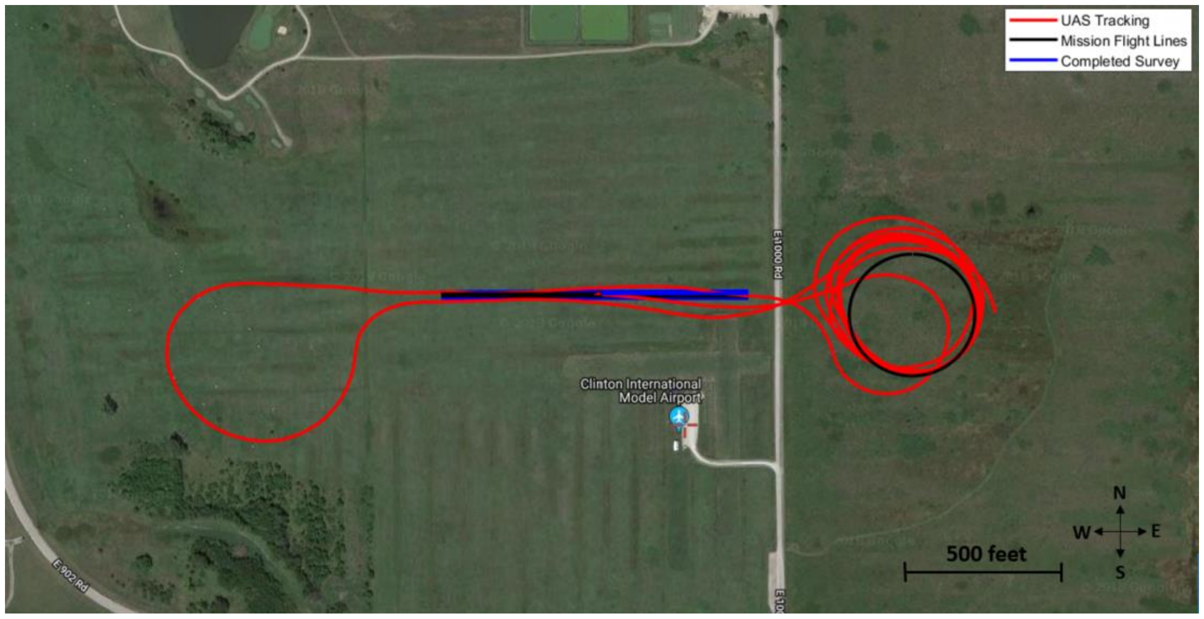


Figure 119: Flight 2 Tracking, t = 113 seconds after deployment

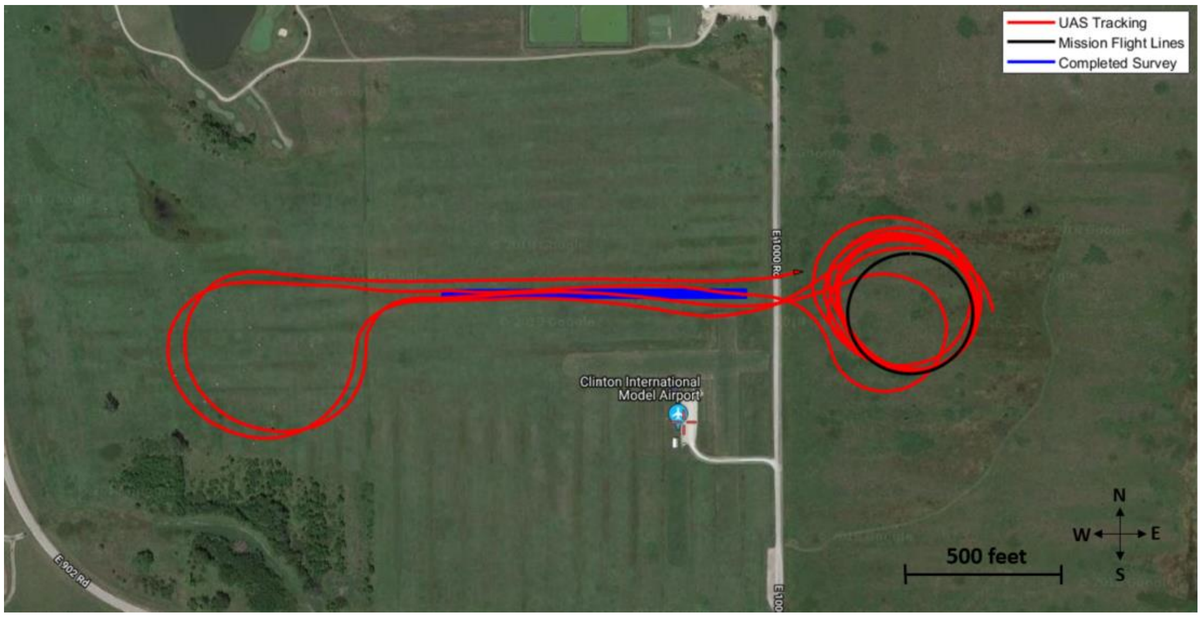


Figure 120: Flight 2 Tracking, t = 179 seconds after deployment

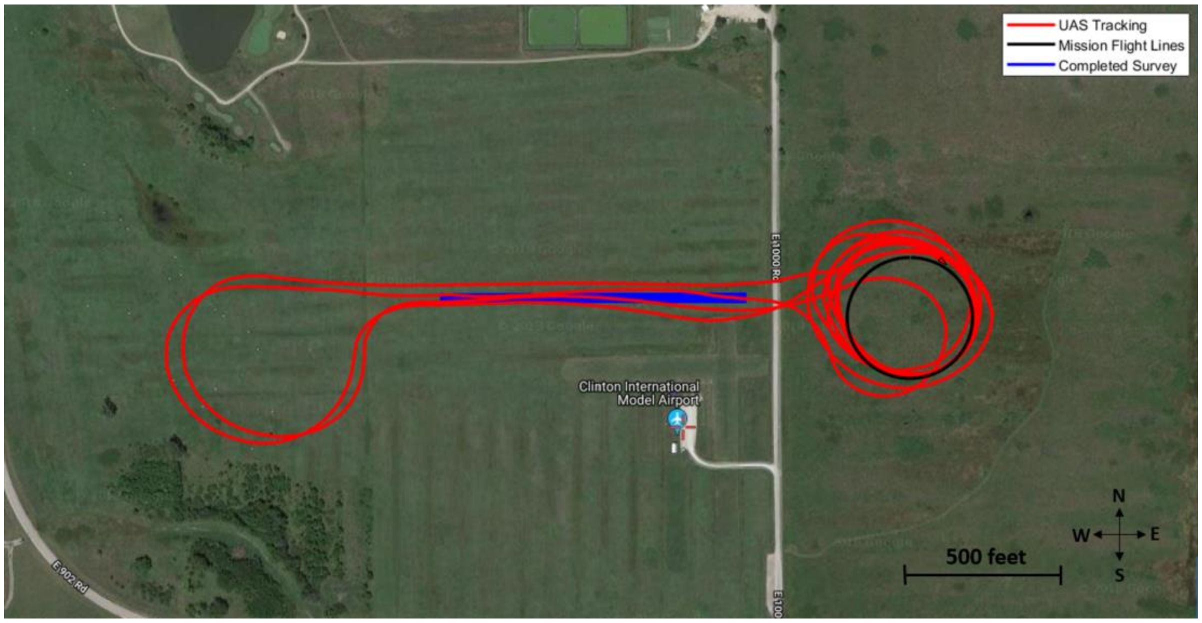


Figure 121: Flight 2 Final Tracking

Figure 122 shows the overall tracking performance of the flight controller with respect to the commanded roll angle from the developed autonomy, where it can again be noted that oscillations and steady state errors will degrade from the desired tracking performance. Again, in addition to the flight controller, wind disturbances and aeroelastic characteristics of the small UAS can contribute to these observed errors.

