Holistic Energy Awareness for Intelligent Drones

Srinivasan Iyengar
Microsoft Research India
t-sriyen@microsoft.com

Ravi Raj Saxena
Indian Institute of Science
ravisaxena@iisc.ac.in

Joydeep Pal
Indian Institute of Science
joydeeppal@iisc.ac.in

Bhawana Chhaglani*
University of Massachusetts Amherst
bchhaglani@cs.umass.edu

Anurag Ghosh*
Carnegie Mellon University
anuraggh@andrew.cmu.edu

Venkata N. Padmanabhan
Microsoft Research India
padmanab@microsoft.com

Prabhakar T. Venkata
Indian Institute of Science
tvprabs@iisc.ac.in

ABSTRACT

Drones represent a significant technological shift at the convergence of on-demand cyber-physical systems and edge intelligence. However, realizing their full potential necessitates managing the limited energy resources carefully. Prior work looks at factors such as battery characteristics, intelligent edge sensing considerations, and planning in isolation. But a global view of energy awareness that considers these factors and looks at various tradeoffs is essential. To this end, we present results from our detailed empirical study of battery charge-discharge characteristics and the impact of altitude and lighting on edge inference accuracy. Our energy models, derived from these observations, predict energy usage while performing various maneuvers with an error of 5.6%, a 2.5X improvement over the state-of-the-art. Furthermore, we propose a holistic energy-aware multi-drone scheduling system that decreases the energy consumed by 21.14% and the mission times by 46.91% over state-of-the-art baselines. We release an open-source implementation of our system. Finally, we tie all of these pieces together using a people counting case study.

CCS CONCEPTS

• **Computer systems organization → Embedded and cyber-physical systems**.

KEYWORDS

energy-awareness, scheduling, low-cost sensing, drones, UAVs

ACM Reference Format:
Srinivasan Iyengar, Ravi Raj Saxena, Joydeep Pal, Bhawana Chhaglani, Anurag Ghosh, Venkata N. Padmanabhan, and Prabhakar T. Venkata. 2021. Holistic Energy Awareness for Intelligent Drones. In *ACM International Conference on Systems for Energy-Efficient Built Environments (BuildSys '21), November 17–18, 2021, Coimbra, Portugal.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3486611.3486651

*Work performed while the author was at Microsoft Research India.

BuildSys '21, November 17–18, 2021, Coimbra, Portugal

ACM ISBN 978-1-4503-9114-6/21/11…$15.00
https://doi.org/10.1145/3486611.3486651

1 INTRODUCTION

Unmanned Aerial Vehicles (UAVs), popularly known as drones, represent a significant technological shift, at the convergence of on-demand cyber-physical systems and edge intelligence. They have shown tremendous promise in collecting data that are either dangerous, expensive, or impractical to obtain otherwise [31]. Drones have provided disruptive solutions for monitoring natural and built environments in varied applications such as agriculture [38], renewables [23], wildlife conservation [33], search and rescue [34], etc. Drone-based applications are taking off because of the availability of open-source hardware and software platforms [27, 40] at an affordable cost [11], and regulatory clarity in many countries [4, 5].

As drones are mostly battery-powered, managing their energy resources is critical, whether we consider an individual drone in isolation or a fleet of drones deployed for a task. Increased battery life leads to longer flight times, enabling more extensive coverage with the available drones, and a higher uptime with fewer charging pitstops. To highlight the significance of energy constraints on drones, consider a disaster recovery scenario illustrated in Figure 1 that shows a flood-affected area along the banks of a river. The two drones deployed — shown as blue and red crosses — fly along pre-determined routes, with the mission of identifying possible human subjects in harm's way. The aerial picture shows the objects detected by the blue drone using an on-board camera. Ensuring that the area of interest is covered quickly and effectively by the drones available would require careful management of their battery energy resource, which, in turn, requires overcoming several challenges.

First, charging should ideally happen autonomously (e.g., with a wireless charging pad [22]), so that the drones can remain in a perpetual cycle of getting charged, taking off for their mission, and then landing to get charged again, without any human involvement. There is the question of the level to which to charge the drone batteries. To the extent that the rate of charging is a concave function of time, it might be beneficial to charge partially and free up the charging station for another drone to use. Moreover, in a setting with a fleet of drones — a common case for large-scale sensing tasks — there might be heterogeneity in the capacity, chemistry, and age

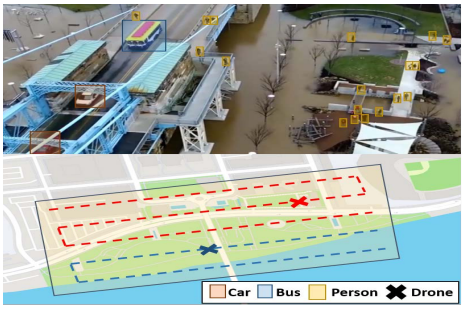


Figure 1: Multiple drones surveying a given area for disaster recovery

Figure 2: Varying weather, lighting and altitude

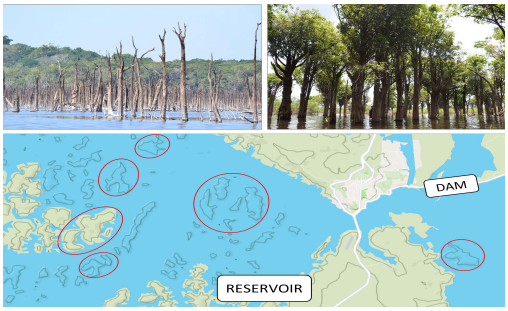


Figure 3: Distributed irregular survey regions for ecological assessment of dam construction

of batteries. So, we need to create custom charging and discharging profiles for each drone battery.

Second, while most applications to date have used the drone as a "dumb" sensing device [26, 28, 37], there are many situations where it is advantageous to have on-board intelligence on the drone for local sensor stream processing (E.g. bandwidth constraints and privacy concerns). Energy again plays a non-trivial role in edge computation i.e., the interplay between battery energy and the accuracy of computation with varying weather and lighting conditions (see Figure 2). For instance, in our disaster recovery scenario, drones flying at a greater height could capture a larger area using their cameras, thereby enabling greater coverage for a given amount of battery drain. However, the detection accuracy for the objects of interest (e.g., people, vehicles) would likely suffer due to the reduced size of the objects in the captured images (e.g., see Figure 1). Flying the drones at a lower height would likely improve the object detection accuracy but at the cost of reduced coverage and hence an increased survey time. Hence, there is a need to balance the energy cost and the task accuracy of each computing platform.

Finally, large-scale surveying tasks might need the services of multiple drones to complete in stipulated time. However, partitioning the task among a heterogeneous fleet of drones is non-trivial. Consider a scenario shown in Figure 3. Here, a surveying task entails assessing the ecological impact after creating a reservoir for a dam by counting the number and species of the submerged trees — similar to a problem discussed in [30]. Due to the undulating bed of the reservoir lake, the partially submerged regions are distributed across a wide area. Thus, it is important to consider the location, shape, and square footage of these regions to divide the surveying task optimally across multiple drones available at one's disposal.

