Multi-drone Aerial Surveys of Penguin Colonies in Antarctica
Kunal Shah¹, Grant Ballard², Annie Schmidt², and Mac Schwager³
¹Department of Mechanical Engineering, Stanford University
²Point Blue Conservation Science
³Department of Aeronautics and Astronautics, Stanford University
*Corresponding Author: k2shah@stanford.edu

Abstract
Speed is essential in wildlife surveys due to the dynamic movement of animals throughout their environment and potentially extreme changes in weather. In this work, we present a new multi-robot path planning method for conducting aerial surveys over large areas designed to make the best use of limited flight time. Unlike current survey path planning solutions based on geometric patterns or integer programs, we solve a series of satisfiability modulo theory (SMT) instances of increasing complexity. Each instance yields a set of feasible paths at each iteration and recovers the set of shortest paths after sufficient time. We implemented our planning algorithm with a team of drones to conduct multiple photographic aerial wildlife surveys of Cape Crozier, one of the largest Adélie penguin colonies in the world containing over 300,000 nesting pairs. Over 2 km² was surveyed in about 3 hours. In contrast, previous human-piloted single-drone surveys of the same colony required over 2 days to complete. Our method reduces survey time by limiting redundant travel while also allowing for safe recall of the drones at any time during the survey. Our approach can be applied to other domains, such as wildfire surveys in high-risk weather conditions or disaster response.

Summary
A path planning algorithm enables autonomous multi-drone aerial surveys of Adélie penguin colonies in Antarctica.

Introduction
We consider the problem of planning paths for a team of aerial robots to conduct a photographic aerial survey of a given region on the ground. The robots must collect a set of images with their downward-facing cameras such that the entire area on the ground is
covered. This requires that the images be as high-resolution as possible, while ensuring there is sufficient overlap for the images to be later stitched together to create a high-resolution mosaic of the area. Furthermore, due to weather variability, lighting variability, and dynamic conditions on the ground, the aerial survey should be conducted as quickly as possible, with all robots operating simultaneously, while respecting battery life and altitude limits, as well as respecting wildlife disturbance constraints and avoiding no-fly zones in the airspace. A video overview of the project can be seen in Movie S1.

We present Path Optimization for Population Counting with Overhead Robotic Networks (POPCORN) which plans paths for a team of aerial robots to conduct aerial photographic surveys under these conditions. The core advantages of this algorithm are that it is faster than existing optimization approaches (1, 2), and that the it can be halted at any moment (after the first iteration) to return paths that satisfy the problem constraints (in this regard the algorithm can be considered to be “anytime”). Furthermore, our method is based on the highly expressive language of Satisfiability Modulo Theory (SMT), which allows us to encode requirements such as “start and end locations must coincide,” leading to cyclical paths that result in a safe recall property that is absent from current methods like sweep or wave front coverage planning (3, 4). We demonstrate our method in extensive field surveys of Adélie penguin (Pygoscelis adeliae) colonies on Ross Island, Antarctica (77.5294°S, 167.2123°E). Our survey used autonomous, multi-rotor Unpiloted Aerial Vehicles (UAVs), popularly known as "drones." Our multi-robot survey system reduced the time for an aerial survey of a ~300,000 nest Adélie penguin colony spanning over 2 km² from 2 days to 3 hours.

We formulate the problem of taking aerial surveys as a constrained path planning problem. We construct a lattice over the environment such that images taken at each point in the lattice will provide the required resolution and image overlap to produce the desired
photomosaic. Our goal is then to find minimum length paths for the UAVs through the lattice ensuring that each point in the lattice is visited at least once, while respecting battery constraints. We also require that each robot enters and exits the lattice at the same point. Our core algorithmic contribution, POPCORN, solves a series of Boolean assignment problems within an SMT framework. Each problem can be solved notably faster than a Mixed Integer Linear Program (MILP) of the same size to produce paths that satisfy the constraints (such as a maximum path length bound, or the requirement that the paths be cyclical). At each iteration, the path length bound is decreased until the problem becomes infeasible, at which point the algorithm has found the minimal length paths for the survey. Such constraints are not possible to enforce when using a geometric method such as a sweep or spiral pattern. This algorithm has the advantage that it is “anytime,” meaning that, at any iteration it provides paths that satisfy the survey requirements; however, these paths can potentially be suboptimal. This is essential when the application requires quick actionable results, as may be the case in extreme outdoor environments. The POPCORN algorithm together with a variety of supporting utilities is available open source as a python software package named WADL.

Our problem is an example of robotic coverage planning, similar to the algorithms that drive robotic vacuum cleaners to cover the floor of a room. Finding routes that physically cover an area has traditionally been solved by utilizing an underlying geometric pattern, such as a sweep, spiral, or space-filling curve to achieve coverage (3–6). These techniques, while fast to compute, do not necessarily apply to cases where multiple robots are tasked with simultaneously covering an arbitrarily shaped area. Geometric patterns can over-constrain how the coverage area is partitioned, which can lead to inefficient overall usage of multiple drones (7). Furthermore, these geometric methods do not consider collision avoidance constraints, or the extra travel needed when the take-off and landing zones are
physically far from the survey area. As a result, there can be a large gap between the start and end points of the survey section causing the UAV to re-traverse the same area, wasting battery life. In our case study, battery life conservation was paramount considering lithium polymer batteries have reduced capacity in cold weather. This required us to consider the entire trajectory of the UAV from launch to landing to make the most effective use of our limited battery life. By solving for cyclical paths, our method can cover the same area with shorter paths. For example, regarding the penguin colony survey task described in this paper, our method provides a 17.3% average path length reduction when compared to sweep-style paths over the same sized area.