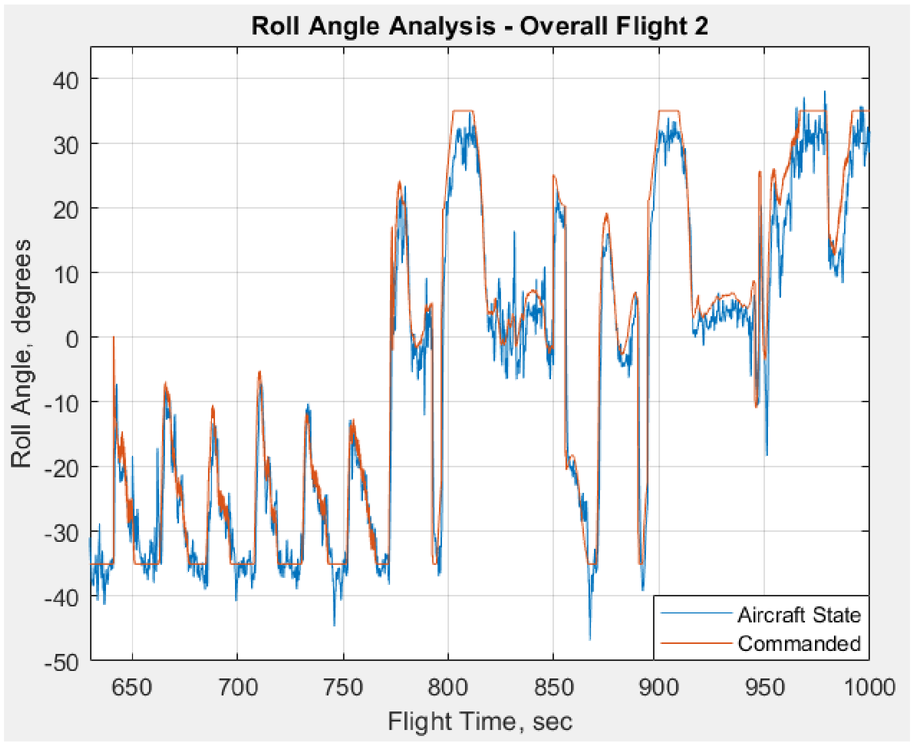


Figure 122: Flight 2, Overall Roll Angle Tracking

Figure 123 shows the off-track positional errors for the aircraft during the end-to-end survey of the mission flight lines, with the start of new flight lines depicted as vertical partitions on the graph. Figure 124 and Figure 125 depict the aircraft roll angle and aircraft heading errors during these flight portions, respectively. Again, note that the errors between the commanded and actual roll angles in Figure 124 directly affect the off-track positional error of the aircraft during the flight line survey. Notably for Flight Line 4, a steady state positional error of ~35 feet occurs along the stretch of the flight line, which corresponds to a steady state error between the developed autonomy command and the achieved aircraft roll angle.