Prior work has looked at the effect of battery charecteristics [10, 36], path planning and scheduling [6, 20, 39], and effect of variation in drone imagery [13] on drone operation in isolation. However, a holistic view is needed to break the silos by considering tradeoffs between these factors to develop energy-aware drone scheduling. We address these challenges through the following contributions:

(1) **Drone energy modeling:** We present an extensive empirical study based on over 100 drone flights to arrive at several observations regarding the drone charging and discharging characteristics. Based on these, we perform a detailed and accurate modelling of battery performance. We report an error of just 1.94% in predicting the battery charge times. Furthermore, our novel predictor accurately predicts energy consumption for different drone manoeuvres, with an average error of 5.6% — a 2.5× reduction in error from the state of the art [36]. We also discuss the tradeoff between drone energy consumption and varying altitude, lighting conditions, etc.

(2) **Energy-aware Scheduling:** Informed by our observations from the empirical study, we build a holistic system to create energy-aware flight paths for a given task. Our system generates energy-aware flight path for a given shape via a novel path planning algorithm and splits the task among a fleet of drones via an Integer Linear Programming (ILP) model. Thus, our system decreases the average energy consumption by 21.14% when the shape regularity is less than 0.85 (and overall improvement of 10.15%) and an average relative time reduction of 46.91% in a multiple drone setup, over the variety of baselines and the state of the art.

(3) **Open-source implementation:** We create an open-source toolkit for charging and discharging energy models of drone. We also implement our system as an extension to the most widely used open-source mission planning software, qGroundControl [40]. Our implementation involved writing over 2000 lines of C++ code and 400 lines of QML scripts.

2 ENERGY CHARACTERIZATION AND MODELING

In this section, we discuss some interesting and non-intuitive patterns learned through an extensive empirical study, which included completing over 100 flights on drones with varying payload along with charging the batteries in varied settings (i.e., different initial charging current). We use a custom-built quadcopter of 1-meter diagonal length with 15-inch propellers and weighing around 900 grams without the battery. For the experiments presented in this paper, we used an 8000 mAh LiPo battery. Moreover, we introduce accurate models for the time taken to charge the batteries and the energy consumption on various maneuvers.

2.1 Battery Charging Characterization

High discharge currents with high energy density, and flexibility in shape and size make Lithium-Ion (Li-ion) or Lithium-Polymer (LiPo) batteries ideal candidates for powering drones. Similar to any other

battery chemistry, Li-based ones also have some peculiarities. Here, we highlight the effect of two main factors — i) starting current, and ii) age — on the LiPO battery used in our drones.

Impact of starting current: The initial battery charging happens at a constant current level (CC mode). After a while, the charging process reaches an inflection point beyond which the charging happens at a constant voltage level (CV mode). Higher currents allow faster charging, while at the same time, the battery capacity degrades more quickly over multiple cycles [8]. Over the entire empirical study, we experimented with different charging currents. Figure 4(a) illustrates the impact of the initial current on the charging time of the battery. The figure presents the change in the charging current, the battery voltage, and the cumulative energy delivered for the three different charging sessions. For this experiment, we considered the batteries of the same age, i.e., ones that had undergone the same number of charge-discharge cycles. Specifically, we looked at the time taken to deliver the last 140 Wh to charge the batteries fully [1]. The overall time taken with the initial current of 6.4A, 3.7A, and 1.8A is 72 minutes, 112 minutes, and 207 minutes, respectively. For applications that require higher drone uptime, with disregard for battery capacity degradation down the line, one must choose higher initial charging currents.

Observation: *Higher starting current leads to a commensurate reduction in charging times.*

Impact of Aging: The energy density of Lithium-based batteries is affected by ageing [8]. Battery ageing is caused by cell oxidation — an irreversible process. Over multiple charge-discharge cycles, the usable capacity drops consistently. We observed the effects of ageing over 200 charging cycles spread across multiple batteries.

Figure 4(b) illustrates the impact of ageing on the time taken to deliver the final 140Wh of energy to charge two batteries fully. Both batteries are charged with the same initial charging current of 6.4A. **Battery 1** is a new battery with fewer than five charging cycles, whereas **Battery 2** had been put through over 130 charge-discharge cycles.As shown, the current and energy charging profiles are identical in the first 34 minutes, as both batteries are in the CC mode of operation. However, the voltage increase in **Battery 2** is much quicker as it reaches the inflection point faster. After reaching the inflection point, **Battery 2** continues to get charged in the CV mode at a much slower rate, as indicated by the flatter energy curve. **Battery 1** switches from CC to CV mode much later — around the 45-minute mark. To fully charge **Battery 1** and **Battery 2**, it takes 79 and 113 minutes respectively. Thus, the time taken to fully charge batteries with the same quantum of energy depends on their respective age.

Observation: *Older batteries reach the inflection point faster and take longer to charge a given amount of energy.*

2.2 Battery Charging Modeling

The charging process is non-linear (due to the CC and CV modes) and varies with charging current and battery age. Thus, there are opportunities for drone operators to create drone flight schedules that optimize drone uptime by knowing the charging profiles of various drone batteries in their fleet. Using the observations made

[1] As batteries degrade over time, their capacity reduces. In our setup, 140 Wh was the minimum energy needed to fully charge the oldest battery.

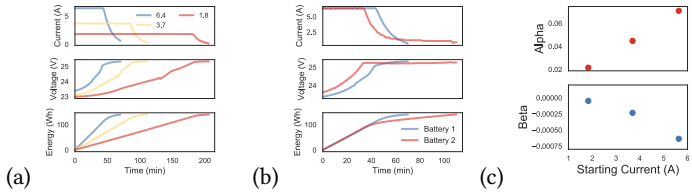


Figure 4: (a) Impact of starting current, (b) Impact of aging on charging time, (c) Charging parameters vs current

through our empirical study, we now present our model to characterize the charging profiles accurately. Figure 4(a) shows how the current profile is flat until the inflection point and experiences an exponential reduction in the CV mode. On the other hand, the voltage profile follows a quadratic curve until it reaches the inflection point and stabilizes at a maximum value in the CV mode.

Let ρ represent the time elapsed since the start of charging until when the inflection point occurs. I^{start} is the starting current applied for charging, and the final maximum voltage attained in the CV mode is V^{max}, which is constant for a specific battery-charger pair. Let V_t and I_t represent the voltage and the current levels at time $t \in [0, T]$, where T is the total number of time intervals needed for charging. We model the current and the voltage profiles using a piece-wise curve fit. In the CC mode, when $\rho > t$, we have:

$$V_t = V_{max} - \alpha * (\rho - t) - \beta * (\rho - t)^2; \qquad I_t = I^{start}$$

α and β are the quadratic coefficients that fit the voltage curve. These parameters varies for different starting current, battery age, etc. In the CV mode, when $\rho < t$, we have:

$$V_t = V^{max}; \qquad I_t = I^{start} * \gamma^{(t-\rho)}$$

Here, γ is the exponential parameter to fit the current curve. Once again, this parameter vary for different starting current, battery age, etc. The above equations can be combined as:

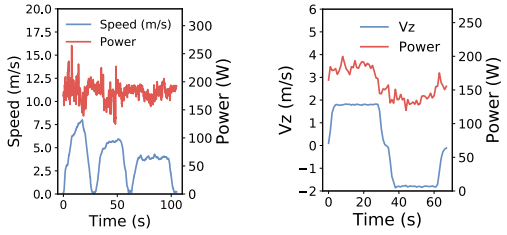
$$V_t = V^{max} - \alpha * (\rho - t)^+ - \beta * ((\rho - t)^+)^2$$
$$I_t = I^{start} * \gamma^{(t-\rho)^+}$$

For estimate the parameters such as ρ, α, β, and γ, we use a Bayesian Inference approach that provides probability density functions for each of these parameters. We describe the complete Bayesian formulation in Table 1. Here, we assume that the error associated with fitting the voltage and current profiles are normally distributed $\mathcal{N}(0, \sigma)$. Also, we use weakly informative prior for the different battery charging parameter based on the range of values relevant to the domain. Using the Markov chain Monte Carlo (MCMC) method, we can utilize the *evidence* (i.e. known quantities such as V_t and $I_t \forall t \in T$) to get the *posterior* distributions for the charging model parameters such as α, β, γ, and ρ starting from the initial *prior* beliefs. The priors were set using the recommendations provided by Gelman et al. [16].

