# Energy-Aware Planning-Scheduling for Autonomous Aerial Robots

Adam Seewald[1], Héctor García de Marina[2], Henrik Skov Midtiby[1], and Ulrik Pagh Schultz[1]

***Abstract*—This letter presents an online planning-scheduling approach for battery-powered autonomous aerial robots. The approach plans a coverage path and schedules simultaneously onboard computational tasks. It further derives a novel variable coverage motion robust to airborne constraints and an empirically motivated energy model. The model is enhanced with the energy contribution of the schedule, utilizing a previously proposed automatic modeling tool. Data show how an initial flight plan is adjusted online in the function of the available battery, accounting for uncertainty. Here, the proposed approach additionally remedies possible in-flight failure in case of unexpected battery drops, e.g., due to adverse atmospheric conditions, and increases the overall fault tolerance.**

***Index Terms*—Aerial Systems: Perception and Autonomy, Planning, Planning under Uncertainty**

## I. Introduction

Use cases involving aerial robots span broadly. They compromise diverse planning and scheduling strategies and often require high autonomy under strict energy budgets. An instance is coverage path planning (CPP) in the literature [1], [2], consisting of, e.g., an aerial robot visiting every point in a given space [3], running assigned computational tasks. Here, the aerial robot might detect ground patterns and notify other ground-based actors with little human interaction. Such use cases arise in, e.g., precision agriculture [4], where harvesting involves ground [5]–[10], damage prevention aerial robots [11], [12]. Microcontrollers and heterogeneous computing hardware [13] (i.e., with CPUs and GPUs) running power-demanding computational tasks are frequently mounted onto the robots in these and many other scenarios [14]–[17]. We refer to computational tasks that can be scheduled with an energy impact as computations. We are interested in the energy optimization of motion plans and computations schedules in-flight and refer to it as energy-aware planning-scheduling. Studies that deal with planning-scheduling energy awareness for mobile robots are scarce [18]–[21] and focused on ground-based robots [13], [20]–[23]. Yet, aerial robots are particularly affected by various energy considerations. Indeed it would be generally required to land to recharge the battery. Such a state of practice has prompted us to propose the planning-scheduling approach for autonomous aerial robots, combining the past body of knowledge but addressing aerial robots' peculiarities such as the atmospheric, battery, and turning radius constraints. Numerical simulations and experimental data of both static and dynamic plans and schedules show improved power savings and fault tolerance with the aerial robots remedying in-flight failures. Fig. 1 illustrates the intuition: an aerial robot flies full plan and schedule (i), that is optimized respecting the battery state (ii), and altered due to, e.g., unexpected battery defects (iii).

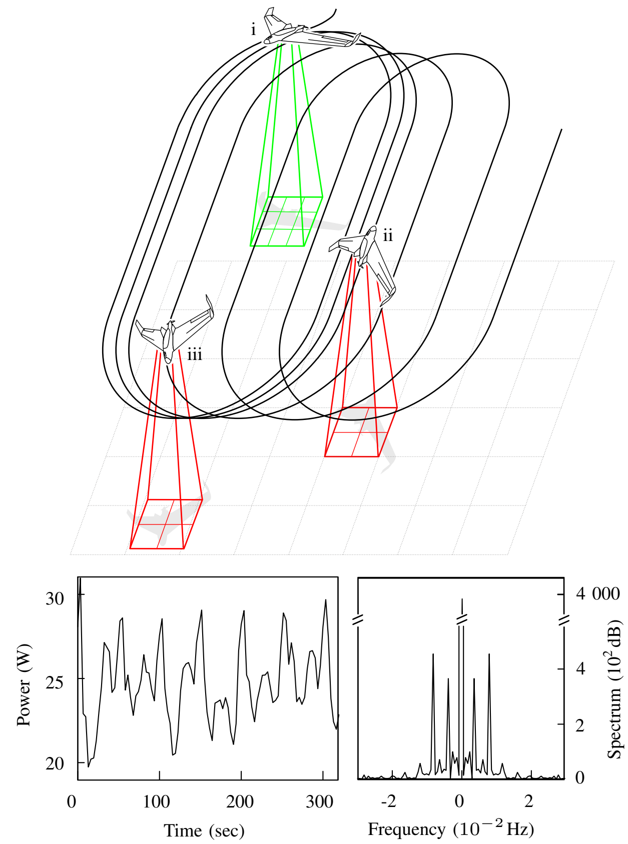



Fig. 1: An initial plan (in i) is re-planned online changing the resolution, detection rate, or other computational aspects (in ii) and changing the number of fly-byes or other motion aspects (in iii). Follow the collected energy data of a physical aerial robot flying the static coverage.

There are numerous planning approaches applied to a variety of robots. An instance is an algorithm selecting an energy-optimized trajectory [24] by, e.g., maximizing the operational time [25]. Many apply to a small number of robots [26] and focus exclusively on planning the trajectory [27], despite compelling evidence of the energy influence of consumptions [13], [18], [19], [21]. In view of the availability of powerful heterogeneous computing hardware [28], the use of computations is further expected to increase in the foreseeable future [29]–[31]. In this context, planning-scheduling energy awareness is a recent research direction. Early studies (2000–2010) varied

Manuscript received: Month, Day, Year; Revised Month, Day, Year; Accepted Month, Day, Year.

This paper was recommended for publication by Editor Editor A. Name upon evaluation of the Associate Editor and Reviewers' comments.

[1]A. S., H. S. M., and U. P. S. are with the Unmanned Aerial Systems Center, Mærsk Mc-Kinney Møller Institute, University of Southern Denmark, Odense. Email: ads@mmmi.sdu.dk;

[2]H. G. is with the Faculty of Physics, Department of Computer Architecture and Automatic Control, Universidad Complutense de Madrid, Spain.

Digital Object Identifier (DOI): see top of this page.

hardware-dependent aspects, e.g., frequency, voltage, along with motion aspects, e.g., motor and travel velocities [13], [18], [22], [32] whereas the literature from the past decade derives energy-aware plans-schedules in broader terms. These include simultaneous considerations for planning-scheduling in perception [21], localization [20], navigation [23], [33], [34], and anytime planning [19].

Our focus is on fixed wings, i.e., airborne robots where wings provide lift, propellers forward thrust, and control surfaces maneuvering. Here, motion and computations energies are within an order of magnitude from each other [35]. Indeed there are other classes where planning-scheduling energy awareness leads to irrelevant savings, i.e., when the motion energy contribution far outreaches the computations and vice-versa. The occurrence frequently happens with rotary-wing aerial robots (e.g., quadrotors or quadcopters, hexacopters, etc.) and lighter-than-air aerial robots (e.g., blimps). It is a common theme with wider planning-scheduling literature, focusing on energy-efficient ground-based robots such as Pioneer 3DX [13], [23], [33], [34], ARC Q14 [20], [21], and Pack-Bot UGV [22].

**Contribution.** We extend the literature on planning-scheduling to CPP with aerial robots, model the overall and computations energies, and battery evolution, and derive a variable coverage motion for constrained systems, e.g., fixed wings.

To guarantee energy awareness, our approach uses optimal control where both the paths and schedules variations are trajectories, varying between given bounds (i.e., physical constraints of the robot and computing hardware, the desired quality of the coverage, etc.). Numerous past planning-scheduling studies also employ optimization techniques [18], [20], [21], [32], whereas some others greedy [13], [19], [22] and reinforcement learning-based approaches [23], [33], [34]. Both the variations trajectories are derived for future time instants employing computations and overall energies and battery models. The energy model for the computations uses regressional analysis from our earlier study on heterogeneous computing hardware energy modeling [36], [37], whereas the battery an equivalent circuit model (ECM) from the literature [38]–[40]. The overall wraps the past two in a cohesive model that uses differential and periodic modeling to predict future energy behavior of different plans and schedules. In Fig. 1, collected energy data (bottom left) and spectrum analysis (right) of a fixed-wing aerial robot flying CPP motivates the overall energy model: the evolution is periodic, an observation we exploit in Section III.

The remaining sections of the letter are then organized as follows. Section II provides basic constructs, such as the concepts of the stages, path functions, triggering and final points, and plan, as well as the problem formulation. Section IV describes in detail the methodology of planning-scheduling. Section V presents the results and showcases the performances, and Section VI concludes and provides future perspectives.

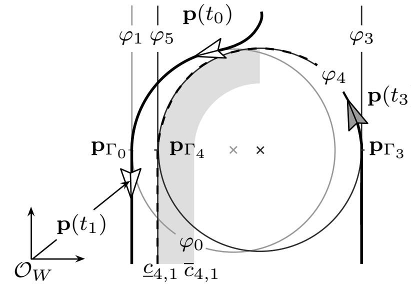


Fig. 2: Definitions II.1–4 on a slice of the plan $\Gamma$. $\mathbf{p}_{\Gamma_i}$ are triggering points in which proximity happens change of stages $\Gamma_i$. Each contains a path function $\varphi_i$ and $j$ parameters to alter the path and schedule $c_{i,1}, \dots, c_{i,j}$.