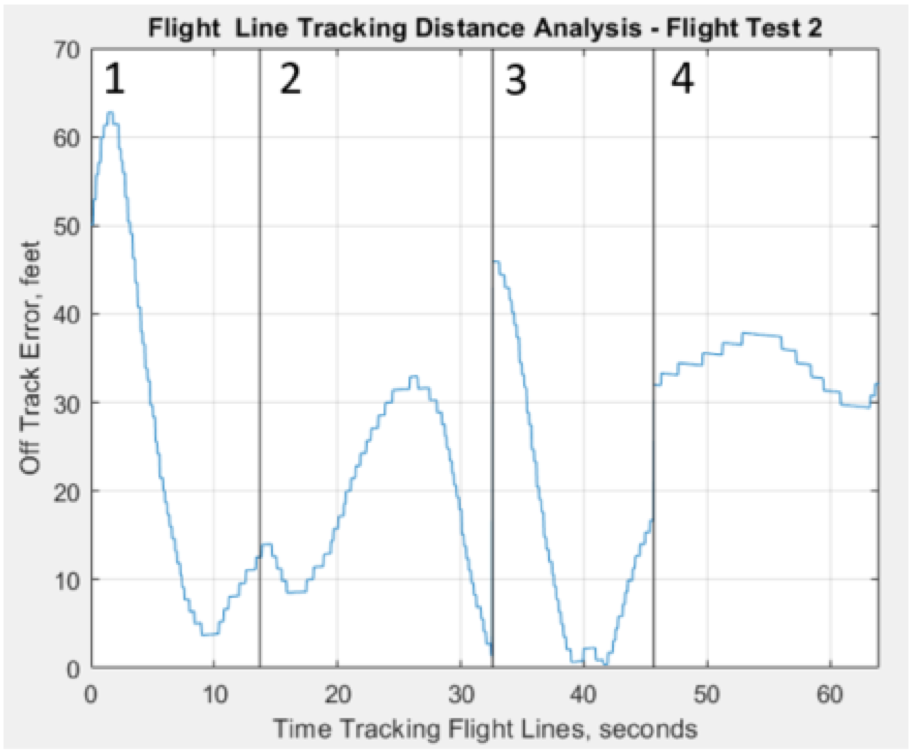


Figure 123: Flight 2, Off-Track Distance Analysis during Flight Line Tracking

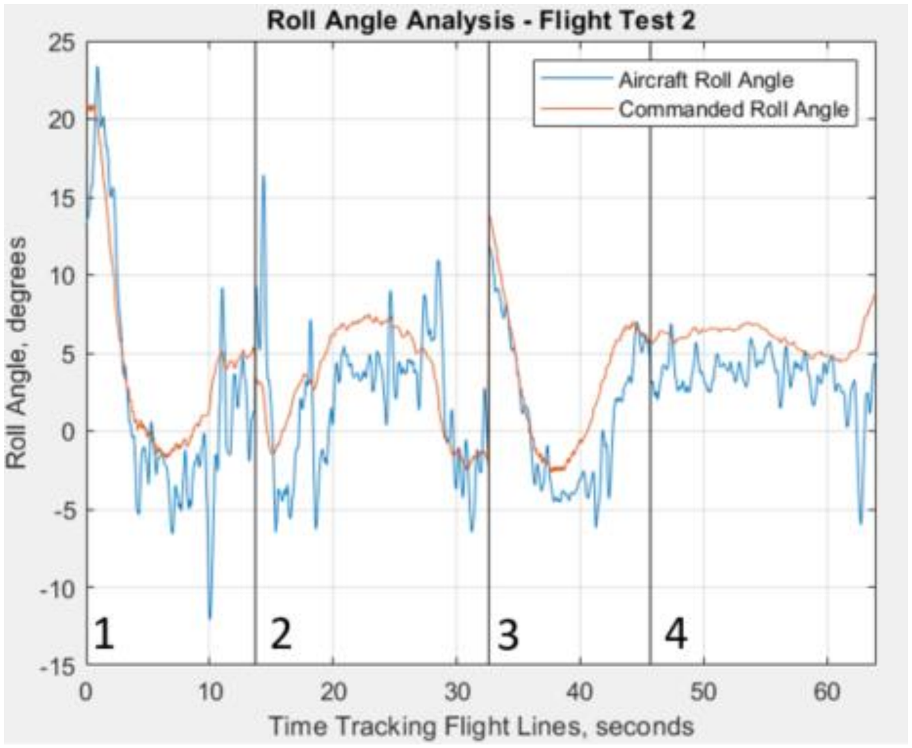


Figure 124: Flight 2, Roll Angle Tracking Analysis during Flight Line Tracking

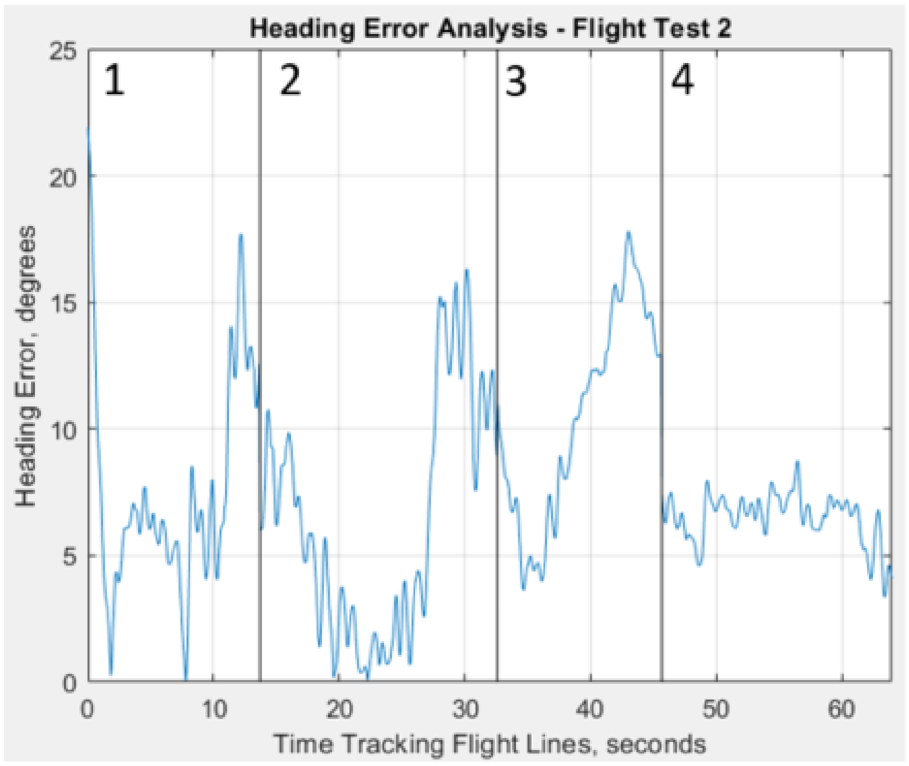


Figure 125: Flight 2, Heading Error Analysis during Flight Line Tracking

Table 18 shows the overall survey results for this mission scenario, while Table 19 shows the tracking results for each individual flight line in the mission. In this mission scenario, while the aircraft was able to successfully achieve the desired maneuvers between the flight lines, significant improvement can be made in terms of the off-track error during the tracking of the flight lines themselves.

Table 18: Flight 2, Overall Survey Results

Total Deployment Time	Max Distance from Flight Line	RMS Tracking Error from Flight Line	Max Roll Angle	Max Heading Error
176 seconds	62.76 feet	28.22 feet	23.39 degrees	21.88 degrees

Table 19: Flight 2, Flight Line Survey Analysis

Flight Line Number	Max Distance from Line (feet)	RMS Tracking Error from Flight Line (feet)	Max Roll Angle (degrees)	Max Heading Error (degrees)
1	62.76	32.84	23.39	21.88
2	32.93	24.95	16.44	16.32
3	45.88	21.06	11.83	17.81
4	37.88	40.14	-6.89	8.77

5.4 Mission Planning for Manned Operations

Additional validation for the developed autonomy was achieved during a manned deployment to the Black Hills National Park in South Dakota in February, 2020. Funded by the National Oceanic and Atmospheric Administration (NOAA), this operation featured the CReSIS Snow Radar equipped onto a Cessna-172 in order to obtain spatially continuous snow depth measurements for seasonal prediction of streamflow in hydrological models [85]. Given the fuel constraints for the aircraft, the TSP heuristics in this work were utilized as an off-board package for first determining what minimum spacing could be achieved for full coverage of the main basin of the park, as well as what optimal route should be used to survey the mission flight lines. The aircraft departed and returned from the Rapid City Regional Airport, and a strict mission duration constraint of 3 hours was applied for mission planning in order to ensure safe operations for the crew. To capture various orientations of the park while staying within this endurance constraint, missions for 3 km spacing were created, as shown in Figure 126 and Figure 128. Using the TSP heuristics in this work, trivial solutions for the mission route were created, as shown in Figure 127 and Figure 129. Subsequent flights with similar line spacing and orientations were later designed for 1.5 km offsets from the previously flown lines, in order to obtain a 1.5 km grid of the area. Finally, a more complex flight was designed that surveyed various SNOWTEL sites (where a known snow depth is measured and can be used to validate radar measurements), as well as areas of varying ground terrain and vegetation (where differences in radar measurements could be analyzed). This mission scenario is shown in Figure 130, where the TSP algorithms in this work were used to develop an efficient path, shown in Figure 131. Through this project, the developed autonomy is this work also shows promise in improving the mission efficiency for future manned CReSIS operations, by acting as an automated mission planner that incorporates aircraft fuel constraints.