Based on the past charging sessions, we can learn the different charging parameters. From our empirical study, we found that the value of ρ, changes with the age of the battery. However, for same-aged batteries, ρ corresponds to reaching a specific battery energy level — irrespective of the starting current. From the Figure 4(c),

Prior
$\rho \sim \mathcal{U}(0,T), \gamma \sim \mathcal{U}(0,1), \alpha \sim \mathcal{N}(0,2), \beta \sim \mathcal{N}(0,2),$
$\sigma^I \sim Cauchy(0,5), \quad \sigma^V \sim Cauchy(0,5)$
Regression Equation
$\mu_t^V = V^{max} - \alpha * (\rho - t)^+ - \beta * ((\rho - t)^+)^2$
$\mu_t^I = I^{start} * \gamma^{(t-\rho)^+}$
Model Likelihood
$I_d \sim \mathcal{N}(\mu_t^I, \sigma^I), \quad V_d \sim \mathcal{N}(\mu_t^V, \sigma^V)$
Parameter Bounds
$0 < \rho < T, \quad 0 <= \gamma <= 1, \quad -1 <= \alpha, \quad \beta <= 1$

Table 1: Drone charging Bayesian model formulation



(a) Horizontal flights. (b) Vertical flights

Figure 5: Power usage for various drone maneuvers

we observe that both α and β follow a linear relationship with the starting current, reflecting the fact that the higher the charging current, the more rapidly the voltage approaches V^{max} during the CC phase. This linear dependence helps in learning the parameters for a specific charging current by interpolating the parameter values for various other charging currents. On the other hand, the value of γ (≈ 0.91), which pertains to the CV phase, does not change with starting current for the battery. While all (except γ) these parameters change with battery age, the change is gradual. Thus, we can predict quite accurately the time taken to charge the battery.

2.3 Battery Discharging Characterization

Energy consumed to fly the drone depends on the current drawn by the motors to provide the lift and thrust needed to execute the various maneuvers. In turn, these depend on the prevalent environmental conditions (wind speed and direction), the weight of the payload carried, etc. Here, we highlight the impact of two main factors — i) the drone's speed, and ii) weight of the payload — on the battery drain observed. This study involved recording energy consumption data over 100 flights involving a diverse set of maneuvers in different wind conditions and carrying different amounts of payload.

Impact of speed: While flying the drone at a higher speed will result in a shorter time to complete a mission it is unclear if the energy consumed will also reduce. Thus, to illustrate the impact of speed, we show the results from two specific flight paths — (i) linear horizontal flights between two points approximately 130 meters apart at varying speeds, and (ii) vertical flights between the altitudes of 10 and 50 meters above the ground at 2 m/s.

Figure 5(a) shows the horizontal speed and the power consumed of the drone moving between two points while switching its maximum speed from 8 m/s to 6 m/s and finally to 4 m/s. Clearly, the maneuvers are completed in a shorter time at higher speeds. However, the power consumption does not increase proportional to the increase in speeds. Thus, the reduction in energy consumed at higher speeds is the result of a reduction in time without an equivalent increase power. Figure 5(b) presents the relationship between vertical up and down movement and power consumption. Ascending higher is represented by a positive value of the vertical velocity component represented as V_z, whereas a negative value represents the drone descending towards the ground. As shown, working against gravity (while ascending) incurs a 30% higher power consumption on average compared to descending, assisted by gravity.

Observation: *At higher horizontal speeds, the energy usage is lower due to reduction in time without an equivalent increase power. Ascending incurs a 30% higher power compared to descending.*

Impact of weight: Drones often carry an additional payload for specific missions. Thus, it is essential to characterize the energy consumption when carrying different weights. To study the impact of weight in detail, we created two drone workloads, i.e., flight plans involving several maneuvers. We call the first workload *Horizontal Oblique Triangle*, in which the drone moves horizontally from the start point to the endpoint in a straight line separated by a distance of 240m at 10 m/s. After reaching the endpoint, the drone follows an oblique path upwards to the center of this line but at a higher altitude of 20m at 5 m/s. Then the drone returns to the start point by moving in an oblique path downwards at 5m/s. These series of maneuvers forms a triangular flight path. The drone completes three such triangular flight paths for a given payload. This workload is completed in roughly 341 seconds. The second workload is called *Vertical Shift Hover*. Here, the drone ascends from a height of 10 to 30 meters in 10 meter increments. At each point, the drone hovers for 5 minutes before ascending. After hovering for 5 minutes at 30 meters height, the drone descends for landing. This workload takes approximately 920 seconds to complete. Table 2 summarizes our findings of executing these two workloads with different weights attached to the drone. We observe a linear relationship between the energy consumed and the drone weight.

Observation: *The increase in weight results in a linearly proportional increase in energy consumption.*

2.4 Battery Discharging Modeling

Having discussed the impact of speed and payload, we want to predict the instantaneous power consumed while performing various maneuvers. By integrating the predicted power values, we can easily calculate the energy spent on a given set of maneuvers. Below, we discuss our power prediction model in more detail.

We can think of the power required to maintain a stable flight as a combination of several components. Let P_{xy}, and P_z be the power needed to sustain the flight in the horizontal and vertical direction, respectively. Similarly, let P_{wind} be the power demand to resist the deviation due to the wind. Finally, P_{drag} is the power necessary to move against drag. We know that Power = Force*Velocity = Mass*Acceleration*Velocity. Note that while hovering, drone's vertical acceleration (A_z) has to be equal and opposite to acceleration due to gravity, i.e., 9.8 m/s^2 near sea-level.

Workload	Time (s)	Weight (kg)	Average Power (W)
Horizontal	341	1.895	170.01
Oblique		1.995	181.20
Triangle		2.095	190.87
Vertical	920	1.895	172.51
Shift		1.995	181.60
Hover		2.095	206.51

Table 2: Impact of weight on power consumption

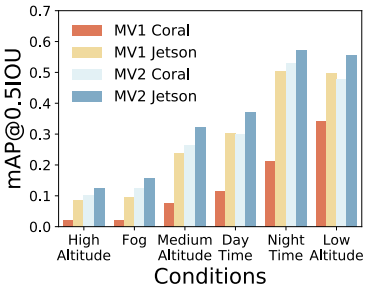


Figure 6: Detection Accuracy across various conditions

$$P_{xy} = m * A_{xy} * V_{xy}; P_z = m * A_z * V_z; P_{wind} = m * A_{xy} * V_{wind};$$

Here, m is the mass of the drone, A_{xy} and V_{xy} are acceleration and velocity in the horizontal direction. Similarly, A_z and V_z are acceleration and velocity in the vertical direction. V_{wind} denotes the velocity of the wind. Further, from drag equations [7], we know that P_{drag} is proportional to $V_{xy} \cdot (V_{wind} + V_{xy})^2$. Note that acceleration and velocities are vector quantities and have both magnitude and direction associated with them. One can find a closed-form expression linking the overall power consumption of the drone, given by P_{total}. However, these components may not be independent of each other. Thus, we choose random forest regression to learn a power prediction model using the four components of power as features to predict instantaneous power use.