## II. Problem Formulation

Before defining the problem of energy-aware planning-scheduling in Section II-B, Section II-A provides necessary preliminaries. For CPP and, e.g., pattern detections in precision agriculture, we assume that aerial robot travels a *plan* composed of *stages*. At each stage, the aerial robot travels a path and runs a schedule on the computing hardware. Both are to be altered in Section IV within given boundaries with *path*- and *computations*-specific *parameters*.

### A. Preliminaries

**Definition II.1** (Stage)**.** Given a generic point $\mathbf{p} \in \mathbb{R}^2$ w.r.t. a reference frame $\mathcal{O}_W$ of the aerial robot flying at a given altitude $h \in \mathbb{R}_{>0}$, the $i$th *stage* $\Gamma_i$ at time instant $t$ is

$$\Gamma_i := \{\varphi_i(\mathbf{p}(t), c_i^\rho), c_i^\sigma \mid \forall j \in [\rho]_{>0},\, c_{i,j} \in \mathcal{C}_{i,j}, \\ \forall k \in [\sigma]_{>0},\, c_{i,\rho+k} \in \mathcal{S}_{i,k}\},$$

where $c_i^\rho$ and $c_i^\sigma$ are $\rho$ *path* and $\sigma$ *computations parameters*. $\mathcal{C}_{i,j} := [\underline{c}_{i,j}, \overline{c}_{i,j}] \subseteq \mathbb{R}$ is the $j$th path parameter $c_{i,j}$ constraint set, and $\mathcal{S}_{i,k} := [\underline{c}_{i,\rho+k}, \overline{c}_{i,\rho+k}] \subseteq \mathbb{Z}_{\geq 0}$ is the $k$th computation parameter constraint set.

For a set $\mathbb{X}$, $\mathbb{X}_{\geq 0}$ then indicates it is positive, $\mathbb{X}_{>0}$ strictly positive, and $|\mathbb{X}|$ its cardinality. $\mathbb{Z}, \mathbb{R}$ are the sets of integers and reals.

The notation $[\,\cdot\,]$ denotes positive naturals up to $\cdot$, i.e., $\{0, 1, \dots, \cdot\}$, $\cdot'$ the transpose of $\cdot$, and $[\underline{\cdot}, \overline{\cdot}]$ the upper/lower bounds of a parameter $\cdot$, i.e.,

$$\underline{\cdot} \leq \cdot \leq \overline{\cdot}. \tag{1}$$

The function $\varphi_i$ is a *path function* specifying the path. These are stage-dependent mathematical functions the aerial robot tracks as it travels the coverage.

**Definition II.2** (Path functions)**.** $\varphi_i : \mathbb{R}^2 \times \mathbb{R}^\rho \to \mathbb{R}, \forall i \in \{1, 2, \dots\}$ are *path functions*, forming the path. They are a function of $\mathbf{p}(t)$ and path parameters $c_i^\rho(t)$ and are continuous and twice differentiable.

The change of stages happens in the proximity of given points termed *triggering points*, whereas the plan is complete at the occurrence of the *final point*.

**Definition II.3** (Triggering and final points)**.** The *triggering point* $\mathbf{p}_{\Gamma_i}$ allows the transition between stages. The *final point* is the last triggering point $\mathbf{p}_{\Gamma_l}$ relative to the last stage $\Gamma_l$.

The plan merges the concepts from Definitions II.1–3.

**Definition II.4** (Plan)**.** The *plan* is a finite state machine (FSM) $\Gamma$, where the state-transition function $s : \bigcup_i \Gamma_i \times \mathbb{R}^2 \rightarrow \bigcup_i \Gamma_i$ maps a stage and a point to the next stage

$$s(\Gamma_i, \mathbf{p}(t)) := \begin{cases} \Gamma_{i+j} & \exists j \in \mathbb{Z},\ \text{if } \|\mathbf{p}(t) - \mathbf{p}_{\Gamma_i}\| < \varepsilon_i \\ \Gamma_i & \text{otherwise} \end{cases}.$$

The stage-dependent value $\varepsilon_i \in \mathbb{R}_{\geq 0}$ in Definition II.4 expresses the radius of an imaginary circle over $\mathbf{p}_{\Gamma_i}$.

Fig. 2 illustrates the concepts in Definitions II.1–4. $\varphi_0, \dots, \varphi_5$ are path functions. $\varphi_0$ and $\varphi_4$ are circles, while $\varphi_1$, $\varphi_3$, and $\varphi_5$ are lines. They are relative to different stages $\Gamma_1, \dots$ but $\Gamma_0$, the starting stage, and are changed in the proximity of $\mathbf{p}_{\Gamma_0}, \dots$. It is possible to alter the paths $\varphi_1, \dots, \varphi_4$ with the parameters $c_{1,1}, \dots, c_{4,1}$; the gray area in the figure.

A convenient way of defining $\Gamma$ is specifying a set of stages, a shift, and a final point. The set is termed *primitive stages* and iterated with the shift up to reaching the final point.

**Definition II.5** (Primitive stages)**.** Given the number of *primitive stages* $n \in \mathbb{Z}_{>0}$, a *shift* $\mathbf{d} \in \mathbb{R}^2$, and a final point $\mathbf{p}_{\Gamma_l}$, the stages $\Gamma_1, \Gamma_2, \dots, \Gamma_n$ are primitive if they form the remainder of the plan with $\mathbf{d}$ up to $\mathbf{p}_{\Gamma_l}$.

In this case, the path functions have a constant distance $e_j$ per each value in $[n]_{>0}$, i.e.,

$$\varphi_{(i-1)n+j}(\mathbf{p} + (i-1)\mathbf{d}, c_1^\rho) - \varphi_{in+j}(\mathbf{p} + i\mathbf{d}, c_1^\rho) = e_j, \quad (2)$$

holds $\forall i \in [l/n - 1]_{>0}, j \in [n]_{>0}$ assuming the total number of stages is known and is $l \in \mathbb{Z}_{>0}$. $e_j \in \mathbb{R}$ given a shift $\mathbf{d}$, initial point $\mathbf{p}$, and initial value of path parameters $c_1^\rho$.

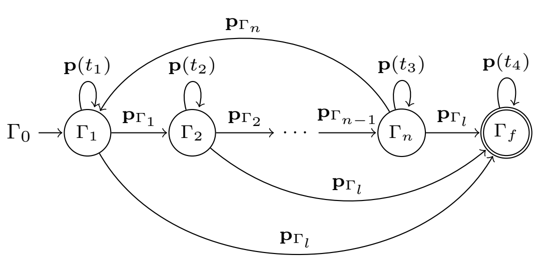



Fig. 3: Plan $\Gamma$ with $n$ primitive stages in Def II.5.

Fig. 3 illustrates the concepts in Definition II.5. A plan composed of $n$ stages $\Gamma_1, \dots, \Gamma_n$ (containing primitive paths $\varphi_1, \dots, \varphi_n$) is reiterated with the shift $\mathbf{d}$. $t_1 < \dots < t_4$ are time instants $\in \mathbb{R}_{>0}$. $\Gamma_f$ is the accepting stage, indicating the plan is complete, $\Gamma_0$ the initial stage where the aerial robot, e.g., awaits the starting command.

### B. Energy-aware planning-scheduling problem

The problem of planning-scheduling is composed of two sub-problems. One is to form a static plan that visits every point in space, the other to re-plan and re-schedule the plan in-flight in an energy-aware way.

**Problem** (Coverage and re-planning-scheduling problem)**.** Consider a finite set of vertices of a polygon $v := \{v_1, v_2, \dots\}$ where each is a point w.r.t. $\mathcal{O}_W$. Let $\underline{r} \in \mathbb{R}_{\geq 0}$, the vehicle's turning radius, and $\mathbf{p}(t_0)$, the starting point at the time instant $t_0$, be given. The *coverage problem* is the problem of finding a plan $\Gamma$ to cover the polygon, whereas the *re-planning-scheduling problem* the optimal parameters $c_i$ in time.

Here, $c_i$ denotes a row vector with both the path and computations parameters in sequence, i.e., $c_i := [\, c_i^\rho \quad c_i^\sigma \,]'$.

## III. Energy Models

The solution to the problem requires energy models, predicting the impact of changes to path and computations parameters on the battery at future time instants. To this end, Sections III-A–C provide models for the overall and computations energies and battery evolution.

### A. Overall energy model

Collected energy data and spectrum analysis in Fig. 1 illustrate the energy of a static coverage plan. Assuming the primitive paths have approximately the same length and the aerial robot has a fixed ground speed, the data exhibit periodic behavior with a constant set of frequencies, independent of the shift. The hypothesis is further backed by the power spectrum analysis, indicating that to model the energy, three frequencies are adequate.