In contrast to geometric methods, we encode our problem in predicate logic which allows statements such as “if the robot is at a point in the lattice at time \( t \), it must be at a neighboring lattice point at \( t + 1 \)” and “every point in the latices must be occupied at least once by at least one robot.” These conditions can be converted to formal logical statements and passed to an SMT solver to find a feasible solution for the resulting decision problem. This framework allows us to find paths for multiple UAVs, as well as encode other constraints such as a maximum path length that is possible within the limits imposed by the limited battery life. We can find shorter paths by iteratively solving the SMT problem and reducing the maximum allowable path length at each step, thus finding the shortest paths once the problem is infeasible. The freedom and flexibly afforded by SMT-based solutions allow our method to automatically produce paths for multiple UAVs to follow simultaneously, without the limitations of a particular geometric pattern.

Shorter aerial survey times allow for more surveys to be completed, capturing more data throughout the season. The Adélie penguin breeding season on Ross Island is approximately from late-October to mid-February (8). The images collected during each multi-robot survey were stitched together to form a large geo-referenced image. These
images will be used to estimate the number of nesting adults and chicks produced, as well as the nesting density in different parts of two colonies, Cape Crozier (77.4740°S, 169.1876°E) and Cape Royds (77.5554, 166.1580'E), on Ross Island, Antarctica. Ultimately, the data gathered will be used to test several hypotheses about the influence of fine-scale nesting habitats, nest density, and breeding success.

Figure 1: Drones at the Field Camp UAV safely landed at the Cape Crozier field camp on Ross Island, Antarctica. The dark colored ridge the UAVs had to fly over before beginning their surveys can be seen ascending to the right behind the field camp (left). UAV fleet taking off just outside of the field camp. The UAVs are circled in red (right).

Related Work

Our method extends the state of UAV field applications (9) by providing an autonomous multi-robot solution to UAV survey planning suited for any size of survey area and giving the user freedom to choose the start/end points for each flight and set a maximum path length. The most common solutions to UAV survey planning problems are sweep-style paths, also known as lawnmower or Boustrophedon paths, which route a vehicle back-and-forth over a rectangular space (10–12). Other similar space-filling patterns include spiral patterns (13, 14), and Hilbert curves (16). There are some multi-robot works involving the aerial inspection of 3D objects (17), but they require 3D geometry models, and are not immediately suited for 2D photogrammetry applications. Other multi-robot methods that are designed to find cyclical paths still can suffer from large amounts of
backtracking (7) when using sweep-style patterns. Large backtracking also occurs in graph
search approaches like the wave front method (3, 15). Energy-aware methods, also
commonly based on geometric primitives (16, 18), only take into account the survey
regions which implicitly assume that the UAV can take off or land near or within the
survey area. Methods that focus on minimizing the number of turns have also been used as
a proxy for minimum energy methods (19, 20), which are better suited for fixed wing or
fast-moving aircrafts. In contrast, we focus on rotary aircrafts which do not consume that
much energy when turning at slower speeds. Even though we do not explicitly solve for it,
our paths in Figure 3 and Figure 8 do contain sub-paths of sweep-like motions which are
optimally short (7, 12) for some rectangular areas. With many of these methods, the start
and end points of these paths tend to be far apart spatially, which becomes a limitation and
results in unnecessary backtracking if safe launch and landing locations are physically far
from the survey area. This is a common occurrence in many survey tasks.

Ultimately, the raw aerial image data gathered from these surveys are stitched
together to form a large area-covering mosaic. This larger image is then used for analysis
e.g., used by ecologists to count fauna and flora (21) for population analysis or wildfire
risk in forests and grasslands (22). Image stitching (23) is normally done via a feature
detection algorithm (24), such as Scale-Invariant Feature Transform (SIFT) (25), and a
bundle adjustment method (26, 27) to align the spatial data. Elevation models can also be
produced by recovering the 3D geometric information from the series of photos. These
tools have been well studied and can be found in open source (28) and commercial (29)
products. While the focus of this work is not on the image stitching, our path planning
must also consider the overlap requirements for the mosaic process. The size of the area
imaged from one photograph taken from the air is a function of the camera’s field-of-
view, sensor aspect ratio, and UAV altitude. This image size, in conjunction with the
UAV speed and photo interval area, is used to calculate the image overlap. The details are given in \((15, 30)\). Furthermore, different survey tasks, from photogrammetry to generating elevation models, may require different overlap requirements altogether. The final overlap requirement can be represented as a lattice with some spacing in meters. Even though our software package is independent of sensor technology and expects a desired lattice spacing rather than an overlap percent, we developed a simple tool to calculate the resulting overlap from the required image overlap from sensor and flight parameters. This tool is available in our open source package along with our POPCORN implementation.

Once the lattice spacing and the start/end points for each UAV are identified, we generate a grid world as a graph \(G(V, E)\) where each vertex \(v \in V\) represents a point in space over the survey area \((31)\). A UAV can travel between two vertices if there is an edge \(e \in E\) between them. This graph can be further refined by a user to specify keep-out zones, which are critical in our penguin survey application. The goal of the path planning algorithm is to return a sequence of vertices for each robot such that the union of all the paths cover the entire area. While decomposition methods that automatically divide survey areas can improve solution times by solving the problem in smaller steps \((31, 32)\), we manually divided the area (see Figure 3) due to the fact that we also had to consider the ground observer and terrain constraints. We assume that we have access to the geofence as well as elevation data of the survey area. For situations in which this data is not available, there exists a rich body of literature for informative path planning \((33–36)\) whereby the robots simultaneously explore their environment and plan paths to increase some information metric. These methods have been applied in plant health monitoring \((37)\) and underwater data gathering \((38)\).