2.5 Energy-aware Inference Characterization

We now turn to the inference performed on the drone, specifically for (aerial) vision-based tasks. We characterize the various tradeoffs in executing ML inference workloads, such as object detection on edge devices. We experiment with two edge platforms, a Raspberry Pi 4 with a Google Coral USB Accelerator (Coral+RPi4) and Nvidia Jetson Nano. To characterize the object detection performance, we fine-tune two Single Shot Detectors (SSD) [25], with MobileNetV1 [19] (MV1) and MobileNetV2 [32] (MV2) backbone networks. The detectors are pretrained on MS-COCO dataset [24] which comprises natural and "on-ground" scenes. We conduct our experiments on the UAVDT dataset (details in Section 5.1). This combination of the SSD detection model with the MobileNet backbones is known to be accurate while being efficient and minimizing latency, even outperforming computationally more expensive models like FasterRCNN in UAV scenes [13].

Impact of Altitude and Lighting conditions: Inference tasks are heavily dependent on the characteristics of the data. For instance, flying the drone at a lower altitude would consume more energy, as a smaller area is captured in the field of view of the camera, requiring the drone to travel more distance to cover a target area. A decrease in flying altitude will result in a *quadratic* increase in the distance to be covered by drone, and thus an altitude drop is costly. However, this may ultimately be necessary if the DNN model suffers from unacceptable degradation in accuracy from using the more distant imagery obtained at higher altitudes. Furthermore, changes in weather conditions, such as fog, could reduce visibility and necessitate flying at a lower altitude. We do not consider the direct power consumption attributable to computation, which could consume ≈ 5-10W (e.g., Jetson Nano, etc.) — equivalent to less than 2% of the energy consumed by the drone used in this study.

Figure 6 shows detection accuracy across various altitudes and lghting conditions for the different hardware and model combinations. As expected, increasing altitude of the drone results in a drastic drop in accuracy, due to the smaller size of the objects in view. It is interesting to note that night time conditions yield better accuracy than day time conditions. This is possibly due to models learning to identify vehicles based on their headlights and taillights being on during the night. Also, foggy conditions are especially bad for model performance.

Observation: *Altitude and lighting conditions significantly influence the accuracy of the sensing task.*

2.6 Energy-aware Inference Modeling

Having discussed the tradeoffs between various on-board edge devices, model quantization and drone altitude, we present our framework for predicting accuracy, given the edge device, model quantization level, the lighting condition and the flying altitude of the drone. Consider altitude as a discretized variable alt (with a bin-size of say b metres), while the variables relevant to a model such as model quantization, lighting etc are denoted by o. Here, we model the performance to be a set of piece-wise linear functions of altitude, conditioned on the other variables. We sample the value set for performance and altitude from the experiments we have conducted earlier,

$$perf_o(alt) = m * alt + c \quad alt_b < alt < alt_{b+1}$$

where m and c are computed from the accuracy values of the task after flying the drone at the specified altitude. Given a desired accuracy level, we can now figure out the highest altitude to fly our drone at. At this altitude our sensing will be accomplished at above the expected accuracy level with the largest field of view, thus minimizing the energy consumed.

3 ENERGY-AWARE SCHEDULING

In this section, we leverage the preceding insights to design an energy-aware drone scheduling system. We first summarize the key characteristics of drone operation that guides our design and then explain the key components of our system.

First, to survey a designated area, the drone is flown in straight lines called transects to cover the ground below the flown region with measurements. Turns are only made to move from one transect

to another and do not contribute to the survey. Thus, minimizing the number of these turns is integral to optimizing energy usage as these are wasteful maneuvers. Second, we earlier observed the impact of speed on energy consumption. These characteristics imply that the transects should be as long as possible to maintain sustained high speeds to save energy. Thus, the orientation of generated transects should be such that they are as long as possible, covering the maximum portion of the area of interest, called the optimum sweep direction [20]. Third, we can restrict the charging only up to the inflection point as the battery capacity increase beyond this stage is much slower. Finally, if in a fleet there are drones with old batteries, it would be beneficial to allocate fewer manoeuvres to them as charging older batteries takes a longer time.

3.1 Path Planning

Concave Partitioning: The shape of the area to be surveyed could either be convex or concave. In case of a concave polygonal shape, optimum sweep direction may not guarantee coverage in minimum time due to redundant turns. Therefore, a set of convex sub-regions ensures that for each sub-region we traverse in the optimum sweep direction to minimize energy consumption. Thus, we partition the concave space into a set of convex sub regions [21].

Finding Transects: In this step, we determine the optimum sweep direction for each convex sub-region. We utilize the algorithm described in [6], which determines the direction of the maximum width of the polygon obtained by rotating it at every orientation (iterated at 1 degree in our implementation). Further, if two adjacent sub-regions have similar optimum sweep direction, we merge them into a single sub-region, via a threshold θ_{min}. Transects are formed along the optimum sweep direction. The other factor determining the number of transects is the altitude (which defines the field of view) and it is estimated from the desired accuracy level of the sensing task (as described in Section 2.6). For a given sub-region, and given sweep direction and altitude, there are four ways in which the entire polygon can be surveyed. Our energy model requires acceleration data to estimate energy usage to cover various transects. Thus, we simulated acceleration data for different transects.

Connecting Sub-Regions: After obtaining the transects, we need to traverse between the sub-regions in such a way that energy consumption is minimized. We call the inter-region traversal distance as transition distance. For connecting the sub-regions, we should minimize the sum of transition distance from one sub-region to another. A brute-force approach compares the total transition distance for all the possible orderings of the sub-regions and for each permutation of four possible traversals of a sub-region. This approach is computationally expensive as run-time is exponential to number of sub-regions. Thus, we first decide the order of visiting every sub-region. We model the sub-region ordering problem as a graph problem. Each sub-region is considered as a vertex, and the edge weights are the inter-centroid euclidean distance between them. The proposed formulation is equivalent to the travelling salesman problem (or the source-t min cost Hamiltonian Path problem) in a *metric graph* (due to the definition of edge weights). However, the formulation is NP-Hard and a polynomial time approximation heuristic exists, the *Christofides algorithm* [9, 18]. The upper bound is 3/2 for the general case and 5/3 for the s-t case. These are currently the best known bounds for the problem. This algorithm gives us the order of traversal of the sub-regions.

We have obtained the ordering of traversal between the sub-regions, however, there are four possibilities while traversing each sub-region. The traversed path can be found by comparing the distance between the four possible end points of the first convex sub-region to the four possible end-points of the second sub-region. After connecting the points with minimum distance, the two sub-region are combined and there are only two possible end-points of the combined sub-region. Then, compare the distance between the two end points of the combined sub-region with the four end points of the third sub-region and so on to get the final path.