An intuitive way of modeling the energy data is thus a Fourier series of a given order $r \in \mathbb{Z}_{\geq 0}$ and period $T \in \mathbb{R}_{>0}$

$$h(t) = a_0/T + (2/T) \sum_{j=1}^{r} (a_j \cos \omega j t + b_j \sin \omega j t), \quad (3)$$

where $h : \mathbb{R}_{\geq 0} \rightarrow \mathbb{R}$ maps time to the instantaneous energy consumption, $\omega := 2\pi/T$ is the angular frequency, and $a, b \in \mathbb{R}$ the series coefficients.

Equation (3) does not account for the variation of parameters, where, e.g., two schedules result in different instantaneous energies. For this latter purpose, we use the dynamics

$$\dot{\mathbf{q}}(t) = A\mathbf{q}(t) + B\mathbf{u}(t), \quad (4a)$$

$$y(t) = C\mathbf{q}(t), \quad (4b)$$

where $y(t) \in \mathbb{R}$ is the instantaneous energy consumption. The state $\mathbf{q} \in \mathbb{R}^m$ with $m := 2r + 1$ contains energy coefficients

$$\mathbf{q}(t) = \begin{bmatrix} \alpha_0(t) & \alpha_1(t) & \beta_1(t) & \cdots & \alpha_r(t) & \beta_r(t) \end{bmatrix}'. \quad (5)$$

The state transition matrix

$$A = \begin{bmatrix} 0 & 0^{1\times 2} & \dots & 0^{1\times 2} \\ 0^{2\times 1} & A_1 & \dots & 0^{2\times 2} \\ \vdots & \vdots & \ddots & \vdots \\ 0^{2\times 1} & 0^{2\times 2} & \dots & A_r \end{bmatrix}, \quad A_j := \begin{bmatrix} 0 & \omega j \\ -\omega j & 0 \end{bmatrix}, \quad (6)$$

where $A \in \mathbb{R}^{m\times m}$ contains $r$ sub-matrices $A_j$ and $0^{i\times j}$ is a zero matrix of $i$ rows and $j$ columns. In matrix $A$, the top left entry is zero, the diagonal entries are $A_1, \dots, A_r$, the remaining entries are zeros.

The output matrix

$$C = (1/T)\begin{bmatrix} 1 & \overbrace{1 \quad 0 \quad \cdots \quad 1 \quad 0}^{2r} \end{bmatrix}, \quad (7)$$

where $C \in \mathbb{R}^m$ (the first value in the first column is one, the pattern one–zero is then repeated $2r$ times).

To define the nominal control and the output matrix, we exploit the effect of variation of path and computations parameters on the energy. Given $c_i(t)$ parameters at two following time instants $t \in \{t_j, t_{j+1}\} \subset \mathbb{R}_{\geq 0}$ s.t. $t_j < t_{j+1}$ for an arbitrary stage $\Gamma_i$, a change in parameters $c_i(t_j) \neq c_i(t_{j+1})$ results in different overall and instantaneous energies for path and computations parameters respectively.

The nominal control and input matrix in Eq. (4) simply includes the change in energy for all time instants, i.e.,

$$\mathbf{u}(t_{j+1}) := \hat{\mathbf{u}}(t_{j+1}) - \hat{\mathbf{u}}(t_j), \quad B = \begin{bmatrix} 0^{1\times\rho} & 1 & \cdots & 1 \\ 0^{1\times\rho} & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0^{1\times\rho} & 0 & \cdots & 0 \end{bmatrix}, \quad (8)$$

shifts the base frequency $\alpha_0$ assuming the energy of the computations does not alter the other frequencies. $B \in \mathbb{R}^{m\times n}$ with $n := \rho + \sigma$ contains zeros but in the first row where the first $\rho$ columns are zeros and the remaining $\sigma$ are ones. Different combinations of $\mathbf{u}$ with matrix $B$ in Eq. (8) are possible, as we discuss briefly in Section VI. The dynamics in Eqs. (4–8) additionally allows us to use state estimation techniques, such as Kalman filter in Section IV-B, to refine the states $\mathbf{q}$ and model the energy of the aerial robot flying under diverse conditions.

The models in Eqs. (3–4) are equal when $\mathbf{u}$ is a zero vector and an initial guess $\mathbf{q}(t_0) = \mathbf{q}_0$ at initial time instant $t_0$

$$\mathbf{q}_0 = \begin{bmatrix} a_0 & a_1/2 & b_1/2 & \cdots & a_r/2 & b_r/2 \end{bmatrix}',$$

i.e., $h, y$ are both harmonic signals with the same frequencies. $\hat{\mathbf{u}}$ in Eq. (8) is then a scale transformation

$$\hat{\mathbf{u}}(t) := \operatorname{diag}(\nu_i) c_i(t) + \tau_i, \quad (9)$$

where $\operatorname{diag}(\,\cdot\,)$ is a diagonal matrix with items of a set $\cdot$ on the diagonal and zeros elsewhere. $\nu_i := \begin{bmatrix} \nu_{i,1} & \cdots & \nu_{i,n} \end{bmatrix}'$ and $\tau_i := \begin{bmatrix} \tau_{i,1} & \cdots & \tau_{i,n} \end{bmatrix}'$ are scaling factors that transform parameters domain (see Definition II.1) to time and power domains.

Let assume for ease of notation that the coverage time evolves linearly. Path parameters $c_i^\rho$ can be transformed into a time measure with scaling factors

$$\nu_{i,j} = \left( (\bar{t} - \underline{t}) / (\bar{c}_{i,j} - \underline{c}_{i,j}) \right) / \rho, \quad (10a)$$
$$\tau_{i,j} = \left( \underline{c}_{i,j} (\underline{t} - \bar{t}) / (\bar{c}_{i,j} - \underline{c}_{i,j}) + \underline{t} \right) / \rho, \quad (10b)$$

$\forall j \in [\rho]_{>0}$ where $\bar{t}, \underline{t}$ are time measures needed to complete the coverage with configurations $\underline{c}_i^\rho, \bar{c}_i^\rho$ $(\underline{\Gamma}, \overline{\Gamma})$

Similarly to Eq. (10), computations parameters $c_i^\sigma$ can be transformed into an instantaneous energy measure with

$$\nu_{i,j} = (g(\bar{c}_{i,j}) - g(\underline{c}_{i,j})) / (\bar{c}_{i,j} - \underline{c}_{i,j}), \quad (11a)$$
$$\tau_{i,j} = \underline{c}_{i,j} (g(\underline{c}_{i,j}) - g(\bar{c}_{i,j})) / (\bar{c}_{i,j} - \underline{c}_{i,j}) + g(\underline{c}_{i,j}), \quad (11b)$$

$\forall j \in [\rho + 1, n]$. The function $g$ is detailed in Section III-B and quantifies the power of the computing hardware.

### B. Energy model for the computations

Models for heterogeneous computing hardware in the literature often rely on analytical expressions [41]–[44] or different techniques, including regressional analysis [36], [45], [46], aiding the selection of hardware- or software-specific parameters. This section summarizes an energy model from our early studies [36], [37] that relies on regressional analysis to quantify the computations energy of any configuration of computations $c_i^\sigma$ within the bounds (see Definition II.1).

The model compromises an automatic modeling and profiling tool [36] named `powprofiler` distributed [47] under the open-source MIT license. It is segmented into two layers. In the *measurement layer*, the tool measures a discrete set of computations parameters and infers the energy of the remaining in the *predictive layer* via a piecewise linear regression.

Let assume there is at least one measuring device, i.e., shunt or internal power resistor, multimeter, or amperemeter, quantifying the power drain of a specific component, e.g., CPU, GPU, memory, etc., or of the entire computing hardware.

**Definition III.1** (Measurement layer)**.** Given a measuring device, computations parameters, and initial and final time instants, the *measurement layer* is the function $\mathbf{g} : \mathbb{Z}_{>0} \times \mathbb{Z}^\sigma \times \mathcal{T} \to \mathbb{R}$ that returns an energy measure.

Here, the notation $\mathcal{T}$ encloses all the time intervals from initial $t_0$ to final $t_f$, i.e., $\mathcal{T} := [t_0, t_f]$.

**Definition III.2** (Predictive layer)**.** Given a measuring device and computations parameters, the *predictive layer* is the function $g : \mathbb{Z}_{>0} \times \mathbb{Z}^\sigma \to \mathbb{R}$ that returns an energy measure.

The energy measures in Definitions III.1–2 can be either average or overall. Additionally, the `powprofiler` tool supports the battery state of charge (SoC) detailed in Section III-C. The function $g$ in Definition III.2 is contained in the computations scaling factors in Eq. (11), assuming the computations energy behaves linearly between $\underline{c}_i^\sigma$ and $\bar{c}_i^\sigma$, otherwise

$$\begin{aligned} g(c_i^\sigma) = {} & (\mathbf{g}(\lceil c_i^\sigma \rceil, \mathcal{T}_1) - \mathbf{g}(\lfloor c_i^\sigma \rfloor, \mathcal{T}_2)) \\ & (c_i^\sigma - \lfloor c_i^\sigma \rfloor) / (\lceil c_i^\sigma \rceil - \lfloor c_i^\sigma \rfloor) + \mathbf{g}(\lfloor c_i^\sigma \rfloor, \mathcal{T}_2), \end{aligned} \quad (12)$$

where notation $\lceil c_i^\sigma \rceil, \lfloor c_i^\sigma \rfloor$ indicates two adjacent measurement layers, and $\mathcal{T}_1, \mathcal{T}_2$ are the corresponding two time intervals. The measuring device in both $\mathbf{g}$ and $g$ is implicit.