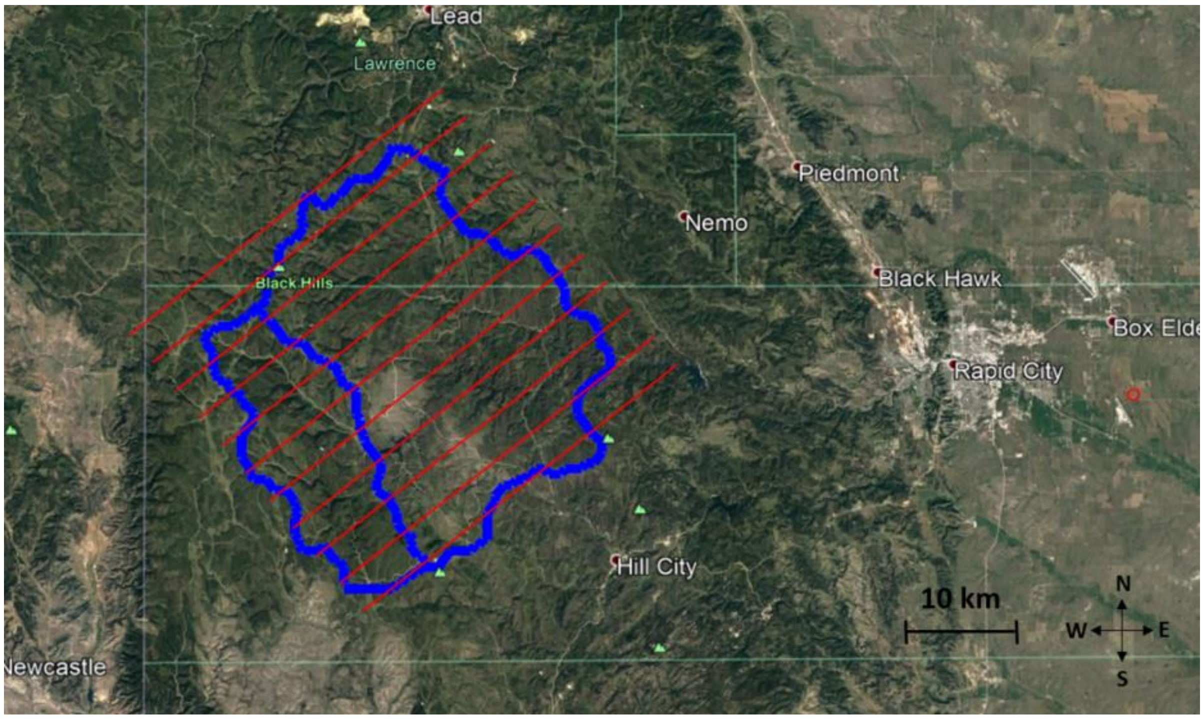


Figure 126: Black Hills Flight 1, Mission Scenario

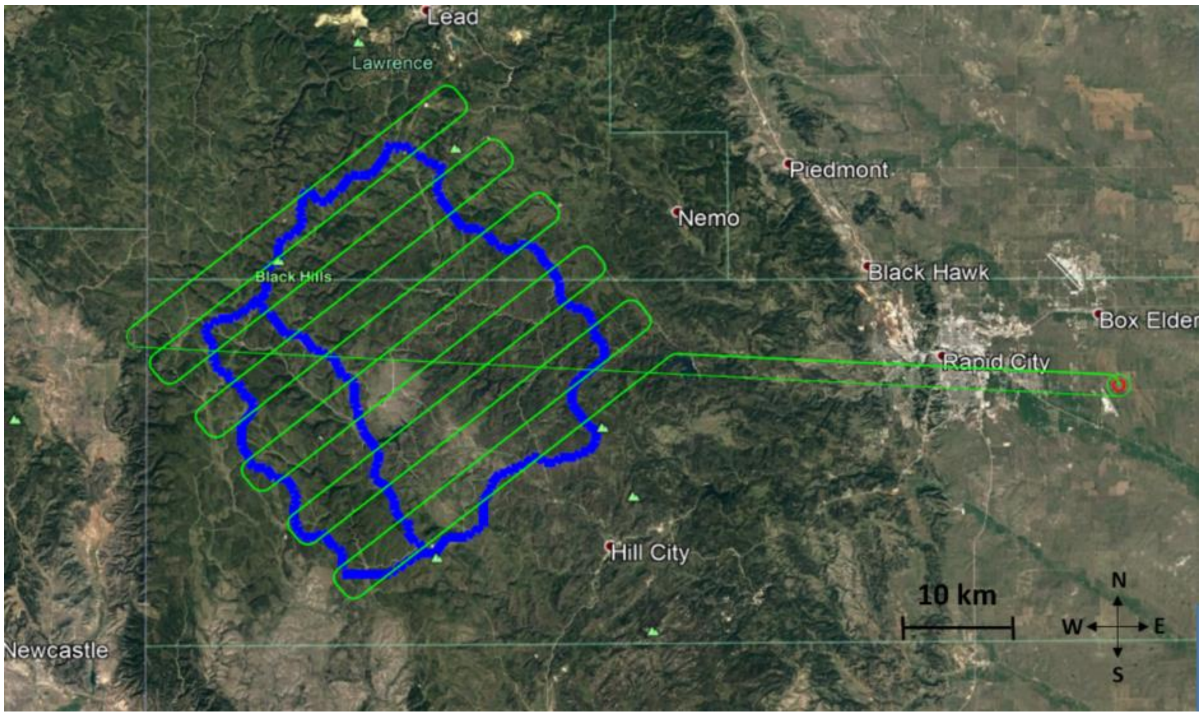


Figure 127: Black Hills Flight 1, Developed Route

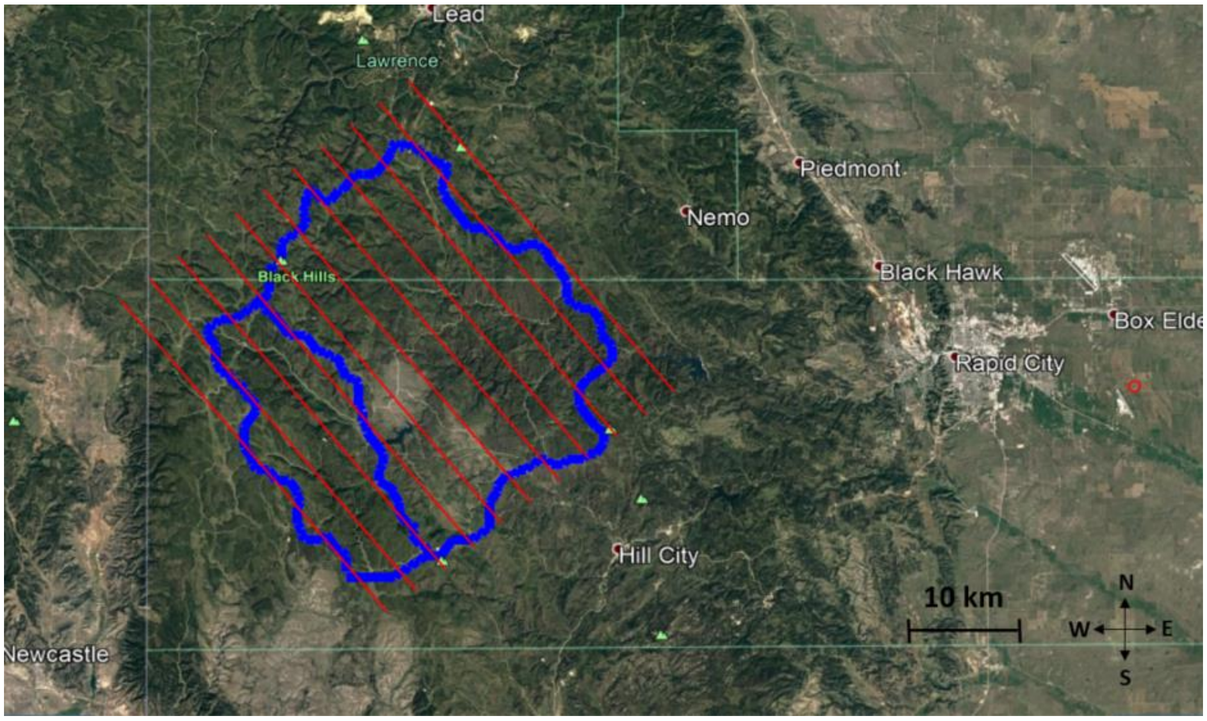


Figure 128: Black Hills Flight 2, Mission Scenario

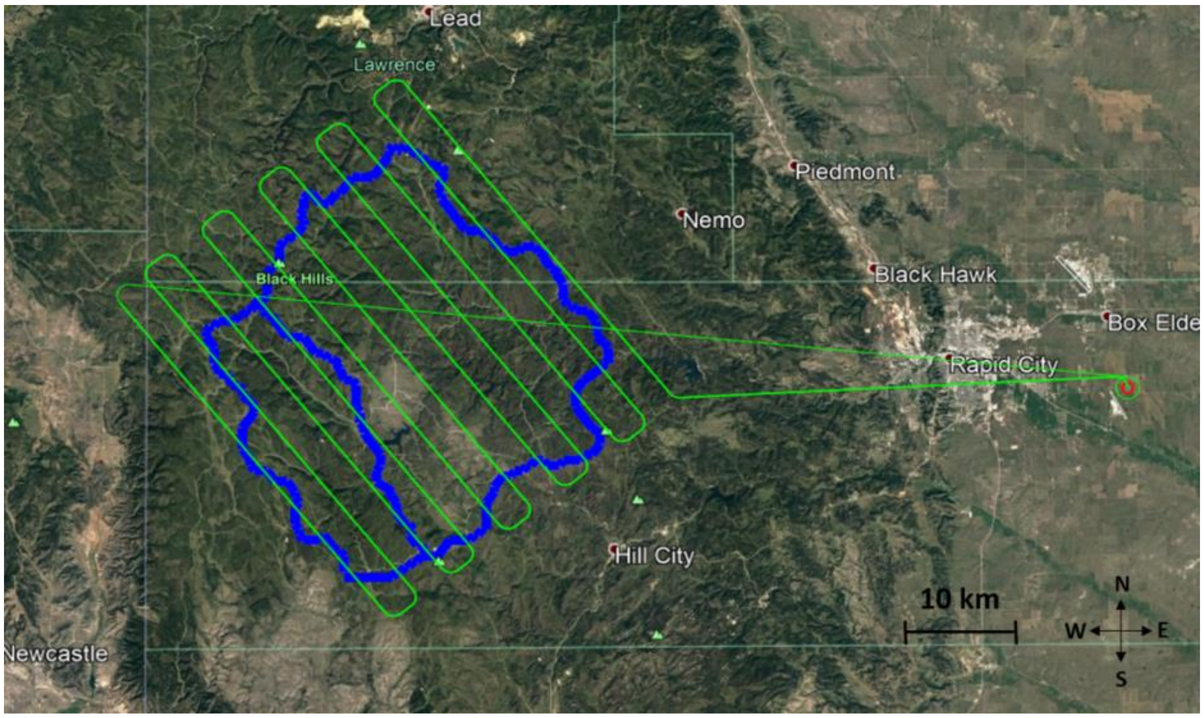


Figure 129: Black Hills Flight 2, Developed Route

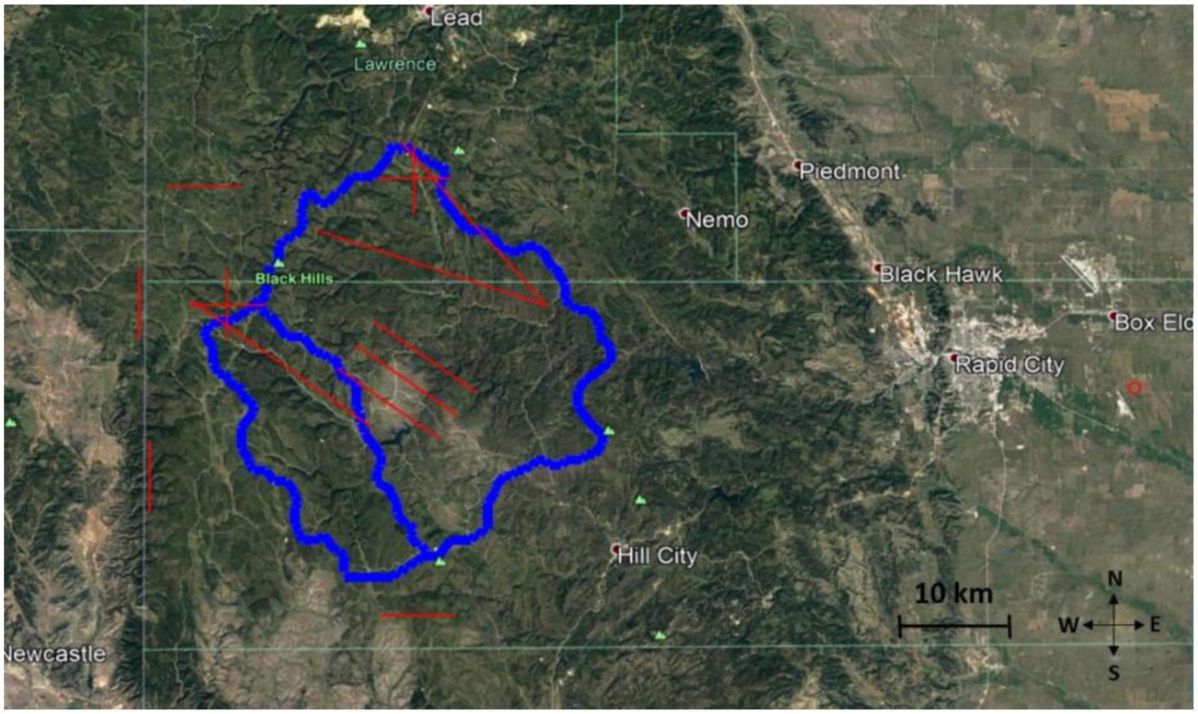


Figure 130: Black Hills Flight 3, Mission Scenario

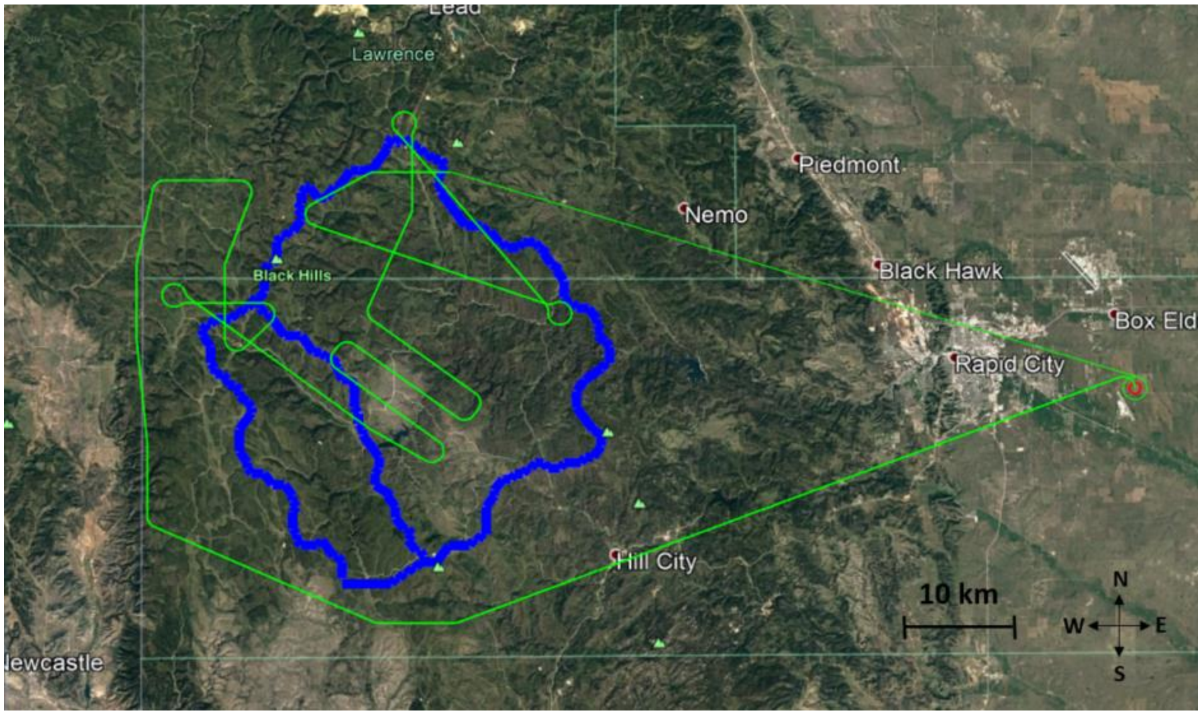


Figure 131: Black Hills Flight 3, Developed Route

Conclusion

In this work, mission-oriented autonomy was designed to improve the operational capabilities and mission efficiency of polar research operations. The motivation for polar research activities, recent technological advances in unmanned systems, and operational considerations that drive these mission-oriented autonomy features and capabilities were discussed. The features of the developed autonomy included (1) automated flight line generation, (2) optimal guidance methods, (3) cognitive real-time routing algorithms, (4) modifications for fuel-constrained surveying, (5) collaborative Multi-Agent surveying operations, (6) deployment scheduling, (7) heterogeneous capabilities, and (8) robustness contingencies for single-agent system failures. These autonomy features were developed in MATLAB and tested in Simulink environments. Case studies for past and future deployments were conducted to assess the potential improvements in mission capabilities that could result from the developed autonomy.