3.2 Task Scheduling

We formulate an integer linear programming program (described in Table 3) to solve task division among the available drones.

Let $\tau \in \Gamma$ be an ordered list of transects and $d \in D$ be the set of available drones. Each drone d takes time $T_{\tau,d}$ to cover transect τ and consumes energy $E_{\tau,d}$. As we know the traversed path and can simulate the drone dynamics, $E_{\tau,d}$ can be estimated by calculating the instantaneous power from simulated acceleration using the model described in Section 2.4, and then integrating the predicted power values. The drone d has to travel from the ending of transect τ to the start of transect $\tau + 1$ (take a turn), and for this it has to waste time $\hat{T}_{\tau,\tau+1,d}$ and spend energy $\hat{E}_{\tau,\tau+1,d}$. $\hat{E}_{\tau,\tau+1,d}$ is again estimated using the model described in Section 2.4, similar to how $E_{\tau,d}$ is estimated. Each drone is allocated a contiguous list of transects to decrease the number of turns (which are wasteful as noted earlier). Let the battery capacity of drone d be denoted by B_d and time taken to charge the exhausted battery till the constant current charging mode be C_d. We can reliably estimate the battery profiles (B_d and C_d) using the models described in Section 2.2 and Section 2.4 as the battery charging process is linear (CC mode). Further, we want to know if the drone has visited the transect τ. Let the binary indicator variable $V_{\tau,d}$ denote if d has visited τ, while another binary indicator variable $\hat{V}_{\tau,\tau+1,d}$ denote if d has covered the distance between τ and $\tau + 1$. In our formulation shown in Table 3, we would want to minimize the time ($\mathbb{M}^{time}$) in which the given task can be completed. To minimize $\mathbb{M}^{time}$, we only need to look at the drone that takes the maximum time to complete its task. Let E_d^{total} be the total energy and T_d^{total} be the total time taken by d to cover the task assigned to it. These two quantities can be defined in terms of the indicator variables,$V_{\tau,d}$, $\hat{V}_{\tau,\tau+1,d}$, the time $\hat{T}_{\tau,\tau+1,d}$ and energy $\hat{E}_{\tau,\tau+1,d}$.

In the formulation, constraint (1) ensures that the variables $V_{\tau,d}$ and $\hat{V}_{\tau,\tau+1,d}$ are binary, i.e., a transect or transition between transects is either completed or not completed. Constraint (2) ensures that the intermediary distance between τ and $\tau + 1$ is considered as visited by drone d, if and only if the same drone visits both the transect τ and $\tau + 1$. Such a constraint ensures that a drone covers contiguous transects. Also, any transect must be visited only once in total (see constraint (3)). As we handle cases where all drones are used for the task, the total number of transitions between transects is constrained by the number of transects Γ subtracting the number of drones D, and is represented by (4). (6) denotes the number the

minimize $\mathbb{M}^{time}$

s.t.

(1) $V_{\tau,d} \in \{0,1\}, \quad \hat{V}_{\tau,\tau+1,d} \in \{0,1\} \qquad \forall \tau \in \Gamma, d \in D$

(2) $\hat{V}_{\tau,\tau+1,d} <= V_{\tau,d}, \quad \hat{V}_{\tau,\tau+1,d} <= V_{\tau+1,d} \qquad \forall \tau \in \Gamma, d \in D$

(3) $\sum_{d \in D} V_{\tau,d} = 1 \qquad \forall \tau \in \Gamma$

(4) $\sum_{\tau=1}^{|\Gamma|-1} \sum_{d \in D} \hat{V}_{\tau,\tau+1,d} = |\Gamma| - |D|$

(5) $E_d^{total} = \sum_{\tau \in \Gamma} V_{\tau,d} \cdot E_{\tau,d} + \sum_{\tau=1}^{|\Gamma|-1} \hat{V}_{\tau,\tau+1,d} \cdot \hat{E}_{\tau,\tau+1,d} \qquad \forall d \in D$

(6) $N_d = E_d^{total} / B_d \qquad \forall d \in D$

(7) $T_d^{total} = \sum_{\tau \in \Gamma} V_{\tau,d} \cdot T_{\tau,d} + \sum_{\tau=1}^{|\Gamma|-1} \hat{V}_{\tau,\tau+1,d} \cdot \hat{T}_{\tau,\tau+1,d} + N_d \cdot C_d \qquad \forall d \in D$

(8) $T_d^{total} <= \mathbb{M}^{time} \qquad \forall d \in D$

Table 3: Task scheduling optimization

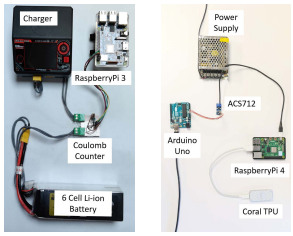


Figure 7: Energy monitoring setups

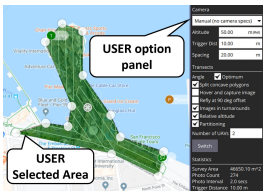


Figure 8: UI for *Smart Survey* feature in QGC

times that drone d has to charge, to complete its task. Constraint (5) and (7) represent the total energy consumed and time taken by a drone d to complete the task allocated to it. Constraint (8) denotes that the total time taken to finish all the tasks (i.e. $\mathbb{M}^{time}$) is at most equal to the maximum time taken by any drone in $d \in D$.

4 IMPLEMENTATION

4.1 Open Source GCS

Existing Ground Control Station (GCS) softwares are highly sophisticated and versatile as they provide support for various platforms, such as Pixhawk [27]. They are capable of handling detailed telemetry and uploading flight plans using multiple connection options such as TCP/UDP, etc. Thus, we implemented our system as an extension in QGroundControl (QGC). QGC is open-source, and the most widely used GCS with a significant active contributors base. Specifically, we implemented our algorithm as an additional tab called *Smart Survey*, which is an improvement over the existing *Survey* tab that implements a basic path planning algorithm with no energy considerations. Figure 8 shows the UI design for our *Smart Survey* tab. The entire implementation involves around 2000 lines of C++ code and around 400 lines of QML code (our open-source implementation: https://github.com/t-sriyen/qgroundcontrol). For simulating accelerometer values for the energy predictions, we implement Allan variance algorithm [1].

4.2 Energy Model Toolkit

To learn the battery charging parameters, we used *PyStan*, a Bayesian modelling library with several MCMC samplers. We use *Sklearn*, a Python-based machine learning library to train our energy discharging models. As part of this work, we release an open-source toolkit consisting of the battery charging and discharging models. Further, the toolkit also contains a framework to calculate the optimal altitude that minimizes energy usage of drones for any UAV-based sensing, given enough domain information. Moreover, we also release our detailed drone flights datasets containing GPS and IMU logs along with flight plans (our open-source toolkit: https://github.com/t-sriyen/DroneEnergyModeling).

4.3 Energy-Aware Model Inference

Our object detection models are trained and fine-tuned on a server equipped with a Nvidia Tesla K80 equipped with 64 GB RAM using Tensorflow 1.12 with the Tensorflow Object Detection API. Our trained models (SSD-MV1 and SSD-MV2) cannot be executed as-is on either edge platforms and thus were converted to their respective proprietary formats — i.e., tflite model for Google Coral and UFF model for Jetson Nano.