### C. Battery model

The battery model predicts the battery SoC in the function of a given load at future time instants. There are multiple models in the literature [48] with varying complexity, accuracy, and ease of implementation ranging from accurate but costly physical models [49], [50], to abstract models [38]–[40], [51] that have compelling trade-offs in terms of the latter two. We model a Li-ion battery of an aerial robot in-flight with an abstract "Rint" ECM in the literature [38]–[40].

The battery SoC changes according to [52], [53], i.e.,

$$\dot{b}(y(t)) = -k_b I(y(t)) / Q_c, \quad (13)$$

where $I(y(t)) \in \mathbb{R}$ is the internal current measured in amperes, $y(t) \in \mathbb{R}_{\geq 0}$ the power drain, and $Q_c \in \mathbb{R}$ the battery

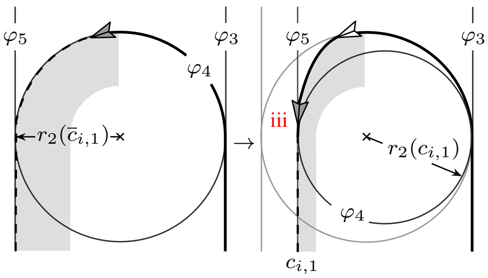



Fig. 4: Alteration of the path parameter $c_{i,1}$, the radius of the circle (i.e., the alteration of the plan in Fig. 1).

constant nominal capacity measured in amperes per hour. $k_b$ is a battery coefficient added to [52], [53] and derived experimentally. The "Rint" circuit models the battery as a perfect voltage source connected with a resistor $R_r \in \mathbb{R}$ measured in ohm, representing the battery resistance. The voltage on the extremes of ECM respects $V_e = V - R_r I$, where $V, V_e \in \mathbb{R}$ are the internal and external battery voltages measured in volts. The former can be retrieved from the battery data sheet [39] and depends on SoC [52].

If the voltage of the power drain is stable, Kirchhoff's circuit laws lead to $V_s I_l = V_e I$, where $I_l$ is the current required by the load measured in amperes. Combining $V_e, V_s I_l$ results in the quadratic expression $R_r I^2 - VI + V_s I_l = 0$. Solving the expression utilizing the negative solution (when $I_l$ is zero, $I$ should also be zero) results in

$$I(y(t)) = (V - \sqrt{V^2 - 4R_r y(t)})/(2R_r). \tag{14}$$

Eq. (4) states that the output $y$ evolves in $\mathbb{R}$, yet, aerial robots usually use a battery. We thus use instead

$$\mathcal{Y}(t) := \{y \mid y \in [0, b\,Q_c V] \subseteq \mathbb{R}_{\geq 0}\}, \tag{15}$$

where $b\,Q_c V$, the maximum instantaneous energy consumption measured in watts, is derived from Eq. (13).

## IV. Planning-Scheduling

The section solves the problem in Section II-B. It provides a plan $\Gamma$ and re-plans-schedules such plan energy-wise.

### A. Coverage

There are various approaches in the literature to solve CPP problems (e.g., Section II-B). These that ensure the completeness are NP-hard [54] and use cellular decomposition, dividing the free-space into sub-regions to be easily covered [1], [2].

An intuitive way to solve the problem is with a back-and-forth motion, sweeping the space delimited by $v$ we term $\mathcal{Q}^v$. Although abundant in both mobile ground-based [1], [55], [56] and aerial [57]–[60] robotics literature, the motion, called *boustrophedon motion* [1], is unsuitable for aerial robots broadly, especially for fixed-wing aerial robots. These robots have reduced maneuverability [61]–[64] and are generally unable to fly quick turns [65].

To address fixed wings and aerial robots generally, this section details a different motion with a wide turning radius. It is similar to another motion in the literature, the *Zamboni motion* [57], but additionally allows variable CPP at the very core of this work. The novel motion is termed *Zamboni-like motion* and is composed of four primitive paths (see Definition II.5): two lines $\varphi_1, \varphi_2$ and two circles $\varphi_3, \varphi_4$.

**Algorithm 1** Zamboni-like motion for CPP

1: **for all** $t \in \mathcal{T}$ **do**
2:   **if** $\mathbf{p}(t) = \mathbf{p}_{\Gamma_l}$ in Definition II.3 **then return** $\Gamma$
3:   **if** $\mathbf{p}(t) = \mathbf{p}_{\Gamma_i}$ **then**
4:     $i \leftarrow i + 1$
5:     **if** $i \notin [n]_{>0}$ **then**
6:       $i \leftarrow 1$
7:       $\varphi_{|\Gamma|+1} \leftarrow$ line in Definition II.2 parallel to ${}^{v_1}|_{v_{|v|}}$ that intersects $\mathbf{p}_{|\Gamma|}$
8:       $\mathbf{p}_{|\Gamma|+1} \leftarrow$ other intersection of $\varphi_{|\Gamma|+1}$ and $v$
9:       $\varphi_{|\Gamma|+2} \leftarrow$ circle whose left most point lays on $\mathbf{p}_{|\Gamma|+1}$
10:       $\mathbf{p}_{|\Gamma|+2} \leftarrow$ other inter. of $\varphi_{|\Gamma|+2}$ and $v$
11:       $\varphi_{|\Gamma|+3} \leftarrow$ line par. to $\varphi_{|\Gamma|+1}$ that inter. $\mathbf{p}_{|\Gamma|+2}$
12:       $\mathbf{p}_{|\Gamma|+3} \leftarrow$ other inter. of $\varphi_{|\Gamma|+3}$ and $v$
13:       $\varphi_{|\Gamma|+4} \leftarrow$ circle in Eq. (17) whose right most point lays on $\mathbf{p}_{|\Gamma|+3}$
14:       $\mathbf{p}_{|\Gamma|+4} \leftarrow$ other inter. of $\varphi_{|\Gamma|+4}$ and $v$
15:       $\Gamma \leftarrow \Gamma \cup \{\Gamma_{|\Gamma|+1}, \ldots, \Gamma_{|\Gamma|+4}\}$ in Definitions II.4–2

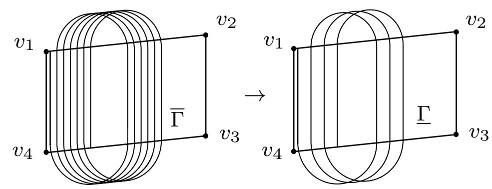



Fig. 5: Zamboni-like motion: ($\bar{\Gamma}$) with four primitive paths (Lines 9–14 in Algorithm 1) can be re-planned ($\Gamma$) with $r_2$ in Eq. (16).

Let assume the vertices $v_1, v_2, \ldots$ are ordered from the top-left-most vertex in clockwise order, the aerial robot can overfly the edges formed by the vertices, and ${}^{v_x}|_{v_y}$ indicates the edge formed by vertices $v_x, v_y$. Algorithm 1 details the procedure to generate the plan $\Gamma$ that covers $\mathcal{Q}^v$ per each discretized time step, i.e., $\mathcal{T} := \{t_0, t_0 + h, \ldots, t_f\}$ for a given step $h \in \mathbb{R}_{>0}$. The algorithm assumes that the line parallel to ${}^{v_1}|_{v_{|v|}}$ is always connected as it swipes $\mathcal{Q}^v$. Nonetheless, complex covering is possible by, e.g., dividing $\mathcal{Q}^v$ into cells to be easily covered and subsequently covering each cell [1].

To implement the variable CPP, the radius $r_2$ of the second circle $\varphi_{|\Gamma|+4}$ on Line 13

$$r_2(c_{i,1}) := \sqrt{r^2 + c_{i,1}}, \tag{16}$$

and is expressed in the function of a path parameter $c_{i,1} \in (\underline{r}^2 - r^2, 0]$, relative to the last circle in each set of primitive stages. $r \in \mathbb{R}_{>0}$ is a given ideal turning radius along with the minimum radius (see Section II-B). The center also changes

$$\varphi_{|\Gamma|+4} := (x - x_{\mathbf{p}_{|\Gamma|+3}} + r_2)^2 + (y - y_{\mathbf{p}_{|\Gamma|+3}})^2 - r_2^2, \tag{17}$$

where $(x_{\mathbf{p}}, y_{\mathbf{p}}) =: \mathbf{p}$ for any point $\mathbf{p}$. Fig. 4 illustrates the concept of $c_{i,1}$ altering the CPP. The radius of the first circle on Line 9 is then $r_1 := r + x_{\mathbf{d}}/2$ (i.e., the radiuses of the two circles ensure that the primitive paths are shifted of $\mathbf{d}$).