Solving for a cyclic path through the coverage graph \(G(V, E)\) in the single robot case can be converted to the classic Traveling Salesman Problem (TSP) \((39–41)\). Since not
all graphs admit a TSP solution, we allow for node re-visitation in our formulation. Although TSP methods can be extended to allow for node re-visitation, the resulting problem is much bigger than the original. A recent single robot solution (42) combats this by combining a sweep planner to plan routes over subregions which are later stitched together via a TSP solver. Since the number of subregions is smaller than the original problem, the solution can still be found in a reasonable amount of time. A number of these formulations can be extended to multiple vehicles, known as the Vehicle Routing Problem (VRP) (43, 44), which are designed for routing vehicles through cities. Methods available for solving the TSP and VRP often resort to heuristic methods for large problems due to their exponential growth with problem size and focus on finding optimal solutions that are constrained to visit every vertex only once. While our planning problem is inherently NP-complete, we make the distinction that our method focuses more on finding feasible paths that satisfy a maximum path length constraint and allows for vertex re-visitation in contrast to traditional TSP/VRP. Since the maximum path length is dictated by the UAV’s battery constraints, we can find solutions tailored to the limitations of the chosen UAV rather than finding a vehicle that would need to meet some endurance requirement.

Conceptually similar to TSP and wave front methods, there are also approaches based on using the minimal spanning tree of a graph for coverage (45, 46); however they also suffer from large amounts of backtracking as some branches in the spanning tree correspond to long sections that are traversed twice.

We formulate the multi-robot coverage problem as an SMT, which is a generalization of Boolean Satisfiability (SAT), because it allows us to encode all the specific survey-related constraints. Others (47) have provided an in-depth description of SMT vs SAT. Most notably, SMT allows us to use predicate logic (universal and existential qualifiers) in addition to prepositional logic used in SAT solvers. While the
coverage problem can also be encoded as a MILP (1, 2), we found that for areas of similar size, MILP methods, even using state-of-the-art solvers, took 4–5 days to solve on a powerful workstation. In contrast, our method produced useable solutions for the same survey area in a few hours on a laptop. Although SMT is commonly used for mathematical theorem proving, it has recently been seen in a variety of robotic applications.

Specifically, SAT and SMT have been of interest in the motion planning community (48, 49) as they can model high level (e.g., task) and low level (e.g., dynamics) constraints within an expressive framework. In contrast to continuous time path planning frameworks, we formulate our problem as a discrete path planning problem on a graph (1, 50) which allows us to easily embed the coverage constraint into the path planning problem. Similar to the method presented in (51, 52), this work uses one-hot style encoding to translate the multi-robot path planning problem into an SMT. The works (51, 52), formulate an SMT problem to find paths for the robots to move to fixed designated goal positions under collision avoidance constraints. In contrast, we solve for paths that move the robots to completely cover the graph under path length constraints. Both our method and those in (51, 52) use an off-the-shelf SMT solver.

These SMT formulations have been successful in finding feasible paths for navigating obstacles (53), expressing reach-avoid problems (54). There have also been some implementations with manipulator arms (55) and multi-robot indoor navigation (56). Integrated-task-and-motion-planning is explored in (57) where the authors use SMT to plan low level paths through a complex kitchen environment to complete high level tasks. Finally, SAT/SMT-based solvers for a series of multi-robot patrolling and periodic routing problems are explored in (58,59) where the authors use SMT in conjunction with a TSP solver to generate paths for highly coupled multi-robot motion planning problems. The
flexibly afforded by SMT allows the particular details of the multi-robot survey problem to be encoded within a single framework. Although SMT is an NP-Complete problem, modern SMT solvers that unitize multiple processors minimst (60) and z3 (61) have accelerated the solution times for large problems (62, 63). Inspired from the SAT/SMT path planning methods, we present a multi-agent coverage path planning algorithm designed to compensate for the short battery life of small UAVs by minimizing backtracking when surveying large areas.

We develop a system for multiple UAVs to provide fast, repeatable aerial surveys of large-scale environments in extreme conditions. In this work, we demonstrate the utility of our system to advance population ecology in aerial surveys of Adélie penguin colonies in Antarctica. Current methods for aerial survey planning using commercial tools, such as Drone Deploy (64) and DJI mission planner (65), allow the user to define a geofence which produces a single, sweep-style path to survey the specified area. POPCORN accepts a geofence along with the desired grid spacing and maximum flight duration or length of the UAV flight. POPCORN will then automatically create a lattice based on the geofence and spacing to produce multiple paths under the maximum flight length. These paths will have notably less backtracking than the corresponding sweep-style path. The interface for POPCORN also allows the user to specify the speed and altitude of the UAV such that resulting paths can be exported to any mission control software such as QGroundControl (66) or UGCS (67).

Traditionally, colony survey data is gathered by expensive and disruptive helicopter surveys (68–70). Detailed noise levels of helicopter and UAV surveys are given in (71) showing that there is a considerable decrease in wildlife disturbance when using small UAVs over full-scale helicopters. Fixed-wing UAVs (airplanes) have also been used
in wildlife applications (72) but takeoff and landing considerations, as well as size requirements, make fixed-wing aircrafts infeasible for our remote survey operations. Fixed wing UAVs may also cause disturbance to wildlife as they can be perceived as predatory birds (73). In contrast, rotary-wing UAV surveys allow for more control over the survey process, as well as higher resolution images, as UAVs can fly at much lower altitudes without disturbing penguins. Additionally, when compared to helicopter surveys, utilizing multiple UAVs in an autonomous system allows for a higher frequency of faster surveys providing the ability to capitalize on brief windows of favorable weather. There has also been some work on conducting population surveys with ground robots roving through the colony (74), but this approach is not suitable for the scale of the colonies we surveyed, nor practical in the uneven, rocky, and steeply-inclined terrain.

Antarctic species have evolved to live in a cold climate, but as the environment changes, understanding how their populations react becomes critical for long-term ecology and conservation. Adélie penguins are important indicators of ecosystem function and change (75) in the Southern Ocean. In the face of rapid changes in sea ice and other factors in their pelagic environment, understanding their response to such changes is critical in assessing their ability to adapt to the changing climate and other potential perturbations.