4.4 Hardware Implementation

For our experiments, we use TATTU Premium 22.2V 6-cell LiPo batteries with an 8000mAh capacity and discharging rate of 15C. We used a quadcopter with an onboard Pixhawk flight controller along with a RaspberryPi 3B+ as a companion computer to log IMU data, along with energy used to perform different drone manoeuvres. For observing charging trends of the battery we used a *Turnigy Reaktor D6 Pro Duo* charger and a coulomb counter, designed using IC *LTC2944*[3]. Figure 7(a) shows this energy monitoring setup. For experiments involving ML inference on the edge platforms, our power logging setup involved a current sensor (ACS 712) and an SMPS regulated at 5 volts, shown in Figure 7(b).

5 EVALUATION METHODOLOGY

5.1 Datasets

ML Inference: We conduct our experiments on the UAVDT [13] object detection dataset, which consists of 80,000 frames of UAV flights. The frames are labelled with object bounding boxes of vehicles and other attributes — such as weather conditions, flying altitude, etc. — that are useful in characterizing performance of object detection algorithms in various real-world scenarios.

Path Planning: We created a dataset of prominent public places, including parks, piers, and city squares, etc. spread across four cities — New York, San Francisco, Bangalore, and London. These areas vary greatly in shapes and sizes, ranging from 0.2 to 5.7 sq. km. These sites form the set of shapes given as input to the algorithms. We will release this dataset as part of the toolkit to aid benchmarking.

5.2 Metrics

Energy Modeling: For this, we use Mean Absolute Percentage Error (MAPE) as our performance metric.

ML Inference: We measure the accuracy of inference using Mean Average Precision with $IOU \geq 0.5$ (mAP@0.5) [14].

Path Planning: We characterize the input shapes using convexity, which is defined as the area of the shape divided by the area of

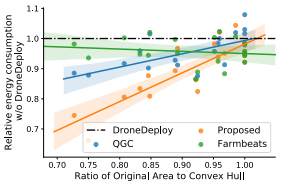


Figure 9: Relative energy improvement of proposed method for shapes with different convexity

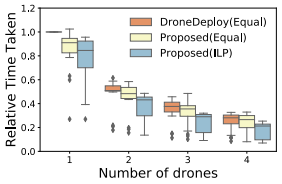


Figure 10: Relative time taken with increasing number of drones

convex hull of the shape [35]. We characterize the performance by measuring both energy and the time taken for a planned path.

5.3 Baselines

Energy Modeling: We compare the performance of our energy discharge model with the start of the art [36], which modelled the drone energy consumption as a linear combination of several factors that include the drone weight, wind speed, and velocity, etc.

Path Planning: We compare the performance of our path planning approach with three existing algorithms - DroneDeploy(East-to-west algorithm) [12], QGC [40] and FarmBeats [38] using energy consumption as the metric. DroneDeploy generates sweeping patterns from east to west or west-to-east regardless of the shape of the surveying area. FarmBeats generates the path that minimizes number of waypoints. In QGC, the sweep direction is left to the user to select. In our evaluation, we kept the sweep direction fixed from north-to-south.

Task Scheduling: We consider the equal transect division algorithm as our baseline, i.e. the transects are equally divided amongst all the drones available to us. This is in contrast to our system wherein a linear programming model is proposed.

6 EXPERIMENTAL RESULTS

6.1 Energy Modeling

As discussed earlier, we model the power consumption using a Random Forest (RF) regression consisting of several components (i.e. P_xy, P_z, P_{wind}, P_{drag}) as features. We used over 100 different drone flight data and categorized them into four groups. The first two groups are called *vertical* and *horizontal* flight datasets. As the name suggests, these flights only involve drones either moving in horizontal or vertical directions. The *vertical triangle* group is similar to our workload discussed in section 2.4. In the dataset group called *full*, every flight path was included.

Within each group, we performed a leave-one-out cross-validation approach to characterize the performance of our model (Proposed (RF)) compared to Tseng et al [36] (see Figure 11). Also, with more data points, as is the case in the *full* dataset group, the performance improves to an average MAPE of 5.6% for the predicted energy consumption across all manoeuvres. Clearly, our model significantly outperforms the model presented by Tseng et al [36], as our average MAPE values are 2.5x lower than theirs in the average case. Our proposed method performs even better in horizontal and vertical manoeuvres, where we report 3 to 3.5x improvements in MAPE.

We conducted a detailed evaluation of our charging models and observed an average MAPE of around 1.94% in predicting the transition point between the CC and the CV mode. Further, the R^2 fit of the charging models for voltage and current over 100 sessions never dropped below 0.98.

Summary: *Our energy model demonstrates an average MAPE of 5.6% — a 2.5x improvement over the state of the art.*

6.2 Path Planning Evaluation

To evaluate our system's performance on path planning tasks, we compare it with three baseline algorithms (DroneDeploy, QGC, Farmbeats) on a dataset discussed in section 5. In many different real-world surveying tasks, such as mapping districts, river deltas, etc., the underlying shape is not convex. Thus, in Figure 9 we show the graph of the relationship between relative energy consumption w.r.t DroneDeploy and Area Convexity. Area Convexity is defined as the ratio of the survey area to the area of its convex hull. When the convexity is less than 0.95, the proposed algorithm shows significant improvement in energy consumption. The percentage improvement in energy when the convexity lies between 0.85 to 0.95 is 11.29%. The percentage improvement in energy consumption when the convexity is between 0.7 and 0.85 is 21.14%. The percentage improvement in the energy consumption when the convexity is greater than 0.95 is 3.09% which is similar to the Farmbeats algorithm. Hence, when the shape is not convex, it is beneficial to partition the shape. When the shape is nearly convex, the paths generated by FarmBeats and our algorithm are very similar. Hence the energy consumption does not change too much.

Summary: *Our system outperforms the three baseline algorithms in energy consumption. The improvement is more pronounced when the area convexity is lower.*

6.3 Task Scheduling Evaluation

To evaluate the performance of our proposed system for multi-drone scheduling, we use the same dataset used in the above path planning evaluation. We vary the number of drones and generate the path using our algorithm and the equal transect division algorithm. Figure 10 shows the relative time taken by three approaches - (i) *DroneDeploy (Equal)*, where the path is generated using DroneDeploy and task division is done using equal transect splitting algorithm, (ii) *Proposed (Equal)*, where our system does path planning and task division is done using equal transect splitting algorithm and (iii) *Proposed (ILP)*, where the path planning and task division is done by our system using the proposed linear programming approach. As expected, the time taken decreases with an increase in the number of drones. In all the cases, the *Proposed (ILP)* based task division algorithm results in the least time taken. Specifically, with just two drones, the average reduction in time taken to complete surveying tasks in our dataset by *Proposed (ILP)* compared to *DroneDeploy (Equal)* the 35.97%. With four drones, the reduction in time is 46.91%. This is because the *Proposed (ILP)* algorithm considers various factors — such as charging profile, energy consumption for various maneuvers, time taken per transect, etc. — to divide the task among available drones optimally.