Algorithm 1 initializes $i$ to minus one and builds the first four primitive functions $\varphi_1, \ldots, \varphi_4$. The remaining $\Gamma$ is built with the shift $\mathbf{d}$ up to the final point $\mathbf{p}_{\Gamma_l}$. The initial point is $\mathbf{p}_{\Gamma_1}$, placed s.t. the line $\varphi_1$ is at the same distance from an eventual previous line, e.g., $x_{\mathbf{p}_{\Gamma_1}} = x_{v_1} + x_{\mathbf{d}}/2$ in Fig. 5.

### B. Re-planning-scheduling

Past literature on planning-scheduling often relies on optimal control and optimization related approaches [18], [20],

**Algorithm 2** Coverage re-planning-scheduling

1: **for all** $t \in \mathcal{T}$ **do**
16: $\quad \mathbf{q}(\mathcal{K} \setminus \{t+N\}), c_i(\mathcal{K}) \leftarrow$ solve NLP $\arg\max_{\mathbf{q}(k),c_i(k)} l_f(\mathbf{q}(t+N), t+N) + \sum_{k\in\mathcal{K}} l_d(\mathbf{q}(k), c_i(k), k)$ in Eq. (18) on $\mathcal{K} = \{t, t+h, \dots, t+N\}$
17: $\quad k \leftarrow t$
18: $\quad$ **while** $b_d(y(k)) > 0$ **do**
19: $\quad\quad$ **if** $k + h \notin \mathcal{K}$ **then**
20: $\quad\quad\quad \mathbf{q}(k+h) \leftarrow$ solve model in Eq. (4a)
21: $\quad\quad b_d(y(k+h)) \leftarrow$ solve model in Eq. (13)
22: $\quad\quad k \leftarrow k + h$
23: $\quad t_b \leftarrow k - t$
24: $\quad t_s \leftarrow (\mathrm{diag}(\nu_i^\rho) c_i^\rho(t) + \tau_i^\rho)\overbrace{[1 \quad 1 \quad \cdots \quad 1]}^{\rho}$
25: $\quad t_r \leftarrow (t_s/\bar{t})(\bar{t} - t)$
26: $\quad$ **if** $t_r < t_b$ **then**
27: $\quad\quad c_i^\rho(t) \leftarrow$ find $c_i^\rho$ with $t_c \in [0, t_b]$, otherwise take $\underline{c}_i^\rho$
28: $\quad \hat{\mathbf{q}}(t+h) \leftarrow$ estimate $\mathbf{q}$ in Eq. (4a) with energy sensor $\Upsilon(t)$
29: $\quad \hat{y}(t+h) \leftarrow$ derive $y$ from Eq. (4b) with est. state $\hat{\mathbf{q}}(t+h)$

[21], [32]. We similarly derive an optimal control problem returning the trajectory of parameters $c_i(\mathcal{T})$ with $\mathcal{T} := [t_0, t_f]$ (see Definition III.1). Since the final time instant and the exact value of the state $\mathbf{q}$ are not known, we use output model predictive control (MPC) that derives the configuration for a finite horizon on an estimated state $\hat{\mathbf{q}}$, i.e., $t_f := t_0 + N$ for a given $N \in \mathbb{R}_{>0}$ [66].

An optimal control problem (OCP) that selects the highest configuration of $c_i$ and respects the constraints, with $\mathbf{q}(t)$ and $c_i(t)$ the state and parameters trajectories

$$\max_{\mathbf{q}(t),c_i(t)} l_f(\mathbf{q}(t_f), t_f) + \int_{t_0}^{t_f} l(\mathbf{q}(t), c_i(t), t)\, dt, \tag{18a}$$
$$\text{s.t. } \dot{\mathbf{q}} = f(\mathbf{q}(t), c_i(t), t), \tag{18b}$$
$$c_{i,j}(t) \in \mathcal{C}_{i,j},\; c_{i,\rho+k}(t) \in \mathcal{S}_{i,k}\; \forall j \in [\rho]_{>0},\; k \in [\sigma]_{>0}, \tag{18c}$$
$$\mathbf{q}(t) \in \mathbb{R}^m,\; y(t) \in \mathcal{Y}(t), \tag{18d}$$
$$\mathbf{q}(t_0) = \hat{\mathbf{q}}_0 \;\text{ given (last estimated state), and} \tag{18e}$$
$$b(t_0) = b_0 \;\text{ given,} \tag{18f}$$

where $l : \mathbb{R}^m \times \mathcal{C}_i \times \mathcal{S}_i \times \mathbb{R}_{\geq 0} \to \mathbb{R}$ is a given initial initial cost function with the quadratic expression

$$l(\mathbf{q}(t), c_i(t), t) = \mathbf{q}'(t) Q \mathbf{q}(t) + c_i'(t) R c_i(t), \tag{19}$$

where $Q \in \mathbb{R}^{m\times m}, R \in \mathbb{R}^{n\times n}$ are given positive semidefinite matrices.The final cost function $l_f : \mathbb{R}^m \times \mathbb{R}_{>0} \to \mathbb{R}$ is also a quadratic expression but with no control [66]

$$l_f(\mathbf{q}(T), T) = \mathbf{q}'(T) Q_f \mathbf{q}(T), \tag{20}$$

where $Q_f \in \mathbb{R}^{m\times m}$ is a given positive semidefinite matrix.

Furthermore, Eq. (18b) is the differential periodic energy model in Eq. (4). The model requires a value of the period $T$, which is simply the time needed to fly the four primitive paths in the Zamboni-like motion, i.e., time elapsed between two positive evaluations of the condition on Line 5.

Eq. (18c) are the parameters constraints sets in Definition II.1. Eq. (18d) are the state and output constraints in Eq. (15) that evolves the battery model in Eq. (13). Eq. (18e) is the state guess estimated via state estimation (the very first estimate is given). Eq. (18f) is the initial battery SoC from, e.g., flight controller.

Line 16 in Algorithm 2 contains a transcribed version of the OCP in Eq. (18) into a nonlinear program (NLP) that can be easily solved with available NLP solvers [66]. Its solution leads to both trajectories of parameters and states for future $N$ instants. Here, the sets $\mathcal{K}, \mathcal{T}$ have possibly different steps $h$ (not to be confused with the altitude), tuning the precision. The functions $l_d, b_d$ are the discretized versions of Eq. (19) and Eq. (13), with, e.g., Runge-Kutta or Euler methods [67].

Lines 17–23 estimate the time needed to completely drain the battery, exploiting the SoC already predicted previously on Line 16. The coverage is then replanned accordingly on Lines 24–27 using scaling factors from Eq. (10). Lines 28–29 estimate the energy model's state with current energy sensor reading $\Upsilon$, with, e.g., Kalman filter [68].

Algorithm 2 implements Eq. (18) for the purpose of energy-aware re-planning-scheduling of $\Gamma$ from Algorithm 1, i.e, Lines 16–29 continue after Line 15 in Algorithm 1.