Although nest density of Adélie penguins is known to vary with terrain (76, 77), the effect of nest density itself on nest success is challenging to examine, in part due to difficulties estimating breeding success for large areas and having enough variability of terrain represented to tease apart the independent effects of density and nesting habitat. Understanding the mechanisms of density dependence and the relative influence of nest density vs. nesting habitat on breeding success, will be critical for assessing this species’ ability to persist as warming increasingly leads to habitat change.
Once UAV data collection and the image analysis have been performed, we will estimate breeding success (chicks produced per breeding pair) as well as nest density for two entire Adélie penguin colonies (Cape Royds and Cape Crozier). Combining this data with the high-resolution digital elevation models generated from the UAV imagery will allow us to test the influence of fine-scale variability in nesting habitat and nest density on breeding success at all locations in the colonies. This analysis will be presented in a future publication focused on the ecology findings of the research, while this paper is focused on the multi-UAV robotic system developed to conduct the aerial surveys. The Cape Crozier colony is large enough to display substantial variability in nest habitat, density, and success. Thus, having data on the entire colony, as well as data for Cape Royds, will allow us to test hypotheses about spatial variability in habitats that otherwise might require data from numerous colonies.

As the climate warms, new areas in and near penguin colonies are being exposed by retreating glaciers and snowfields \( (78) \). Whether or not this new habitat is colonized may depend on the quality of the new habitat relative to the existing habitat. Habitat quality may be quantified, in part, in relation to its ability to mitigate the effects of climate change, as extreme weather may affect some parts of the colony more than others, turning some nesting locations into localized population sinks \( (79–81) \). Additionally, fine-scale elevation change, perhaps as little as a few inches, can affect the risk of a nest flooding and subsequently failing. Image data gathered from the surveys will be used to better understand the effects of climate change on penguin habitat quality.

**Results**

**Mission Outline**

We fielded a team of 4 UAVs executing paths planned with POPCORN to conduct a total of 10 aerial surveys of two Adélie penguin colonies on Ross Island, Antarctica between
November 2019 and February 2020. A video overview of the project can be seen in Movie S1. The chief focus of our surveys was the Cape Crozier colony (Figure 2), the largest colony with over 300,000 nests spreading over an area of 2 km². The Cape Crozier penguin rookery is situated from roughly sea level to 100 m above median sea level (AMSL) on the coast of Ross Island. The Crozier rookery is divided into two sections, the larger West Crozier and the smaller East Crozier, as seen in Figure 2. Between the rookery and a small 5-person field camp there are rocky ridges (100–400 m AMSL), permanent snow, ice fields, and nesting south polar skuas (*Stercorarius maccormick*) (Figure 4). These geographic constraints, as well as penguins moving about the colony, frequent high winds, and rocky snowy terrain, made it difficult to establish safe takeoff and landing zones within or around the edge of the colony. Furthermore, keeping batteries warm enough for flight (15°C) in the face of subzero conditions (temperatures over the austral summer season are -15°C to 0°C) for hours at a time was infeasible in the field. These restrictions required the UAVs to take off at the field camp, enter the colony (2 km from the camp) at key safe points, lower altitude (50 m) above the colony, decelerate to a non-disruptive speed (4 m/s) to take survey photos, and finally exit the colony near or at the
same key safe point to accelerate and fly back to the field camp as seen in Figure 1.

![Figure 2: Survey Locations. Cape Royds and Cape Crozier with location (star) on the Antarctic continent (top right). An example of the full mosaics produced during our survey of Cape Royds (bottom left) and Cape Crozier (bottom right) are shown, along with an individual image from the Crozier survey with hundreds of nesting penguins expanded to show individual nests.](image)

A separate colony on the other side of Ross Island, Cape Royds, with \( \sim 3,000 \) nests (see Figure 2) was also surveyed. Although these areas are much smaller than the main Crozier colony, similar geographic and logistical constraints were present. We conducted 8 surveys of Cape Crozier, and 2 of Cape Royds.

**Survey Path Planning**

The two colonies of interest (Crozier and Royds) with their locations on Ross Island are shown in Figure 2. An individual image with 72.2m × 55.4m ground area from the main Crozier area with hundreds of nesting penguins and an expanded view of individual nesting penguins can also been seen. Over 2,000 images were taken during each survey.
and stitched together to produce large mosaics which cover the whole colony, like those depicted in Figure 2. Ecologists will use these mosaics to identify adult and juvenile penguins during the egg incubation and chick brooding periods in an effort to generate population estimates of the colonies. In this article, we focus on the engineering advancements rather than the ecology.

Figure 3 shows the West Cape Crozier area with a regular grid with 40 m spacing between vertices. The figure also shows planned paths over the area along with the photo coverage and final stitched image. The figure illustrates the workflow of our system which takes in the border of a designated geographic area, and outputs paths for a given number of UAVs to survey the area. First, the area of interest is specified and overlaid with the coverage grid, Figure 3A. Next, we select the start and end locations and apply POPCORN to generate the paths offline seen in Figure 3B. Then, we launch the UAVs which execute these routes to collect the survey images. Simulated photo coverage of each UAV seen as the colored patches is shown in Figure 3C. Finally, we stitch the images together to produce the resulting mosaic image Figure 3D.

Due to the elevation variation across the colony, we divided West Crozier into 6 zones (z0-z5), seen as colored polygons in Figure 3B. This was done to ensure an observer was able to maintain visual line-of-sight to all UAVs from a single point for the entire route within the zone. The observer path is shown in Figure 3B as a magenta line. An observer needed to be present for compliance with the United States Antarctic Program (USAP) flight mandate for all UAV activity, as well as to monitor for any wildlife disturbance. The interior zones were also placed in such a way that approach paths were not above any nesting penguins. As a result, we could use the majority of the battery life for the survey task without having to risk losing a UAV by flying over sea ice or open water at the north side of the colony boundary. The Cape Crozier colony required 13
flights (11 for West Crozier and 2 for East) per survey. By implementing our multi-UAV survey method, we were able to achieve a 3-hour (2 hours of flight time) survey time for Cape Crozier, in contrast to the two-day single manual piloted survey done in the previous year (82). Cape Royds, on the west side of Ross Island, is home to about 3,000 breeding pairs and only required a single seven-minute flight per survey.