Summary: *Our proposed system achieves a greater reduction in mission times with multiple drones than the baselines.*

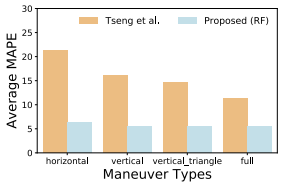


Figure 11: MAPE of our proposed model with different regression algorithms

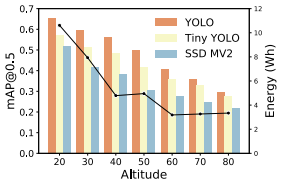


Figure 12: Performance of the object detection models at different altitudes

6.4 Impact of altitude

To study the effect of the altitude on the accuracy of object detection models and the consequent impact on task-level energy consumption, we use UAVDT [13] dataset for identifying vehicles. We choose three object detection models YOLO [29], Tiny YOLO [29] and SSD-MV2 [25, 32] to compare their accuracy at different altitudes. Figure 12 shows the variation of mAP@0.5 of different models and task-level energy consumption with the altitude of drone. At higher altitudes, the field of view of the camera has increased coverage. Hence, the flight time needed to complete a surveying task could reduce. We observe that object detection models have lower accuracies at higher altitudes due to reducing object size (in terms of the number of pixels). YOLO has maximum accuracy out of the three models used at all altitudes. Thus, if an embedded device can run an expensive model like YOLO, it will yield greater accuracy. Further, we see that the energy consumption decreases with the increase in altitude, but the decrease is not uniform. Instead, the energy decreases in a step-wise manner. This behavior is because the the flight time and hence the energy consumption depends on the cumulative length of the transects, and an increase in altitude does not necessarily mean a decrease in the number of transects. In some cases, the number of transects decreases with an increase in altitude, while in other cases it remains unchanged, and so the energy consumption also remains essentially unchanged.

Summary: *Increasing altitude hurts object detection accuracy. However, energy consumption decreases in a step-wise manner.*

7 CASE STUDY: PEOPLE COUNTING

This section describes a people counting case study with our system.

Setup: We conducted this experiment using two drones mounted with a camera and an onboard computer (Jetson Nano). The application objective was to detect people walking on two adjacent sports grounds of area 106,563 and 112,601 *sq.ft.* in a university (anonymized). Our open-source ground control software (QGC with energy-aware path planning and scheduling) allocated flight paths to the drones to repeatedly cover each ground until the batteries are exhausted. Several students were asked to walk on the ground so that the drones could detect them. The application entailed continuous monitoring of the sports grounds until two batteries are exhausted by each of the two drones. In this case study, we had two new batteries (<10 charge-discharge cycles) and two older ones (>100 cycles). Each drone started the mission with a newer battery, and then we swapped it with older ones. The onboard computers have a pre-trained tinyYOLO object detector on the previously collected people detection dataset. As discussed in section 2.6, we selected an altitude of 15 meters for the flight to have a high object detection accuracy (*F1 score* >0.8 in the training set).

Observations: The two drones covered a total distance of 10.358 km in 44.54 minutes and 9.73 km in 41.1 minutes, respectively. The first ground was covered 14.2 times, while the second was covered 15.75 times. Overall, the object detector identified people with a precision of 0.8 and a recall of 0.89. Overall, the mean average precision (mAP@0.5) of 83.65%.

8 RELATED WORK

8.1 Energy Modeling in Drones

Optimal usage of a drone's battery can significantly increase in its flight time. Past works have looked at creating an energy model for the drone [10, 36]. [10] presents an energy model derived from real-world measurements and then uses it to derive the path planning algorithm. [36] propose a regression model of energy consumption for drones. However, these systems do not talk about energy-efficient task division among multiple drones across multiple charging events — a fundamental requirement in large-scale surveying tasks. Moreover, none of the existing work considers the interplay between energy consumption and the sensing quality.

8.2 Drone path planning and scheduling

Most approaches to solve drone path planning problem involve dividing the survey area into cells. Following which, a travelling salesman algorithm is applied to find the coverage path [39]. Further, several algorithms find optimal sweep direction to minimize the number of turns while surveying an area [6, 20]. All these works restrict themselves to only the geometrical aspects of the survey area and do not consider factors such as battery capacity, weight, etc. Recently, an approach to energy-aware path planning was introduced [15]. However, the treatment was quite preliminary and did not consider factors such as wind, payload weight, etc. Neither did it look at the impact of the plan on sensing accuarcy. There are also many powerful mission planning and ground control stations(GCS) having various capabilities. QGround Control [40] is the most used open-source GCS having an ample amount of contributors on GitHub. UGCS [2] is closed-source but has some great features compared to open-source counterparts like multiple drone support even in simulation, 3D surveying, etc. None of these have implemented energy-focused path planning and leaves the choice of the mission flight altitude to the user.

8.3 Drone orchestration systems

Work in several areas (computer systems, robotics, etc.) has looked at developing drone orchestration, i.e., a system to manage a fleet of drones for a set of applications [17, 26, 28, 37]. AnDrone [37] is a virtual drone computing platform that attempts to use drone-as-a-service to make it accessible like a cloud resource. Through this, a single physical drone can run multiple *virtual drones* instances for different applications simultaneously. Voltron [28] presents a programming model to manage a fleet of drones using programming constructs. More recently, BeeCluster [17] looked at provides APIs

to abstract complex drone tasks. However, none of these systems considers battery energy modelling (charging and discharging) and its implications in reducing mission times. Moreover, none of these focuses on the interplay between application objectives — i.e., model inference accuracy in surveying tasks — and the choice of altitude, with the resulting impact on the flight time for a mission and the battery drain. However, an interesting future direction would be integrating our energy-aware path planning and task scheduling into these drone orchestration systems.

9 CONCLUSION

Realizing the full potential of drones necessitates managing the limited energy resources carefully while maintaining sensing accuracy. In this paper, we presented a holistic view on drone scheduling by considering tradeoffs across several factors such as battery characteristics, variation in drone imagery, etc. We conducted an extensive empirical study and explored the impact of altitude and lighting on sensing accuracy. While uncovering several real-world battery characteristics, we built accurate models for learning charge and discharge profiles. For example, our energy model could predict energy consumed by various maneuvers with an error of 5.6% — comfortably beating earlier state-of-the-art baseline. We use the insights from our empirical studies and the learned models to design our holistic, energy-aware, multi-drone scheduling system that reduces the energy consumed by 21.14% and time taken by 46.91%. We implemented our system as an open-source extension to a popular drone ground control software. We presented a people counting case study that performed with an accuracy of 83.65%.