These case studies showed that the developed autonomy was able to improve the end-to-end tracking performance for mission flight lines compared to traditional waypoint guidance methods, was able to reduce mission times by creating efficient routes through complex mission flight lines, and was able to adapt to range constraints in order to ensure safe return to the home loiter circle in the events of infeasible mission plans. Additionally, these case studies showed how the implementation of multi-agent operations could significantly improve the operational capabilities and data collection for these polar research operations. Case studies for collaborative surveying and deployment scheduling were conducted, as well as studies demonstrating heterogeneous operations and the robustness of the autonomy design towards single agent system failures. The autonomy was then implemented onto an embedded system aboard a small UAS for preliminary flight testing, where the real-time decision making and path planning for the developed autonomy showed promising results. However, significant work remains for the flight test validation of this work, particularly for incorporating multi-agent operations. Finally, the developed autonomy was used to assist mission planning for a manned CReSIS operation.

The impact of this work is the development of a software package for future polar research deployments which includes a flexible, cognitive environment for automated and adaptive mission planning, with application for both manned or unmanned, single agent or multi-agent, and homogeneous or heterogeneous operations. Additionally, the software package is hardware agnostic, and the associated guidance can be tuned for various platform types. Through automated flight line generation and mission planning, significant workload can be alleviated from human operators, who can then transition their focus onto other mission critical tasks. While several COTS tools currently exist that can be used to assist operators for mission planning purposes, the developed autonomy has been tailor-made for polar research operations, and can be integrated onboard the autopilot systems for providing real-time, cognitive decision making that is adaptive to changing mission parameters. Additionally, as the mission objectives in this work are flight line surveys, the developed autonomy could directly be utilized for other applications that can benefit from automated mission planning and adaptive re-planning, such as crop monitoring or search and rescue operations.

References

[1] C.J. van der Veen, “Polar ice sheets and global sea level: how well can we predict the future?”, Global and Planetary Change, Volume 32, Issues 2–3, 2002, pp. 165-194, ISSN 0921-8181.

[2] Levin, Lisa. “The Deep Ocean under Climate Change.” Science, vol. 350, no. 6262, 2015, pp. 766–768., doi:10.1126/science.aad0126.

[3] Church, John A., “Sea Level Change”, Climate Change 2013: The Physical Science Basis. Contribution of Working Group I to the Fifth Assessment Report of the Intergovernmental Panel on Climate Change. pp. 1137-1216

[4] Neumann, Barbara, Athanasios T Vafeidis, Juliane Zimmermann, and Robert J Nicholls. "Future Coastal Population Growth and Exposure to Sea-level Rise and Coastal Flooding--a Global Assessment." PloS One 10.3 (2015): E0118571.

[5] Landy, Jack C., Jens K. Ehn, and David G. Barber. "Albedo Feedback Enhanced by Smoother Arctic Sea Ice." Geophysical Research Letters 42.24 (2015): 10,714-0,720.

[6] Benn, D., Evans, D., “Glaciers and Glaciation,” 2nd edition, 2010. London: Routledge

[7] Allison, I., Alley, R., Fricker, H., Thomas, R., & Warner, R. (2009). “Ice sheet mass balance and sea level”. Antarctic Science, 21(5), 413-426.

[8] Reeh, N., Mayer, C., Miller, H., Thomsen, H., and Weidick, A., “Present and past climate control on fjord glaciations in Greenland: Implications for IRD-deposition in the sea,” Geophysical Research Letters, Vol. 26, No.8, pp.1039-1042, April 15, 1999.

[9] Iverson, Neal, Denis Cohen, Thomas Hooyer, and Urs Fischer. "Effects of Basal Debris on Glacier Flow." Science 301.5629 (2003): 81-4.

[10] Kwok, R., Kurtz, N. T., Brucker, L., Ivanoff, A., Newman, T., Farrell, S. L., King, J., Howell, S., Webster, M. A., Paden, J., Leuschen, C., MacGregor, J. A., Richter-Menge, J., Harbeck, J., and Tschudi, M.: "Intercomparison of snow depth retrievals over Arctic sea ice from radar data acquired by Operation IceBridge", The Cryosphere, 11, 2571-2593.

[11] Rodriguez-Morales, F., Gogineni, S., Leuschen, C., et al., "Advanced Multifrequency Radar Instrumentation for Polar Research," IEEE Transactions on Geoscience and Remote Sensing. 52.

[12] Watts, A.C.; Ambrosia, V.G.; Hinkley, E.A. "Unmanned Aircraft Systems in Remote Sensing and Scientific Research: Classification and Considerations of Use." Remote Sens. 2012, 4, 1671-1692.

[13] Bejiga, M.B.; Zeggada, A.; Nouffidj, A.; Melgani, F., "A Convolutional Neural Network Approach for Assisting Avalanche Search and Rescue Operations with UAV Imagery." Remote Sens. 2017, 9, 100.

[14] Kachamba, Daud & Ørka, Hans & Gobakken, Terje & Eid, Tron & Mwase, Weston. "Biomass Estimation Using 3D Data from Unmanned Aerial Vehicle Imagery in a Tropical Woodland." Remote Sensing. 8. 10.3390/rs8110968, (2016).

[15] Gindraux, S.; Boesch, R.; Farinotti, D. "Accuracy Assessment of Digital Surface Models from Unmanned Aerial Vehicles' Imagery on Glaciers.“ Remote Sens. 2017, 9, 186.

[16] Efron, Shira, "The Use of Unmanned Aerial Systems for Agriculture in Africa: Can It Fly?". Santa Monica, CA: RAND Corporation, 2015.

[17] Shakhatreh, Hazim et al. "Unmanned Aerial Vehicles: A Survey on Civil Applications and Key Research Challenges." CoRR abs/1805.00881 (2018)

[18] Steven Wegener, Susan Schoenung, Joe Totah, Don Sullivan, J. Frank, Francis Enomoto, C. Frost, and Colin Theodore. "UAV Autonomous Operations for Airborne Science Missions", AIAA 3rd "Unmanned Unlimited" Technical Conference, Workshop and Exhibit, 2014.

[19] Podgorny, I, Lubin D, Perovich DK., "Monte Carlo study of UAV-measurable albedo over Arctic Sea ice." Journal of Atmospheric and Oceanic Technology. 35:57-66., 2018.

[20] Cai, Guowei et al. "A Survey of Small-Scale Unmanned Aerial Vehicles : Recent Advances and Future Development Trends." (2014).