Figure 3: Planning Workflow. The coverage grid (A) of West Crozier with planned UAV paths in blue and green with a human observer path in magenta and six zone locations (Z0-Z5) (B). The resulting photo coverage with individual path coverage in blue and green hues corresponding with Fig3B (C) with the final stitched mosaic (D).

The paths in Figure 3 were imported in UGCS (67), a flight ground control software which handles the interface between our route planner and the UAV’s onboard flight computer. While the paths were generated offline, their cyclical property allowed the user to place the starting node anywhere along the path without recalculating the whole path to account for unforeseen issues that may arise in the field. For example, we
encountered a group of skuas along one of the entry paths from the launching area and were able to adjust the start point accordingly in the field without having to re-plan the entire route.

Figure 4 shows the total path for each route, including the field camp staging area in the lower left corner. The human safety pilots were placed at the yellow dot to maintain line-of-sight to the UAV so that manual control could be established in case of an emergency. The safety pilots were purely ancillary in our case, watching as the UAVs executed their path autonomously. Orange no-fly zones mark the area of skua territory and were avoided in order to minimize disturbance to nesting birds. After the pilots initiated launch, the UAV flight controller executed the routes and autonomously landed back at the field camp to be prepped for the next flight.

**Figure 4: Survey Routes over Cape Crozier.** All routes for the Crozier area are shown here. The paths in blue-green were planned by our method and executed with a team of UAVs (unpiloted aerial vehicle) to conduct aerial surveys of the Cape Crozier and East Rookery penguin colonies. No-fly zones in orange mark the area of mountains or nesting skuas. Takeoff and landing zones at the field camp are marked by the green arrow in the lower left corner. To comply with USAP (United States Antarctic Program) safety guidelines, the safety pilots were stationed at the yellow dot to maintain line-of-sight to the UAVs and the ground observer followed the magenta path.

**Flight Time Efficiency**
A major focus of this research was to reduce the total flight time of the survey; thus, we consider path length as a constraint rather than a consequence of the planner. We found experimentally that in Antarctic weather conditions (-10° to 0°C and 5–10 knots), the battery drained at .08 ± .01 %/s. Factoring the large transit distances, (1–2 km), the UAVs needed to traverse between the colony and field camp at the speed limit within the colony (4 m/s) resulting in about 10 minutes of usable survey time per UAV. Our planner was designed to minimize the amount of backtracking while still meeting the coverage requirements. To capture the amount of redundant travel due to our planner, we define the path efficiency of a set of \( M \) plans as,

\[
\eta_{\text{path}} = \frac{\sum_{i=1}^{M} V_{\text{unique}}^{(i)}}{\sum_{i=1}^{M} V_{\text{total}}^{(i)}}.
\]

Where \( V_{\text{unique}}^{(i)} = V_{\text{total}}^{(i)} - V_{\text{repeat}}^{(i)} \) is the number of unique vertices traversed by UAV \( i \) and \( V_{\text{total}}^{(i)} \) is the total number of vertices in the path for UAV \( i \). In the coverage problem, the number of unique vertices is also the total the number of vertices in the graph, \( \sum_{i=1}^{M} V_{\text{unique}}^{(i)} = N_{\text{vertex}} \).

Likewise, the total number of vertices is just the number of steps taken by all the UAVs in aggregate, \( \sum_{i=1}^{M} V_{\text{total}}^{(i)} = N_{\text{path}} \). An efficiency of 1 implies that no UAV needs to backtrack (\( V_{\text{repeat}}^{(i)} = 0 \) for all \( i \)) over any vertex in order to complete the path. We compared POPCORN to the sweep-style method common in the literature (7, 10–12, 21), as well as a recent cellular decomposition polygon coverage path planning method (42). The sweep routes were planned using the area coverage feature already implemented in the UGCS software (67). The polygon coverage method (42) first divides the desired area into polygons and then uses a sweep method to plan a coverage path over each polygon. These
routes are then stitched together via a TSP solver. The intermediate polygonal division step allows the planner to locally optimize the sweep direction for each polygonal cell, which leads to shorter paths in most cases. Despite the increased efficiency of polygon coverage method over the naïve sweep method, both methods still suffer considerable backtracking, as the core planning strategy in both methods is to use a sweep pattern. A performance comparison between our POPCORN algorithm, the standard sweep method, and the polygon decomposition (42) method are given in Figure 5. Efficiency for all methods is calculated as the ratio between the non-repeated section and total path length. Table 1 shows the path efficiency for our method, as well as the sweep and polygon methods; the more efficient set of routes is in bold. POPCORN obtains the best efficiency in all cases but one, with an average efficiency improvement of 13.7% over the sweep planner and 9.0% over the polygon coverage method.

Using the battery drain data and the transit distance between the colony and field camp, we can find the flight length limit for a zone, which can be thought of as a “distance budget” due to the predicted remaining battery. Zones that were far from the field camp (z2, East Crozier) or needed fewer UAVs (z5, Royds) had a smaller budget than the other zones. If a set of routes exceeds this limit, then at least one UAV will not have enough battery to complete the route and return home safely. This limit and the total transit distance, as well as the aggregate route lengths, are also given (in km) in Table 1 with the length underlined if it exceeds the distance allowance. POPCORN never exceeded this distance allowance, while the other planners exceeded the allowance in at least 2 of 8 cases. Each zone specifies a section of the survey where a ground observer can maintain visual-line-of-sight to the UAV over the entire course of the route. The larger zones had a size of just over 100 vertices, which corresponds to about 8 New York City blocks. The
paths planned with POPCORN shown in Figure 3 backtracked on 4 vertices (about 120 m) in the worst case.

