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*—In this letter, we present an online planning-scheduling approach for battery-powered autonomous aerial robots. The approach consists of simultaneously planning a coverage path and scheduling onboard computational tasks. We further derive a novel variable coverage motion robust to airborne constraints and an empirically motivated energy model. The model includes the energy contributions of the schedule based on an automatic computational energy modeling tool. Our experiments show how an initial flight plan is adjusted online as a function of the available battery, accounting for uncertainty. Our approach furthermore 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*—Motion and Path Planning, Energy and Environment-Aware Automation**

I. Introduction

USE CASES involving aerial robots span broadly. They comprise diverse planning and scheduling strategies and often require high autonomy under strict energy budgets. One such use case is coverage path planning (CPP) [1], [2], which consists of, e.g., an aerial robot visiting every point in a given space [3] while running assigned computational tasks. Here, the aerial robot might detect ground patterns and notify other ground-based actors. Such use cases arise inprecision agriculture [4] where information collection prior to an harvesting operation and damage prevention during the operation involve aerial robots [5], [6]. Microcontrollers and heterogeneous computing hardware [7] (i.e., with CPUs and GPUs) running power-demanding computational tasks are frequently mounted onto the robots in these and many other scenarios [8], [9]. We refer to onboard 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*. The energy optimization of computations schedules can be achieved by, e.g., varying the quality of service between specific bounds [10] and frequency and voltage of the computing hardware [7], [11], [12]. We focus on the former aspect and schedule the onboard computations altering their quality while simultaneously changing the quality of the coverage. Concretely, we alter how often the aerial robot detects ground patterns along with the distance of the lines that form the coverage. Fig. 1 illustrates the intuition: an aerial robot flies a plan with maximal coverage and schedule (i), that is optimized during flight to respect the battery state (ii), and altered due to, e.g., unexpected battery defects (iii).

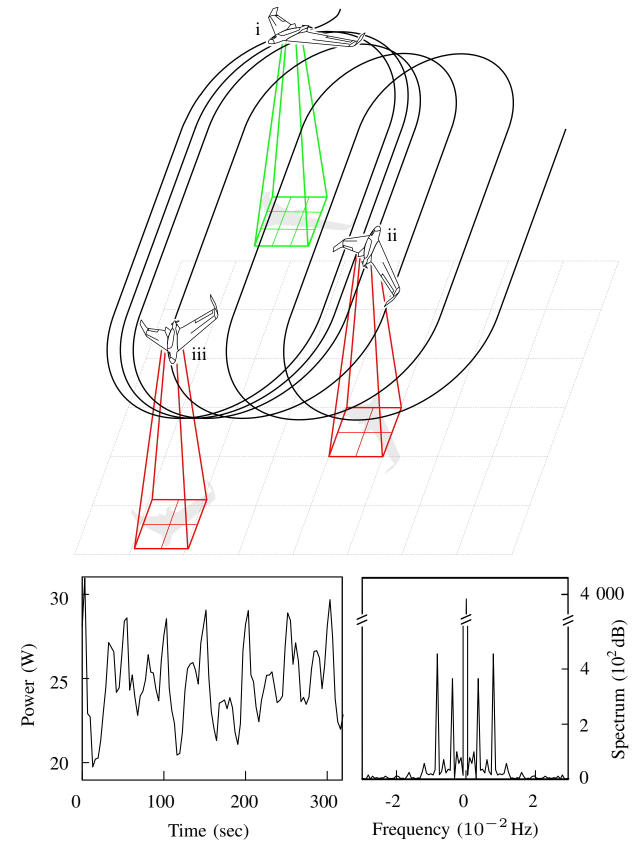


Fig. 1: The figure shows an initial plan (in i) re-planned online, changing the resolution, detection rate, or other computational aspects (in ii) and changing the number of fly-byes or other motion aspects (in iii). Bottom-left shows the energy data of a physical fixed-wing aerial robot flying a static coverage plan similar to the one illustrated here. Bottom-right shows a spectrum analysis of the energy data, revealing the periodicity of the data exploited in the overall energy model.

There are numerous planning approaches applied to a variety of robots. An instance is an algorithm selecting an energy-optimized trajectory [13] by, e.g., maximizing the operational time [14]. Many approaches apply to a small number of robots [15] and focus exclusively on planning the trajectory [16], despite compelling evidence of the energy influence of onboard consumptions [7], [11], [17], [18]. In view of the availability of powerful heterogeneous computing hardware [19], the use of onboard computations is further expected to increase in the foreseeable future [20]. In this context, planning-scheduling energy awareness is a recent research direction [11], [17], [18], [21]. Early studies (2000–2010) varied hardware-dependent aspects, e.g., frequency, voltage,

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.

along with motion aspects, e.g., motor and travel velocities [7], [11], [12], [22] whereas the literature from the past decade derives energy-aware plans-schedules in broader terms. These include simultaneous considerations for planning-scheduling in perception [17], localization [21], navigation [10], and anytime planning [18]. These studies are focused on ground-based robots [7], [17], [21], [22], yet, aerial robots are particularly affected by energy considerations, as it would be generally required to land to recharge the battery. In terms of aerial coverage, past work considers criteria including the completeness of the coverage and resolution [23], and deals with aspects such as the quality of the cover [24], but neglects the energy expenditure of computations and favors rotary-wing aerial robots rather than aerial robots broadly. 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 robot remedying in-flight failures.