[21] Hale, R., Donovan, W., Ewing, M., Siegele, K., Jager, R., Leong, E., Liu, W., "The Meridian UAS: Detailed Design Review," CReSIS Technical Report, CReSIS TR 124, June 2007.

[22] Tom, Jonathan. "Planning, Execution, and Analysis of the Meridian UAS Flight Test Program Including System and Parameter Identification". Master's Thesis, , University of the Kansas, Lawrence, KS, USA, 2010.

[23] Mahmood, A., "Design, Integration, and Deployment of UAS borne HF/VHF Depth Sounding Radar and Antenna System." Master's Thesis, University of the Kansas, Lawrence, KS, USA, 2017.

[24] Leuschen, C., Hale, R., Keshmiri, S., et al. (2014). "UAS-Based Radar Sounding of the Polar Ice Sheets." Geoscience and Remote Sensing Magazine, IEEE. 2. 8-17. 10.1109/MGRS.2014.2306353.

[25] Keshmiri, S., Ewing, M., Hale, R., et al., "Multi-Agent Airborne Laboratory for Cryospheric Remote Sensing Final Report", CReSIS Technical Report, University of Kansas, Number 164, p.15 (2016).

[26] Keshmiri, S., Arnold, E., & Blevins, A., et al., "Radar ECHO sounding of Russell Glacier at 35 MHz using compact radar systems on small unmanned aerial vehicles.", 2017, 5900-5903. 10.1109/IGARSS.2017.8128352.

[27] Rodriguez-Morales, F., Arnold, E., Hale, R., et al., "Multi-spectral radar measurements of ice and snow using manned and unmanned aircraft". 2017, Geosciences and Remote Sensing.

[28] Rudol, P. (2011). “Increasing Autonomy of Unmanned Aircraft Systems Through the Use of Imaging Sensors” (Licentiate dissertation). Linköping University Electronic Press, Linköping.

[29] Genova, Daniel, Kong, Zhaodan, Hess, Ron, and Lin, Xinfan. “Optimized UAV Trajectory Tracking of a Dynamically Unknown Ground Target” (2019): PQDT - Global.

[30] Panella, I. "Artificial Intelligence Methodologies Applicable to Support the Decision-Making Capability on Board Unmanned Aerial Vehicles." 2008 Bio-inspired, Learning and Intelligent Systems for Security (2008): 111-18.

[31] Schneider, Toby. (2013). “Advances in integrating autonomy with acoustic communications for intelligent networks of marine robots”.

[32] J. Weaver and D. Dunlap, "Hell bay trials: A multinational collaborative marine autonomy field experimentation program," OCEANS 2016 MTS/IEEE Monterey, Monterey, CA, 2016, pp. 1-7, doi: 10.1109/OCEANS.2016.7761387.

[33] Ono, M., Fuchs, Steffy, Maimone, and Jeng Yen. "Risk-aware Planetary Rover Operation: Autonomous Terrain Classification and Path Planning." 2015 IEEE Aerospace Conference 2015 (2015): 1-10.

[34] Peters, Matthew E. et al. “Along-Track Focusing of Airborne Radar Sounding Data From West Antarctica for Improving Basal Reflection Analysis and Layer Detection.” IEEE Transactions on Geoscience and Remote Sensing 45 (2007): 2725-2736.

[35] Wu, Xiaoqing et al. “Ice Sheet Bed Mapping With Airborne SAR Tomography.” IEEE Transactions on Geoscience and Remote Sensing 49 (2011): 3791-3802.

[36] Braaten, D., Gogineni, S., Tammana, D., Namburi, S., Paden, J., & Gurumoorthy, K. (2002). “Improvement of radar ice-thickness measurements of Greenland outlet glaciers using SAR processing.” Annals of Glaciology, 35, 73-78.

[37] Smith, L. “Validation of CReSIS Synthetic Aperture Radar Processor and Optimal Processing Parameters”. Master’s Thesis, , University of the Kansas, Lawrence, KS, USA, 2014.

[38] Lin, Chung-Chi et al. “Surface Clutter Suppression Techniques Applied to P-band Multi-Channel SAR Ice Sounder Data from East Antarctica.” (2012).

[39] Vincent, J., and Arnold, E., “Beamforming sensitivity of airborne distributed arrays to flight tracking and vehicle dynamics.” 2017 IEEE Aerospace Conference (2017): 1-14.

[40] Dubins, L. E. “On Curves of Minimal Length with a Constraint on Average Curvature, and with Prescribed Initial and Terminal Positions and Tangents.” American Journal of Mathematics, vol. 79, no. 3, 1957, pp. 497–516. JSTOR

[41] Reeds, J. A.; Shepp, L. A. “Optimal paths for a car that goes both forwards and backwards”. Pacific J. Math. 145 (1990), no. 2, 367--393.

[42] Y. Lin and S. Saripalli, "Path planning using 3D Dubins Curve for Unmanned Aerial Vehicles," 2014 International Conference on Unmanned Aircraft Systems (ICUAS), Orlando, FL, 2014, pp. 296-304, doi: 10.1109/ICUAS.2014.6842268.

[43] Wang X, Jiang P, Li D, Sun T. “Curvature Continuous and Bounded Path Planning for Fixed-Wing UAVs”. Sensors (Basel). 2017;17(9):2155. Published 2017 Sep 19.

[44] Blevins. A, et al, "Dubins Path Guidance for Fixed-Wing UAS Remote Sensing Applications", 2020 UAS Guidance, Navigation, and Control Conference, AIAA SciTech Forum, (AIAA 2020-1236).

[45] A. Kaplan, P. Uhing, N. Kingry and R. D. Adam, "Integrated path planning and power management for solar-powered unmanned ground vehicles," 2015 IEEE International Conference on Robotics and Automation (ICRA), Seattle, WA, 2015, pp. 982-987, doi: 10.1109/ICRA.2015.7139296.

[46] Howard, Thomas & Pivtoraiko, Mihail & Knepper, Ross & Kelly, Alonzo. (2014). “Model-Predictive Motion Planning: Several Key Developments for Autonomous Mobile Robots”. Robotics & Automation Magazine, IEEE. 21. 64-73. 10.1109/MRA.2013.2294914.

[47] Korayem, M. & Nazemizadeh, Mostafa & Azimirad, Vahid. (2011). “Optimal trajectory planning of wheeled mobile manipulators in cluttered environments using potential functions”. Scientia Iranica. 18. 1138–1147. 10.1016/j.scient.2011.08.026.

[48] Karapetyan, Nare & Moulton, Jason & Lewis, Jeremy & Quattrini Li, Alberto & Okane, Jason & Rekleitis, Ioannis. (2018). “Multi-robot Dubins Coverage with Autonomous Surface Vehicles”. 2373-2379. 10.1109/ICRA.2018.8460661.

[49] Cai, Wenyu & Zhang, Meiyan & Zheng, Rosa. (2017). “Task Assignment and Path Planning for Multiple Autonomous Underwater Vehicles Using 3D Dubins Curves”. Sensors. 17. 1607. 10.3390/s17071607.

[50] BEARD, RANDAL W., and TIMOTHY W. McLAIN. “Small Unmanned Aircraft: Theory and Practice”. Princeton University Press, 2012.

[51] Matai, Rajesh et al. “Traveling Salesman Problem : An Overview of Applications , Formulations , and Solution Approaches.” (2012).

[52] Johnson, David & A. McGeoch, Lyle. “The Traveling Salesman Problem: A Case Study in Local Optimization”. Local Search in Combinatorial Optimization, (1997).