Figure 5 shows a side-by-side comparison of the standard sweep-style, and polygon coverage method against our method. Not accounting for transit portion of the path, the sweep method results in a total path over 1 km longer than the path produced by our method. The polygon method produced a path 400 m longer than the one produced by our method. This 400 m reduction saves 100 seconds of flight time, or 10.4% of the total battery available. Across all instances, POPCORN has an average reduction of 469 m over the polygon method, shortening flights by an average of 2 minutes or 12.5% of the 16 minutes battery life.

| Table 1: Route Efficiency |

<table>
<thead>
<tr>
<th>Area</th>
<th>Croz z0</th>
<th>Croz z1</th>
<th>Croz z2</th>
<th>Croz z3</th>
<th>Croz z4</th>
<th>Croz z5</th>
<th>E. Croz</th>
<th>Royds</th>
</tr>
</thead>
<tbody>
<tr>
<td>Transit Limit</td>
<td>5.8 km</td>
<td>4.9 km</td>
<td>5.6 km</td>
<td>4.3 km</td>
<td>7.0 km</td>
<td>1.9 km</td>
<td>9.6 km</td>
<td>0.2 km</td>
</tr>
<tr>
<td>POPCORN Length</td>
<td>4.6 km</td>
<td>4.7 km</td>
<td>3.5 km</td>
<td>3.6 km</td>
<td>4.4 km</td>
<td>2.5 km</td>
<td>3.9 km</td>
<td>1.1 km</td>
</tr>
<tr>
<td>N vertex 104</td>
<td>106</td>
<td>80</td>
<td>82</td>
<td>103</td>
<td>56</td>
<td>89</td>
<td>51</td>
<td></td>
</tr>
<tr>
<td>N path     108</td>
<td>110</td>
<td>84</td>
<td>84</td>
<td>104</td>
<td>58</td>
<td>92</td>
<td>55</td>
<td></td>
</tr>
<tr>
<td>η path    0.967</td>
<td>0.964</td>
<td>0.952</td>
<td>0.976</td>
<td>0.99</td>
<td>0.955</td>
<td>0.967</td>
<td>0.927</td>
<td></td>
</tr>
<tr>
<td>Sweep Length</td>
<td>5.9 km</td>
<td>5.4 km</td>
<td>3.8 km</td>
<td>4.6 km</td>
<td>5.0 km</td>
<td>3.6 km</td>
<td>4.6 km</td>
<td>1.3 km</td>
</tr>
<tr>
<td>Backtrack 1.1 km</td>
<td>1.1 km</td>
<td>0.35 km</td>
<td>0.67 km</td>
<td>0.78 km</td>
<td>0.27 km</td>
<td>0.86 km</td>
<td>0.06 km</td>
<td></td>
</tr>
<tr>
<td>η path    0.821</td>
<td>0.801</td>
<td>0.908</td>
<td>0.926</td>
<td>0.843</td>
<td>0.926</td>
<td>0.813</td>
<td>0.953</td>
<td></td>
</tr>
<tr>
<td>Polygon (42) Length</td>
<td>5.7 km</td>
<td>4.7 km</td>
<td>4.3 km</td>
<td>4.2 km</td>
<td>4.5 km</td>
<td>2.9 km</td>
<td>5.7 km</td>
<td>1.1 km</td>
</tr>
<tr>
<td>Backtrack 0.83 km</td>
<td>0.47 km</td>
<td>0.51 km</td>
<td>0.43 km</td>
<td>0.63 km</td>
<td>0.30 km</td>
<td>0.46 km</td>
<td>0.12 km</td>
<td></td>
</tr>
<tr>
<td>η path    0.854</td>
<td>0.900</td>
<td>0.881</td>
<td>0.898</td>
<td>0.861</td>
<td>0.896</td>
<td>0.921</td>
<td>0.895</td>
<td></td>
</tr>
</tbody>
</table>
**Figure 5: Comparison of Route Planning Methods.** Comparison of our method, POPCORN (left), sweep-style path (middle), polygon coverage (42) (right) over Cape Crozier zone, z5. For this zone, our method produces a 30% and 14% shorter path than the sweep and polygon methods, respectively, when planned from the same start and end point.

Although the polygon coverage method (42) mostly produced shorter and more efficient routes than the sweep planner, in some cases it generated nearly degenerate (large aspect ratio) polygons during its decomposition step. The route segments planned in these nearly degenerate polygons resulted in considerable backtracking once stitched together for the final route. POPCORN results in an average reduction of 11.5% in path length when compared to the polygon method. Figure 6 shows the comparison between our method and the same polygon coverage method over all zones in the main Crozier colony. While the sweep method produces adequate coverage, it still suffers from a large portion of backtracking as shown in Figure 5. The polygonal coverage (42) method reduces this slightly by changing the sweep direction at various points, but still suffers from backtracking when stitching together adjacent regions, which can be seen in Figure 6.

Figure 3 shows that, like continuous space planners akin to the polygon coverage method (42), POPCORN still effectively contours to the boundary of the survey region, despite its discrete formulation.

**Figure 6: Comparison of Route Planning Methods.** Comparison of our method, POPCORN (left) and the polygon coverage (42) (right) over all west Crozier zones (z0–z5). While the polygon method contours better to the edges of the survey area, it can lead to degenerate polygons resulting in more backtracking once
the paths are stitched together. This can be seen in the top left of z0 and bottom right of z2. POPCORN results in an average reduction of 11.5% in path length when compared to the polygon method. Our method also automatically splits the paths over multiple UAVs whereas the polygon methods is for a single UAV.