REFERENCES

[1] [n. d.]. https://github.com/Aceinna/gnss-ins-sim.
[2] [n. d.]. https://www.ugcs.com/.
[3] [n. d.]. LTC2944. https://www.analog.com/media/en/technical-documentation/data-sheets/2944fa.pdf.
[4] Federal Aviation Administration. 2016. Summary of Small Unmanned Aircraft Rule (Part 107). https://www.faa.gov/uas/media/Part_107_Summary.pdf. [Online; accessed 10-August-2019].
[5] European Union Aviation Safety Agency. 2019. Civil drones (Unmanned aircraft). https://www.easa.europa.eu/easa-and-you/civil-drones-rpas. [Online; accessed 10-August-2019].
[6] J. F. Araújo, P. B. Sujit, and J. B. Sousa. 2013. Multiple UAV area decomposition and coverage. In *2013 IEEE Symposium on Computational Intelligence for Security and Defense Applications (CISDA)*. 30–37. https://doi.org/10.1109/CISDA.2013.6595424
[7] Glenn Reesearch Center at National Aeronautics and Space Administration. 2019. https://www.grc.nasa.gov/www/k-12/airplane/drageq.htmls. [Online; accessed 10-August-2019].
[8] Anirudh Badam, Ranveer Chandra, Jon Dutra, Anthony Ferrese, Steve Hodges, Pan Hu, Julia Meinershagen, Thomas Moscibroda, Bodhi Priyantha, and Evangelia Skiani. 2015. Software defined batteries. In *Proceedings of the 25th Symposium on Operating Systems Principles*. ACM, 215–229.
[9] Nicos Christofides. 1976. Worst-Case Analysis of a New Heuristic for the Travelling Salesman Problem.
[10] Carmelo Di Franco and Giorgio Buttazzo. 2016. Coverage Path Planning for UAVs Photogrammetry with Energy and Resolution Constraints. *J. Intell. Robotics Syst.* 83, 3–4 (Sept. 2016), 445–462. https://doi.org/10.1007/s10846-016-0348-x
[11] DJI. 2020. PHANTOM 4 PRO Specs. https://www.dji.com/sg/phantom-4-pro/info. [Online; accessed 10-January-2020].
[12] DroneDeploy. 2020. https://www.dronedeploy.com.
[13] Dawei Du, Yuankai Qi, Hongyang Yu, Yifan Yang, Kaiwen Duan, Guorong Li, Weigang Zhang, Qingming Huang, and Qi Tian. 2018. The unmanned aerial vehicle benchmark: Object detection and tracking. In *Proceedings of the European Conference on Computer Vision (ECCV)*. 370–386.
[14] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. 2010. The pascal visual object classes (voc) challenge. *International journal of computer vision* 88, 2 (2010), 303–338.
[15] C. D. Franco and G. Buttazzo. 2015. Energy-Aware Coverage Path Planning of UAVs. In *2015 IEEE International Conference on Autonomous Robot Systems and Competitions*. 111–117. https://doi.org/10.1109/ICARSC.2015.17
[16] Andrew Gelman et al. 2006. Prior distributions for variance parameters in hierarchical models (comment on article by Browne and Draper). *Bayesian analysis* 1, 3 (2006), 515–534.
[17] Songtao He, Favyen Bastani, Arjun Balasingam, Karthik Gopalakrishna, Ziwen Jiang, Mohammad Alizadeh, Hari Balakrishnan, Michael Cafarella, Tim Kraska, and Sam Madden. 2020. BeeCluster: drone orchestration via predictive optimization. In *Proceedings of the 18th International Conference on Mobile Systems, Applications, and Services*. 299–311.
[18] J. A. Hoogeveen. 1991. Analysis of Christofides' Heuristic: Some Paths Are More Difficult than Cycles. *Oper. Res. Lett.* 10, 5 (July 1991), 291–295. https://doi.org/10.1016/0167-6377(91)90016-I
[19] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. 2017. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861* (2017).
[20] W. H. Huang. 2001. Optimal line-sweep-based decompositions for coverage algorithms. In *Proceedings 2001 ICRA. IEEE International Conference on Robotics and Automation (Cat. No.01CH37164)*, Vol. 1. 27–32 vol.1. https://doi.org/10.1109/ROBOT.2001.932525
[21] J Mark Keil and Jack Snoeyink. 1998. On the time bound for convex decomposition of simple polygons. Citeseer.
[22] Majid Khonji, Mohammed Alshehhi, Chien-Ming Tseng, and Chi-Kin Chau. 2017. Autonomous inductive charging system for battery-operated electric drones. In *Proceedings of the Eighth International Conference on Future Energy Systems*. ACM, 322–327.
[23] Qi Li, Keyang Yu, and Dong Chen. 2020. Automatic Damage Detection on Rooftop Solar Photovoltaic Arrays. In *Proceedings of the 7th ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation*. 332–333.
[24] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. 2014. Microsoft coco: Common objects in context. In *European conference on computer vision*. Springer, 740–755.
[25] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. 2016. Ssd: Single shot multibox detector. In *European conference on computer vision*. Springer, 21–37.
[26] Sara Mahmoud, Nader Mohamed, and Jameela Al-Jaroodi. 2015. Integrating uavs into the cloud using the concept of the web of things. *Journal of Robotics* (2015).
[27] Lorenz Meier, Dominik Honegger, and Marc Pollefeys. 2015. PX4: A node-based multithreaded open source robotics framework for deeply embedded platforms. In *International Conference on robotics and automation (ICRA)*. IEEE, 6235–6240.
[28] Luca Mottola, Mattia Moretta, Kamin Whitehouse, and Carlo Ghezzi. 2014. Team-level programming of drone sensor networks. In *Proceedings of the 12th ACM Conference on Embedded Network Sensor Systems*. 177–190.
[29] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. 2016. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 779–788.
[30] Angélica F Resende, Maria TF Piedade, Yuri O Feitosa, Victor Hugo F Andrade, Susan E Trumbore, Flávia M Durgante, Maíra O Macedo, and Jochen Schöngart. 2020. Flood-pulse disturbances as a threat for long-living Amazonian trees. *New Phytologist* 227, 6 (2020), 1790–1803.
[31] GOLDMAN Sachs. 2017. Drones reporting for work.
[32] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. 2018. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 4510–4520.
[33] Richard Schiffman. 2014. Drones flying high as new tool for field biologists.
[34] Mario Silvagni, Andrea Tonoli, Enrico Zenerino, and Marcello Chiaberge. 2017. Multipurpose UAV for search and rescue operations in mountain avalanche events. *Geomatics, Natural Hazards and Risk* 8, 1 (2017), 18–33.
[35] Milan Sonka, Vaclav Hlavac, and Roger Boyle. 1993. *Image processing, analysis, and machine vision*. Chapman and Hall.
[36] Chien-Ming Tseng, Chi-Kin Chau, Khaled M. Elbassioni, and Majid Khonji. 2017. Flight Tour Planning with Recharging Optimization for Battery-operated Autonomous Drones. *ArXiv* abs/1703.10049 (2017).
[37] Alexander Van't Hof and Jason Nieh. 2019. AnDrone: Virtual Drone Computing in the Cloud. In *Proceedings of the Fourteenth EuroSys Conference 2019*. ACM, 6.
[38] Deepak Vasisht, Zerina Kapetanovic, Jongho Won, Xinxin Jin, Ranveer Chandra, Sudipta Sinha, Ashish Kapoor, Madhusudhan Sudarshan, and Sean Stratman. 2017. Farmbeats: An iot platform for data-driven agriculture. In *14th USENIX Symposium on Networked Systems Design and Implementation (NSDI 17)*. 515–529.
[39] J. Xie, L. R. G. Carrillo, and L. Jin. 2019. An Integrated Traveling Salesman and Coverage Path Planning Problem for Unmanned Aircraft Systems. *IEEE Control Systems Letters* 3, 1 (Jan 2019), 67–72.
[40] E Zurich. 2013. Qgroundcontrol: Ground control station for small air land water autonomous unmanned systems.