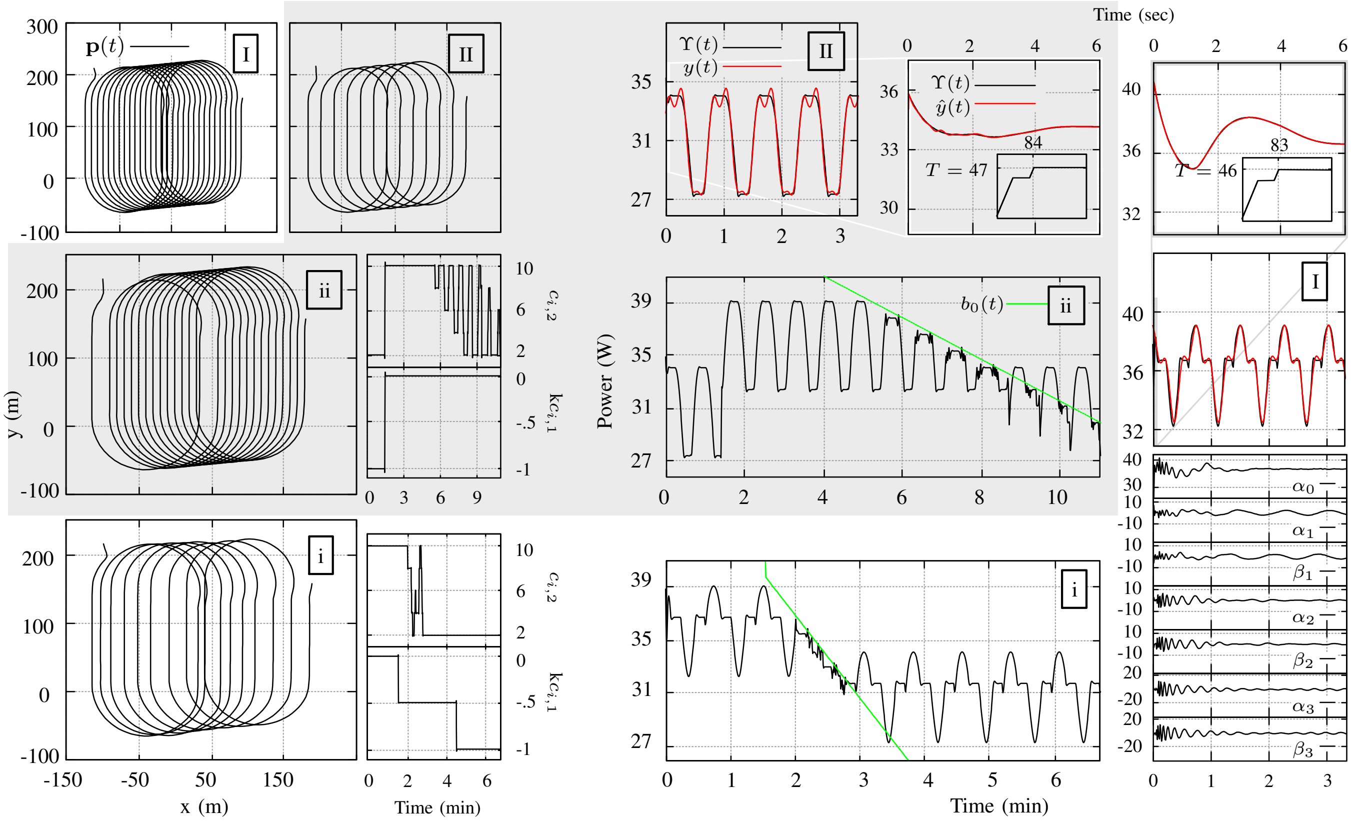
## V. Numerical Simulations

Numerical simulations of Algorithms 1–2 in this section are implemented in MATLAB (R), complement physical flights of a static coverage plan with the open-source Paparazzi flight controller [69], and are extended with the computations energy model on NVIDIA (R) Jetson Nano (TM) heterogeneous computing hardware. The computing hardware caries a camera as a peripheral and is evaluated independently of the aerial robot with the `powprofiler` (see Section III-B). The scheduler, implemented through Robot Operating System (ROS) middleware [70], varies a computation parameter $c_{i,2}$ relative to the ground patterns detection rate from two to ten frames-per-second (FPS). The detection uses PedNet, a Convolutional Neural Network (CNN) [71], also implemented through ROS. The planner varies the path parameter $c_{i,1}$ in Eq. (16) between zero and -1000 (i.e., the planner-scheduler is the concrete implementation of Algorithms 1–2). The set of parameters is unaltered through the flight, i.e, $c_i := \begin{bmatrix} c_{i,1} & c_{i,2} \end{bmatrix}', \forall i$.

Fig. 1 details the data of the physical flight in standard atmospheric conditions. Fig. 6 extends the flight with the computing hardware aided by a flight simulation implemented in MATLAB (R). Upper-case roman numerals I,II indicate the plans are static (i.e., solely Algorithm 1), lower-case i,ii exploit planning-scheduling in the letter.

Fig. 6a illustrates the same plan $\Gamma$ under different conditions. Flights I–i have a constant wind speed of five meters per second, a wind direction of zero degrees, and initial parameters $c_{i,1}, c_{i,2}$ values of zero and ten (i.e., full $r_2$ and detection). Flights II–ii (see added gray background for clarity) the same but a wind direction of 90 degrees and the initial parameters values of -1000 and two (i.e., minimum $r_2$ and detection).

Fig. 6b illustrates first the power ($\Upsilon$ on Line 28 in Algorithm 2), and then the energy model ($y$ on Line 20). Flight i simulates a battery (green line, the battery behavior $b_0$) drop at approximately one minute and a half and four minutes and a half. Planner-scheduler optimizes the path in the proximity



(a) Path, parameters $(c_{i,1}, c_{i,2})$ evolutions

(b) Energies, model states $(\alpha_0, \dots \in \mathbf{q})$, and parameters $(T, b_0)$ evolutions

Fig. 6: CPP with novel Zamboni-like motion (I,II) and Planning-scheduling of CPP and ground patterns detections with PedNet CNN (i,ii) in terms of the path, energies, and plans-schedules under different conditions (I–i,II–ii): wind speed and direction, battery behavior, and parameters initial values.

of the drops to ensure that the flight is completed, whereas it maximizes the parameter $c_{i,2}$ when the battery is discharging, respecting the output constraint (Eq. (15)). Flight ii simulates the opposite scenario. The lowest configuration of parameters and no battery defects. The path parameter increases as soon as the algorithm estimated enough data (two periods $T$) and the computation parameter decreases mapping the battery discharge rate. For both cases, scaling factors are derived empirically, the horizon $N$ is set to six seconds similarly to relevant literature [72]–[75], order $r$ is three (see Fig. 1), and the matrices $Q, R, Q_f$ are chosen such that the cost is merely squared control. The figure further details the energy model's estimate (see detail view for I–II) on an initial slice of the model ($\hat{y}$), power ($\Upsilon$), and period ($T$). The bottom detail of I illustrates the evolutions of the state $\mathbf{q}$ in time, concluding that approximately two periods are sufficient to obtain a consistent state estimate.

Output MPC on Line 16 relies on a software framework for nonlinear optimization called CasADi [76]–[78], and the popular NLP solver IPOPT [79].

## VI. Conclusions and Future Directions

The letter provides a planning-scheduling approach for autonomous aerial robots powered by a limited power source, extending past literature. It proposes a novel coverage motion for variable CPP robust to aerial robots constraints such as the turning radius of fixed wings. Energy modeling in the letter exploits collected empirical data of the fixed-wing aerial robot flying static CPP and further incorporates the energy of the computing hardware via the `powprofiler` tool. The approach compromises two algorithms: one derives a static coverage plan, whereas the other re-plans-schedules the plan on a finite horizon via MPC. It evolves the state of the energy model while optimizing battery usage and remedying possible defects. The plan compromise multiple stages, where at each the aerial robot flies a path and runs the computations, allowing further extensibility in terms of constructs and approaches.

Indeed, we are currently extending the results to a standard flight controller. The guidance on the coverage, coverage with variable altitude, and distributed planning-scheduling merits alike further investigation, as well as the study of the implications of planning-scheduling on other energy-critical mobile robots. Here, our preliminary study led to possible savings [80], in line with relevant literature [20], [21].

Further future directions include the study of different energy models. Among the others, these include, e.g., aperiodic energy models, different linear combinations of the variations of parameters, and stage dependent energy models. In a unconventional setting with, e.g., multiple agent utilized for overall coverage, an agent-dependent model might be employed to achieve energy awareness.

## References

[1] H. Choset, "Coverage for robotics–a survey of recent results," *Annals of Mathematics and Artificial Intelligence*, vol. 31, pp. 113–126, 2001.

1, 5

[2] E. Galceran and M. Carreras, "A survey on coverage path planning for robotics," *Robotics and Autonomous Systems*, vol. 61, no. 12, pp. 1258–1276, 2013. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S092188901300167X 1, 5

[3] T. a. M. Cabreira, L. B. Brisolara, and P. R. Ferreira Jr., "Survey on coverage path planning with unmanned aerial vehicles," *Drones*, vol. 3, no. 1, 2019. [Online]. Available: https://www.mdpi.com/2504-446X/3/1/4 1

[4] S. S. H. Hajjaj and K. S. M. Sahari, "Review of research in the area of agriculture mobile robots," in *8th International Conference on Robotic, Vision, Signal Processing & Power Applications*. Springer, 2014, pp. 107–117. 1

[5] F. Qingchun, Z. Wengang, Q. Quan, J. Kai, and G. Rui, "Study on strawberry robotic harvesting system," in *International Conference on Computer Science and Automation Engineering (CSAE)*, vol. 1. IEEE, 2012, pp. 320–324. 1

[6] F. Dong, W. Heinemann, and R. Kasper, "Development of a row guidance system for an autonomous robot for white asparagus harvesting," *Computers and Electronics in Agriculture*, vol. 79, no. 2, pp. 216–225, 2011. 1

[7] Z. De-An, L. Jidong, J. Wei, Z. Ying, and C. Yu, "Design and control of an apple harvesting robot," *Biosystems Engineering*, vol. 110, no. 2, pp. 112–122, 2011. 1

[8] A. Aljanobi, S. Al-Hamed, and S. Al-Suhaibani, "A setup of mobile robotic unit for fruit harvesting," in *19th International Workshop on Robotics in Alpe-Adria-Danube Region (RAAD)*. IEEE, 2010, pp. 105–108. 1