[53] Chauhan, C., Gupta, R., Pathak, K., “Survey of Methods of Solving TSP along with its Implementation using Dynamic Programming Approach”. International Journal of Computer Applications. 52. 12-19, (2012).

[54] Rego, César, Dorabela Gamboa, Fred Glover, and Colin Osterman. "Traveling Salesman Problem Heuristics: Leading Methods, Implementations and Latest Advances." European Journal of Operational Research 211.3 (2011): 427-41.

[55] Potvin, Jy. "Genetic Algorithms for the Traveling Salesman Problem." Annals Of Operations Research 63 (1996): 339-70.

[56] Basu, Sumanta. "Tabu Search Implementation on Traveling Salesman Problem and Its Variations: A Literature Survey." American Journal of Operations Research 2.2 (2012): 163-73.

[57] Dorigo, Marco & Birattari, Mauro & Blum, Christian & Clerc, M & Stützle, Thomas & Winfield, Alan & Bautista-Valhondo, Joaquín. (2008). "Ant Colony Optimization and Swarm Intelligence: 6th International Conference", ANTS 2008. 10.1007/978-3-540-87527-7.

[58] Dorigo, Marco & Birattari, Mauro & Stützle, Thomas. (2006). "Ant Colony Optimization". Computational Intelligence Magazine, IEEE. 1. 28-39. 10.1109/MCI.2006.329691.

[59] A. Agarwal, Meng-Hiot Lim, Meng-Joo Er and Chan Yee Chew, "ACO for a new TSP in region coverage," 2005 IEEE/RSJ International Conference on Intelligent Robots and Systems, Edmonton, Alta., 2005, pp. 1717-1722, doi: 10.1109/IROS.2005.1545460.

[60] Anuntachai, A., Wongwirat, O. & Thammano, A. "An application of ant algorithm for searching energy-efficient route a mobile robot takes using energy as a weighting factor". Artif Life Robotics 19, 354–362 (2014). https://doi.org/10.1007/s10015-014-0175-8

[61] Chen, Feiyang & Chen, Nan & Mao, Hanyang & Hu, Hanlin. (2019). "The Application of Bipartite Matching in Assignment Problem".

[62] S. Chopra, G. Notarstefano, M. Rice and M. Egerstedt, "A Distributed Version of the Hungarian Method for Multirobot Assignment," in IEEE Transactions on Robotics, vol. 33, no. 4, pp. 932-947, Aug. 2017, doi: 10.1109/TRO.2017.2693377.

[63] Ji, Meng & Azuma, S. & Egerstedt, M.. (2006). “Role-assignment in multi-agent coordination”. International Journal of Assistive Robotics and Mechatronics. 7. 32-40.

[64] Ringbäck, Rasmus. “Multi-agent autonomous target tracking using distance-based formations.” (2017).

[65] Kuhn, H.W. (1955), “The Hungarian method for the assignment problem”. Naval Research Logistics, 2: 83-97.

[66] F. Rodriguez-Morales et al., "An Improved UWB Microwave Radar for Very Long-Range Measurements of Snow Cover," in IEEE Transactions on Instrumentation and Measurement, doi: 10.1109/TIM.2020.2982813.

[67] H. Zhang et al., "Scheduling methods for unmanned aerial vehicle based delivery systems," 2014 IEEE/AIAA 33rd Digital Avionics Systems Conference (DASC), Colorado Springs, CO, 2014, pp. 6C1-1-6C1-9, doi: 10.1109/DASC.2014.6979499.

[68] Shima, Tal & Schumacher, C.. (2009). “Assigning cooperating UAVs to simultaneous tasks on consecutive targets using genetic algorithms”. JORS. 60. 973-982. 10.1057/palgrave.jors.2602617.

[69] Khosiawan, Yohanes & Park, Young & Moon, Ilkyeong & Janardhanan, Mukund Nilakantan & Nielsen, Izabela. (2016). “Task scheduling system for UAV operations in indoor environment. Neural Computing and Applications”. 10.1007/s00521-018-3373-9.

[70] DARcorporation, “Advanced Aircraft Analysis”, Software Package, KS, 2009.

[71] Roskam, J., “Airplane Flight Dynamics and Automatic Flight Controls (part I)”, DARcorporation, Lawrence, KS, 2003.

[72] Kim, A., Shukla, D., Blevins, A., et al, “ Validation and Verification of GNC for Large Fixed-Wing Unmanned Aerial System in Unstructured and Noisy Environment”. ICUAS 2018 1388-1397.

[73] Kim, A., “Intelligent Guidance, Navigation and Control of Multi-Agent UASs with Validation and Verification”. 2018. University of Kansas, PhD Dissertation.

[74] Blevins, Aaron & Benyamen, Hady & Godfrey, Grant & Shukla, Daksh & Kim, Bella. (2018). “Analysis and Verification of Cost-Effective Design Modifications to Commercially Available Fixed-Wing Unmanned Aerial Vehicle to Improve Performance, Stability and Control Characteristics, and Structural Integrity”. 10.2514/6.2018-2261.

[75] How, J., et al. "A New Nonlinear Guidance Logic for Trajectory Tracking", AIAA Guidance, Navigation, and Control Conference and Exhibit, Guidance, Navigation, and Control and Co-located Conferences, 2004.

[76] Curry, Renwick E. et al. “L+2, an improved line of sight guidance law for UAVs.” 2013 American Control Conference (2013).

[77] Stengel, Robert F. “Flight Dynamics”. Princeton University Press, 2004.

[78] Blevins, A., Kim, A., Shukla, D., et al, “ Validation and Verification Flight Tests of Fixed-Wing Collaborative UASs with High Speeds and High Inertias”. AIAA Aviation Forum, 2018.

[79] Keshmiri, S., et al. “Control of Multi-Agent Collaborative Fixed-Wing UASs in Unstructured Environment.” 2018 International Conference on Unmanned Aircraft Systems (ICUAS) (2018): 1165-1173.

[80] Stastny, T., Gonzalo, G., and Keshmiri, S., “Collision and Obstacle Avoidance in Unmanned Aerial Systems Using Morphing Potential Field Navigation and Nonlinear Model Predictive Control.” Journal of Dynamic Systems, Measurement, and Control 2014.

[81] Stastny, T., “Collision Avoidance for Fixed-Wing Unmanned Aerial Systems Using Morphing Potential Field Navigation with Robust and Predictive Control”. Master’s Thesis, , University of the Kansas, Lawrence, KS, USA, 2014.

[82] Skukla,D., et al, "Validation and Verification Flight Testing of UAS Morphing Potential Field Collision Avoidance Algorithms", 2018 Flight Testing Conference, AIAA AVIATION Forum, (AIAA 2018-4279).

[83] Keshmiri, S., et al., “Flight Test Validation of Collision and Obstacle Avoidance in Fixed-Wing UASs with High Speeds Using Morphing Potential Field”. ICUAS 2018, 589-598.

[84] Kim, A.R., Keshmiri, S., Blevins, A. et al. “Control of Multi-Agent Collaborative Fixed-Wing UASs in Unstructured Environment”. J Intell Robot Syst 97, 205–225 (2020). https://doi.org/10.1007/s10846-019-01057-3

[85] Roundy, J. K., and Arnold, E. “Airborne Snow Depth Retrieval for Improved Hydrological Modeling in the Black Hills of South Dakota”. AMS Annual Meeting, Boston, MA. (2020, January).