**Robustness to Early Battery Depletion**

By optimizing for a looped path, we show that the drone can exit the path at any point and have enough reserve battery to return to the start of the survey and then proceed to the takeoff location in the event the UAV needs to be recalled. The route’s start points were set such that the UAVs would start the survey at no lower than 80% battery and exit with at least 30%. Assuming worst case battery usage (3σ), we can determine a no-return distance for any battery percentage within the survey window. This distance is calculated as the total time remaining at the current percentage times the maximum allowable flight speed, 4 m/s, over the survey area. Figure 7 shows the battery usage data taken from the flight logs of the 11 routes over West Crozier. All routes are designed to exit at or near (one grid point away) the start point. Every route is under the no-return line, indicating that they can safely return to the start of the survey and then proceed home in a way that would not disturb any penguins. Over the course of the entire season, UAVs were safely recalled in the case of camera errors, telemetry loss, or in the case of incoming inclement weather.
Figure 7: Point of No-return Bounds No-return bounds with each of the 11 flights over West Crozier. The no-return (dotted) line marks the maximum distance as a function of battery level the UAV can be from the start of the survey and still return home safely.

Conclusion

Our method provides a general and flexible path planning and solution method for a team of aerial UAVs to complete a coverage task. This planning algorithm allows scientists to survey larger areas quickly and robustly using multiple UAVs. Additionally, utilizing multiple UAVs in an autonomous system allows for a higher frequency of faster surveys with less overhead and higher resolution when compared to helicopter surveys, thus better capitalizing on small weather windows. The UAVs also do not need to backtrack over the coverage region to return to base because we can enforce cyclical paths. Multi-rotor UAVs are constrained by their short flight time, so avoiding such backtracking effectively increases the amount of coverage area per flight, as seen in Figure 5. Furthermore, our algorithm naturally leads to a “safe emergency recall” property for the UAVs. The cyclical path, coupled with a maximum path length constraint, causes the UAVs to fly to farther points in the grid first and work their way backward toward closer points. This allows for safe recall in the case of faster than expected battery drainage or some other hardware
fault, given that the UAV will always have an “exit” trajectory available that is as close as possible to the base. This safety property of our planner is highlighted in Figure 6.

We demonstrated the effectiveness of our algorithm by implementing it on a team of UAVs and conducting field experiments resulting in reduction from 2 days to 3 hours to survey the colony compared to a human-piloted single-drone survey. The added speed in the survey process provides robustness against the rapid changes in weather and allows for more data to be gathered during the breeding season. Other applications requiring quick aerial surveys in extreme conditions, such as surveys of other wildlife populations, disaster sites, or forests and brush lands to detect wildfires in high-risk weather conditions, could also benefit from our algorithm. Our efforts will culminate in a survey planning tool for any number of UAVs that has the capability for on-line re-planning to adapt in real-time to changing survey conditions.

**Materials and Methods**

**SMT** We encode the path planning problem as an SMT using a one-hot encoding scheme similar to (51). Let $X_{i,v,t} \in \{0,1\}$ be a Boolean variable where $i \in \{1,\ldots,N_{\text{robots}}\}$, $v \in V = \{1,\ldots,N_{\text{vertex}}\}$, and $t \in \{0,\ldots,T_{\text{max}}\}$. If $X_{i,v,t} = 1$ it implies that robot $i$ is at vertex $v \in V$ at step $t$. The parameter $T_{\text{max}}$ is the longest path allowed by any robot, e.g., if $T_{\text{max}} = 20$ then robot will traverse at most 20 vertices. Although the value for $T_{\text{max}}$ can be different for each robot, e.g., a heterogenous robot system, for simplicity we assume that $T_{\text{max}}$ is the same for each robot and that there are enough robots to cover the graph. The graph $G(V,E)$ is constructed based on the survey area and the requisite density derived from the image overlap requirements. Various logical statements are set to describe the desired behavior allowed by the robots. The solver attempts to assign each Boolean variable, $X_{i,v,t}$, either 0 (False) or 1 (True) such that these statements, formulated in
predicate logic, evaluate to True. These logical statements can be thought of as constraints, similar to how an optimization problem is formulated.

To enforce the physical constraints of the coverage problem, we first require

\[ \forall (i, t) \exists \nu \, X_{i,\nu, t} = 1, \]

is only true for only one \( \nu \). This statement constrains the robot \( i \) to only be at one point \( \nu \) at any given step \( t \). Next, we enforce the dynamics, allowing robot \( i \) to move to vertex \( y \) at step \( t+1 \) only if it was at vertex \( \nu \) at step \( t \), such that there is an edge, \((\nu \sim y)\), between the two vertices,

\[ \neg X_{i,\nu, t} \lor \nu \sim y \lor X_{i, y, t+1} = 1 \]

In order to promote progress through the graph, we only connect the start/end vertex to itself and all other vertices have no self-loops. Next, we require that

\[ \forall \nu \exists (i, t) \, X_{i,\nu, t} = 1 \]

which enforces the coverage task constraint requiring that for every vertex \( \nu \) in the graph, there is at least one step \( t \) where there is some robot \( i \) that occupies that space. If we were to enforce that there exists only one pair \((i, t)\) where this constraint is met, then this would be equivalent to the TSP. Inter-robot collision avoidance constraints can also be added by requiring that there is at most one \( i \) such that

\[ \forall \nu, t \, X_{i,\nu, t} = 1. \]

Lastly, we encode the start and end points. Let \( \nu_i^0 \) be the start and end vertex for robot \( i \),

\[ X_{i,\nu_i^0, 0} = X_{i,\nu_i^0, \tau_{\text{max}}} = 1. \]

We desire a closed loop as a path to ensure that the robot does not have to re-traverse the same area to return to the launch site. If a closed loop path is not desired, then separate boundary conditions can be set.
Since the maximum path length is informed by the battery usage of the robot, finding a feasible path implies that the robot will have enough battery to reach the end of the path. In the case of an emergency recall, the robot will have enough battery to safely return home because the Euclidean distance between any point on the path to the end is at most the length of the rest of the path. When the robot cannot safely complete the survey (e.g., inclement weather area or loss of visual), this feature allows the UAV to safely return to the start of the survey path and proceed back to the launch point while obeying any sensitive wildlife or geographic constraints. This becomes more important if the battery consumption rate was underestimated. This feature encourages the UAV to move closer to the start of the path as it reaches the end of the survey, as can be seen in Figure 7.