[9] Z. Li, J. Liu, P. Li, and W. Li, "Analysis of workspace and kinematics for a tomato harvesting robot," in *2008 International Conference on Intelligent Computation Technology and Automation (ICICTA)*, vol. 1. IEEE, 2008, pp. 823–827. 1

[10] Y. Edan, D. Rogozin, T. Flash, and G. E. Miles, "Robotic melon harvesting," *IEEE Transactions on Robotics and Automation*, vol. 16, no. 6, pp. 831–835, 2000. 1

[11] V. Puri, A. Nayyar, and L. Raja, "Agriculture drones: A modern breakthrough in precision agriculture," *Journal of Statistics and Management Systems*, vol. 20, no. 4, pp. 507–518, 2017. 1

[12] P. Daponte, L. De Vito, L. Glielmo, L. Iannelli, D. Liuzza, F. Picariello, and G. Silano, "A review on the use of drones for precision agriculture," in *Conference Series: Earth and Environmental Science*, vol. 275, no. 1. IOP Publishing, 2019, p. 012022. 1

[13] Y. Mei, Y.-H. Lu, Y. C. Hu, and C. G. Lee, "A case study of mobile robot's energy consumption and conservation techniques," in *12th International Conference on Advanced Robotics (ICAR)*. IEEE, 2005, pp. 492–497. 1, 2

[14] W. Andrew, C. Greatwood, and T. Burghardt, "Aerial animal biometrics: Individual friesian cattle recovery and visual identification via an autonomous UAV with onboard deep inference," in *International Conference on Intelligent Robots and Systems (IROS)*, 2019, pp. 237–243. 1

[15] T. Peng, D. Zhang, R. Liu, V. K. Asari, and J. S. Loomis, "Evaluating the power efficiency of visual SLAM on embedded GPU systems," in *National Aerospace and Electronics Conference (NAECON)*. IEEE, 2019, pp. 117–121. 1

[16] L. Wang, X. Ye, H. Xing, Z. Wang, and P. Li, "YOLO nano underwater: A fast and compact object detector for embedded device," in *Global Oceans Conference*. IEEE, 2020, pp. 1–4. 1

[17] G. Alexey, V. Klyachin, K. Eldar, and A. Driaba, "Autonomous mobile robot with AI based on Jetson Nano," in *Future Technologies Conference (FTC)*. Springer, 2021, pp. 190–204. 1

[18] J. Brateman, C. Xian, and Y.-h. Lu, "Energy-effcient scheduling for autonomous mobile robots," in *IFIP International Conference on Very Large Scale Integration*. IEEE, 2006, pp. 361–366. 1, 2, 5

[19] S. Sudhakar, S. Karaman, and V. Sze, "Balancing actuation and computing energy in motion planning," in *International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 4259–4265. 1, 2

[20] M. Lahijanian, M. Svorenova, A. A. Morye, B. Yeomans, D. Rao, I. Posner, P. Newman, H. Kress-Gazit, and M. Kwiatkowska, "Resource-performance tradeoff analysis for mobile robots," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 1840–1847, 2018. 1, 2, 5, 7

[21] P. Ondrúška, C. Gurău, L. Marchegiani, C. H. Tong, and I. Posner, "Scheduled perception for energy-efficient path following," in *International Conference on Robotics and Automation (ICRA)*. IEEE, 2015, pp. 4799–4806. 1, 2, 5, 7

[22] A. Sadrpour, J. Jin, and A. G. Ulsoy, "Mission energy prediction for unmanned ground vehicles using real-time measurements and prior knowledge," *Journal of Field Robotics*, vol. 30, no. 3, pp. 399–414, 2013. 1, 2

[23] D.-K. Ho, K. Ben Chehida, B. Miramond, and M. Auguin, "Learning-based adaptive management of QoS and energy for mobile robotic missions," *International Journal of Semantic Computing*, vol. 13, no. 04, pp. 513–539, 2019. 1, 2

[24] Y. Mei, Y.-H. Lu, Y. C. Hu, and C. G. Lee, "Energy-efficient motion planning for mobile robots," in *International Conference on Robotics and Automation (ICRA)*, vol. 5. IEEE, 2004, pp. 4344–4349. 1

[25] M. Wahab, F. Rios-Gutierrez, and A. El Shahat, "Energy modeling of differential drive robots," in *Southeast Conference (SoutheastCon)*. IEEE, 2015. 1

[26] C. H. Kim and B. K. Kim, "Energy-saving 3-step velocity control algorithm for battery-powered wheeled mobile robots," in *International Conference on Robotics and Automation (ICRA)*. IEEE, 2005, pp. 2375–2380. 1

[27] H. Kim and B.-K. Kim, "Minimum-energy translational trajectory planning for battery-powered three-wheeled omni-directional mobile robots," in *10th International Conference on Control, Automation, Robotics and Vision*. IEEE, 2008, pp. 1730–1735. 1

[28] S. T. H. Rizvi, G. Cabodi, D. Patti, and M. M. Gulzar, "A general-purpose graphics processing unit (GPGPU)-accelerated robotic controller using a low power mobile platform," *Journal of Low Power Electronics and Applications*, vol. 7, no. 2, p. 10, 2017. 1

[29] A. Abramov, K. Pauwels, J. Papon, F. Worgotter, and B. Dellen, "Real-time segmentation of stereo videos on a portable system with a mobile GPU," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 22, no. 9, pp. 1292–1305, 2012. 1

[30] M. T. Satria, S. Gurumani, W. Zheng, K. P. Tee, A. Koh, P. Yu, K. Rupnow, and D. Chen, "Real-time system-level implementation of a telepresence robot using an embedded GPU platform," in *Design, Automation & Test in Europe Conference & Exhibition (DATE)*. IEEE, 2016, pp. 1445–1448. 1

[31] U. Jaramillo-Avila, J. M. Aitken, and S. R. Anderson, "Visual saliency with foveated images for fast object detection and recognition in mobile robots using low-power embedded GPUs," in *19th International Conference on Advanced Robotics (ICAR)*. IEEE, 2019, pp. 773–778. 1

[32] W. Zhang and J. Hu, "Low power management for autonomous mobile robots using optimal control," in *46th Conference on Decision and Control (CDC)*. IEEE, 2007, pp. 5364–5369. 2, 5

[33] D.-K. Ho, K. Ben Chehida, B. Miramond, and M. Auguin, "QoS and energy-aware run-time adaptation for mobile robotic missions: A learning approach," in *3rd International Conference on Robotic Computing (IRC)*. IEEE, 2019, pp. 212–219. 2

[34] ——, "Towards a multi-mission QoS and energy manager for autonomous mobile robots," in *2nd International Conference on Robotic Computing (IRC)*. IEEE, 2018, pp. 270–273. 2

[35] A. Seewald, H. García de Marina, H. S. Midtiby, and U. P. Schultz, "Mechanical and computational energy estimation of a fixed-wing drone," in *4th International Conference on Robotic Computing (IRC)*. IEEE, 2020, pp. 135–142, https://adamseewald.cc/short/mechanical2020. [Online]. Available: https://adamseewald.cc/short/mechanical2020 2

[36] A. Seewald, U. P. Schultz, E. Ebeid, and H. S. Midtiby, "Coarse-grained computation-oriented energy modeling for heterogeneous parallel embedded systems," *International Journal of Parallel Programming*, vol. 49, no. 2, pp. 136–157, 2021, https://adamseewald.cc/short/coarse2019. [Online]. Available: https://adamseewald.cc/short/coarse2019 2, 4

[37] A. Seewald, U. P. Schultz, J. Roeder, B. Rouxel, and C. Grelck, "Component-based computation-energy modeling for embedded systems," in *SIGPLAN International Conference on Systems, Programming, Languages, and Applications: Software for Humanity (SPLASH)*. ACM, 2019, pp. 5–6, https://adamseewald.cc/short/component2019. [Online]. Available: https://adamseewald.cc/short/component2019 2, 4

[38] H. He, R. Xiong, and J. Fan, "Evaluation of lithium-ion battery equivalent circuit models for state of charge estimation by an experimental approach," *Energies*, vol. 4, no. 4, pp. 582–598, 2011. [Online]. Available: https://www.mdpi.com/1996-1073/4/4/582 2, 4

[39] H. Hinz, "Comparison of lithium-ion battery models for simulating storage systems in distributed power generation," *Inventions*, vol. 4, 2019. [Online]. Available: https://www.mdpi.com/2411-5134/4/3/41 2, 4, 5

[40] S. Mousavi G. and M. Nikdel, "Various battery models for various simulation studies and applications," *Renewable and Sustainable*

*Energy Reviews*, vol. 32, pp. 477–485, 2014. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1364032114000598 2, 4

[41] A. Marowka, "Energy-aware modeling of scaled heterogeneous systems," *International Journal of Parallel Programming*, vol. 45, no. 5, pp. 1026–1045, 2017. 4

[42] M. Goraczko, J. Liu, D. Lymberopoulos, S. Matic, B. Priyantha, and F. Zhao, "Energy-optimal software partitioning in heterogeneous multiprocessor embedded systems," in *45th ACM/IEEE Design Automation Conference*. IEEE, 2008, pp. 191–196. 4

[43] E. Calore, S. F. Schifano, and R. Tripiccione, "Energy-performance tradeoffs for HPC applications on low power processors," in *European Conference on Parallel Processing*. Springer, 2015, pp. 737–748. 4

[44] T.-J. Yang, Y.-H. Chen, and V. Sze, "Designing energy-efficient convolutional neural networks using energy-aware pruning," in *Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2017, pp. 5687–5695. 4

[45] P. E. Bailey, D. K. Lowenthal, V. Ravi, B. Rountree, M. Schulz, and B. R. De Supinski, "Adaptive configuration selection for power-constrained heterogeneous systems," in *43rd International Conference on Parallel Processing*. IEEE, 2014, pp. 371–380. 4

[46] K. Ma, X. Li, W. Chen, C. Zhang, and X. Wang, "GreenGPU: A holistic approach to energy efficiency in GPU-CPU heterogeneous architectures," in *41st International Conference on Parallel Processing*. IEEE, 2012, pp. 48–57. 4

[47] A. Seewald, U. P. Schultz, E. Ebeid, and H. S. Midtiby, "`powprofiler` computations energy modeling tool," https://doi.org/10.5281/zenodo.5562457, oct 2021. [Online]. Available: https://doi.org/10.5281/zenodo.5562457 4

[48] R. Rao, S. Vrudhula, and D. N. Rakhmatov, "Battery modeling for energy aware system design," *Computer*, vol. 36, no. 12, pp. 77–87, 2003. 4

[49] S. J. Moura, F. B. Argomedo, R. Klein, A. Mirtabatabaei, and M. Krstic, "Battery state estimation for a single particle model with electrolyte dynamics," *IEEE Transactions on Control Systems Technology*, vol. 25, no. 2, pp. 453–468, 2017. 4

[50] J. Marcicki, M. Canova, A. T. Conlisk, and G. Rizzoni, "Design and parametrization analysis of a reduced-order electrochemical model of graphite/LiFePO4 cells for SoC/SoH estimation," *Journal of Power Sources*, vol. 237, pp. 310–324, 2013. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0378775313000694 4

[51] Y. Xing, W. He, M. Pecht, and K. L. Tsui, "State of charge estimation of lithium-ion batteries using the open-circuit voltage at various ambient temperatures," *Applied Energy*, vol. 113, pp. 106–115, 2014. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0306261913005746 4

[52] A. Hasan, M. Skriver, and T. A. Johansen, "Exogenous Kalman filter for state-of-charge estimation in lithium-ion batteries," in *Conference on Control Technology and Applications (CCTA)*. IEEE, 2018, pp. 1403–1408. 4, 5

[53] C. Zhang, W. Allafi, Q. Dinh, P. Ascencio, and J. Marco, "Online estimation of battery equivalent circuit model parameters and state of charge using decoupled least squares technique," *Energy*, vol. 142, pp. 678–688, 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0360544217317127 4, 5

[54] E. M. Arkin, S. P. Fekete, and J. S. Mitchell, "Approximation algorithms for lawn mowing and milling," *Computational Geometry*, vol. 17, no. 1, pp. 25–50, 2000. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0925772100000158 5

[55] H. M. Choset, K. M. Lynch, S. Hutchinson, G. Kantor, W. Burgard, L. Kavraki, S. Thrun, and R. C. Arkin, *Principles of robot motion: Theory, algorithms, and implementation*. MIT press, 2005. 5

[56] S. M. LaValle, *Planning algorithms*. Cambridge University Press, 2006. [Online]. Available: http://lavalle.pl/planning/ 5

[57] J. Araújo, P. Sujit, and J. Sousa, "Multiple UAV area decomposition and coverage," in *Symposium on Computational Intelligence for Security and Defense Applications (CISDA)*. IEEE, 2013, pp. 30–37. 5

[58] O. Artemenko, O. J. Dominic, O. Andryeyev, and A. Mitschele-Thiel, "Energy-aware trajectory planning for the localization of mobile devices using an unmanned aerial vehicle," in *25th International Conference on Computer Communication and Networks (ICCCN)*. IEEE, 2016, pp. 1–9. 5

[59] T. a. M. Cabreira, C. D. Franco, P. R. Ferreira, and G. C. Buttazzo, "Energy-aware spiral coverage path planning for UAV photogrammetric applications," *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 3662–3668, 2018. 5

[60] C. Di Franco and G. Buttazzo, "Energy-aware coverage path planning of UAVs," in *International Conference on Autonomous Robot Systems and Competitions*. IEEE, 2015, pp. 111–117. 5

[61] M. Dille and S. Singh, "Efficient aerial coverage search in road networks," in *Guidance, Navigation, and Control (GNC) Conference*. AIAA, 2013, pp. 1–20. [Online]. Available: https://arc.aiaa.org/doi/abs/10.2514/6.2013-5094 5

[62] R. Mannadiar and I. Rekleitis, "Optimal coverage of a known arbitrary environment," in *International Conference on Robotics and Automation (ICRA)*. IEEE, 2010, pp. 5525–5530. 5

[63] A. Xu, C. Viriyasuthee, and I. Rekleitis, "Optimal complete terrain coverage using an unmanned aerial vehicle," in *International Conference on Robotics and Automation (ICRA)*. IEEE, 2011, pp. 2513–2519. 5

[64] ——, "Efficient complete coverage of a known arbitrary environment with applications to aerial operations," *Autonomous Robots*, vol. 36, no. 4, pp. 365–381, 2014. 5

[65] X. Wang, P. Jiang, D. Li, and T. Sun, "Curvature continuous and bounded path planning for fixed-wing UAVs," *Sensors*, vol. 17, no. 9, 2017. [Online]. Available: https://www.mdpi.com/1424-8220/17/9/2155 5

[66] J. B. Rawlings, D. Q. Mayne, and M. Diehl, *Model predictive control: Theory, computation, and design*. Nob Hill Publishing, 2017, vol. 2. 6

[67] A. Iserles, *A first course in the numerical analysis of differential equations*. Cambridge University Press, 2009, no. 44. 6

[68] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Journal of Basic Engineering*, vol. 82, no. 1, pp. 35–45, 03 1960. [Online]. Available: https://doi.org/10.1115/1.3662552 6

[69] Paparazzi, "UAV open-source project," http://wiki.paparazziuav.org/, accessed: 2021-10-12. 6

[70] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, "ROS: An open-source robot operating system," in *ICRA Workshop on Open Source Software*, vol. 3, no. 3.2, 2009, p. 5. 6

[71] M. Ullah, A. Mohammed, and F. Alaya Cheikh, "PedNet: A spatio-temporal deep convolutional neural network for pedestrian segmentation," *Journal of Imaging*, vol. 4, no. 9, p. 107, 2018. 6

[72] F. Gavilan, R. Vazquez, and E. F. Camacho, "An iterative model predictive control algorithm for UAV guidance," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 51, no. 3, pp. 2406–2419, 2015. 7

[73] Y. Kang and J. K. Hedrick, "Linear tracking for a fixed-wing UAV using nonlinear model predictive control," *IEEE Transactions on Control Systems Technology*, vol. 17, no. 5, pp. 1202–1210, 2009. 7

[74] T. Stastny and R. Siegwart, "Nonlinear model predictive guidance for fixed-wing UAVs using identified control augmented dynamics," in *International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE, 2018, pp. 432–442. 7

[75] Z. Chao, L. Ming, Z. Shaolei, and Z. Wenguang, "Collision-free UAV formation flight control based on nonlinear MPC," in *International Conference on Electronics, Communications and Control (ICECC)*. IEEE, 2011, pp. 1951–1956. 7

[76] J. Andersson, J. Åkesson, and M. Diehl, "CasADi: A symbolic package for automatic differentiation and optimal control," in *Recent Advances in Algorithmic Differentiation*. Springer, 2012, pp. 297–307. 7

[77] ——, "Dynamic optimization with CasADi," in *51st Conference on Decision and Control (CDC)*. IEEE, 2012, pp. 681–686. 7

[78] J. A. Andersson, J. Gillis, G. Horn, J. B. Rawlings, and M. Diehl, "CasADi: A software framework for nonlinear optimization and optimal control," *Mathematical Programming Computation*, vol. 11, no. 1, pp. 1–36, 2019. 7

[79] A. Wächter and L. T. Biegler, "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming," *Mathematical programming*, vol. 106, no. 1, pp. 25–57, 2006. 7

[80] A. Seewald, "Beyond traditional energy planning: The weight of computations in planetary exploration," in *IROS Workshop on Planetary Exploration Robots: Challenges and Opportunities (PlanRobo)*. ETH Zürich, Department of Mechanical and Process Engineering, 2020, p. 3, https://doi.org/10.3929/ethz-b-000450120. [Online]. Available: https://adamseewald.cc/short/beyond2020 7