**Variable Reduction** Although these four constraints completely encode the multi-robot coverage problem at an SMT, the number of Boolean variables can be reduced by exploiting the problem structure. Given that each robot can only move one adjacent vertex every timestep, binary variables that represent unreachable states can be removed from the problem to reduce the solution time. Algorithm 1 describes the Forward-Backward Reduction method.

```
Algorithm 1: Forward-Backward Reduction

for each robot i do:
    for each vertex v do:
        d ← Distance(v, vi0)
        for t ∈ {0, ..., d − 1} do:
            Remove(Xi,v,t)
            Remove(Xi,v,Tmax−t)
```

The *Distance()* function returns number of time steps needed to travel between two vertices. In a rectilinear grid world, this *Distance()* function can be thought of as the $L_1$, or Manhattan distance. The *Remove()* function simply removes the Boolean variable by constraining it to be False before the solver assigns the values.
**Sequential SMT** The SMT instances do not optimize an objective function directly and instead only find an anytime feasible solution. To transform a feasibility framework into an optimization one, we use the standard method of descending (or ascending) on an objective function until the problem is infeasible. To find the shortest path, Algorithm 2 preforms this iterative process by repeatedly solving SMT instances where $T_{\text{max}}$ is successively reduced each step according to some search schedule until the user terminates or the problem is infeasible. From the construction of the problem, each successive iteration guarantees the resulting paths are not longer than the previous iterate, assuming a solution is found. As an example, we provide a linear search schedule in for $T_{\text{max}}$ in Algorithm 2, but in practice, any reasonable search schedule is applicable. For the `Solve()` method we use the freely available z3 solver (61).

```
Algorithm 2: Sequential SMT

SMT ← Init(problemData)
sol ← null
while True do:
    if Solve(SMT, $T_{\text{max}}$) is feasible:
        sol ← getSol(SMT)
        $T_{\text{max}} ← T_{\text{max}} - 1$
    else:
        Return sol
```

**Example Instance** Consider 5 UAVs tasked with surveying an arbitrarily shaped area. The coverage requirement is such that the coverage graph has 74 vertices, and the robots enter through a node at the periphery of the survey area. As $T_{\text{max}}$ is reduced, the paths are constrained to be shorter; thus, the paths become more spatially efficient. Although each problem instance has initially around 6,500 Boolean variables, after applying Algorithm 1, about 3,500 variables are removed before the problem is passed to the solver. Table 2 shows the effect Algorithm 1 has on solution times and Figure 8 provides the paths for this example instance. The paths start at the circle symbol and end at the star symbol, which is one step away from the start vertex.
Figure 8 Example Path Planning Instance: A set of paths for a multi-robot coverage problem with 5 robots. The initial/final locations are shown on the lattice grid in (A) and figures (B–D) show the set of routes produced with the corresponding maximum path length, $T_{\text{max}}$.

Table 2: Example cases number of variables and solution times given before and (after) reduction

<table>
<thead>
<tr>
<th>$T_{\text{max}}$</th>
<th>Variables (w/ reduction)</th>
<th>Solution Time (w/ reduction)</th>
</tr>
</thead>
<tbody>
<tr>
<td>20</td>
<td>7400 (3896)</td>
<td>5.82 sec (2.42 sec)</td>
</tr>
<tr>
<td>19</td>
<td>7030 (3526)</td>
<td>11.39 sec (1.32 sec)</td>
</tr>
<tr>
<td>18</td>
<td>6660 (3156)</td>
<td>18.24 sec (9.48 sec)</td>
</tr>
<tr>
<td>17</td>
<td>6290 (2786)</td>
<td>12.6 sec (5.22 sec)</td>
</tr>
</tbody>
</table>

SMT solution times do not always decrease with the number of variables. However, Algorithm 1 removes variables that represent impossible states for the robots; thus, the solver will explore less branches when searching for a feasible assignment.
References and Notes


47. D. Yurichev, Sat/smt by example (2019).
48. E. Scala, M. Ramirez, P. Haslum, S. Thiebaux, Numeric planning with disjunctive global constraints via SMT, in 26th International Conference on Automated Planning and Scheduling (2016).


52. P. Surynek, "Reduced time-expansion graphs and goal decomposition for solving cooperative path finding sub-optimally." In Twenty-Fourth International Joint Conference on Artificial Intelligence (2015)


64. Drone Deploy, Drone software for industry innovators. [www.dronedeploy.com](http://www.dronedeploy.com).

65. DJI, DJI Ground Station Pro. [www.dji.com/ground-station-pro](http://www.dji.com/ground-station-pro).

66. Dronecode, QGroundControl. [qgroundcontrol.com](http://qgroundcontrol.com).


82. UNAVCO, A less invasive, more accurate way to survey penguin colonies. www.unavco.org/highlights/2019/penguins.

Acknowledgments

**Funding:** This work was supported by NSF Grant 1834986 with logistical support provided by the United States Antarctic Program (USAP).

**Data and Code Availability:** An open source version of the codebase for this project can be found at github.com/k2shah/wadl/ and the version used in this work can be found at github.com/k2shah/wadl/releases/tag/0.0.1

**Author contributions:** The algorithm was designed by K.S. and M.S. Field operations were conducted by K.S., G.B., and A.S. The paper was written by K.S., G.B., A.S., M.S.

**Competing interests:** The authors declare that they have no competing interests

The authors thank David Ainley, Katie Dugger, Megan Elrod, Dennis Jongsomjit, Parker Levinson, Amélie Lescroël, and Virginia Morandini for their assistance with the field operations.