Our focus is on fixed wings, i.e., airborne robots where wings provide lift, propellers provide forward thrust, and control surfaces perform maneuvering. Here, motion and computations energies are within an order of magnitude from each other [25]. 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 or vice-versa. The motion outreaching computation energy frequently happens with rotary-wing aerial robots (e.g., quadrotors or quadcopters, hexacopters, etc.), whereas the opposite occurs with 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 [7], [10], ARC Q14 [17], [21], and PackBot 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 and heuristics where both the paths and schedules variations are trajectories, varying between given bounds (i.e., physical constraints of the robot and computing hardware, quality of service, desired quality of the coverage, etc.). Past planning-scheduling studies also employ optimization techniques [11], [12], [17], [21]; some use a greedy approach [7], [18], [22]; whereas others use reinforcement learning-based approaches [10], [26]. Hybrid approaches [17] are also available, where the techniques are mixed. Both the path and schedules 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 [27], [28], whereas the battery uses an equivalent circuit model (ECM) from the literature [29], [30]. The overall model wraps these two aspects

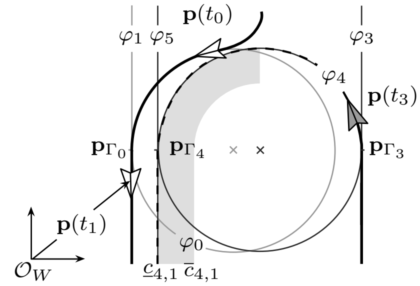


Fig. 2: Definitions II.1–4 on a slice of the plan Γ. $\mathbf{p}_{\Gamma_i}$ are triggering points in which proximity happens change of stages Γ_i. Each contains a path function φ_i and j parameters to alter the path and schedule $c_{i,1}, \dots, c_{i,j}$.

in a cohesive model that uses dynamics modeling to predict the energy behavior of future plans and schedules. In Fig. 1, collected energy data (bottom-left) and spectrum analysis (right) of a fixed wing flying CPP motivates the overall energy model: the evolution is periodic–CPP often involves repetitive motions to cover the space [1], [2]–an observation exploited 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 Section VI concludes and provides future perspectives.

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 contains 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 *computation*-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 ith *stage* Γ_i is

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

where $c_i^\rho := \{c_{i,1}, c_{i,2}, \dots, c_{i,\rho}\}$ and $c_i^\sigma := \{c_{i,\rho+1}, c_{i,\rho+2}, \dots, c_{i,\rho+\sigma}\}$ are ρ *path* and σ *computation parameters*, e.g., $c_i^\rho := \{c_{i,1}\}$ is a value that changes the distance of the coverage lines and $c_i^\sigma := \{c_{i,2}\}$ the detection rate with ρ and σ being one (see Section V). $\mathcal{C}_{i,j} := [\underline{c}_{i,j}, \overline{c}_{i,j}] \subseteq \mathbb{R}$ is the jth path parameter constraint set, and $\mathcal{S}_{i,k} := [\underline{c}_{i,\rho+k}, \overline{c}_{i,\rho+k}] \subseteq \mathbb{Z}_{\geq 0}$ is the kth computation parameter constraint set. Indices j and k serves to differentiate path and computation parameters constraints and to indicate that each parameter can have a different constraint set.

For a set $\mathbb{X}$, $\mathbb{X}_{\geq 0}$ indicates its members are positive, $\mathbb{X}_{>0}$ strictly positive, and $|\mathbb{X}|$ its cardinality. $\mathbb{Z}, \mathbb{R}$ are integers and reals. Bold letters indicate vectors. The notation $[x]$ denotes positive naturals up to x, i.e., $\{0, 1, \dots, x\}$, $[x]_{>0}$ strictly positive naturals, i.e., $\{1, 2, \dots, x\}$, x' the transpose of x, and $[\underline{x}, \overline{x}]$ the upper/lower bounds of a parameter x, i.e.,

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

The function φ_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 $\mathbf{p}$ and path parameters c_i^ρ 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* $\mathbf{p}_{\Gamma_i}$ describes the transition between stages. The *final point* is the last triggering point $\mathbf{p}_{\Gamma_l}$ relative to the last stage Γ_l.

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

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

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

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

Fig. 2 illustrates the concepts in Definitions II.1–4. $\varphi_0, \dots, \varphi_5$ are path functions. φ_0 and φ_4 are circles, while φ_1, φ_3, and φ_5 are lines. They are relative to different stages $\Gamma_1, \dots$ but Γ_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 $c_{1,1}, \dots, c_{4,1}$ illustrated by the gray area in the figure.

A convenient way of defining Γ 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^ρ.

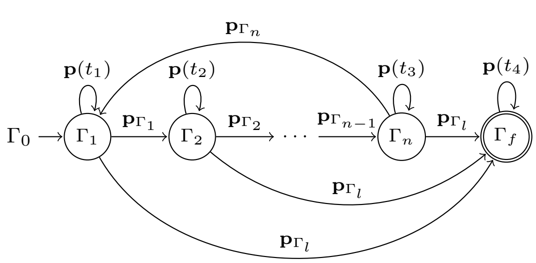


Fig. 3: Plan Γ with n primitive stages in Def II.5.

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 $\in \mathbb{R}_{>0}$. Γ_f is the accepting stage, indicating the plan is complete, Γ_0 the initial stage where the aerial robot 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 Γ to cover the polygon, whereas the *re-planning-scheduling problem* is finding the energy-aware trajectory of parameters c_i in time.

Here, c_i denotes a row vector with both the path and computation 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 computation parameters on the battery at future time instants. To this end, Sections III-A–C provide models for the overall and computations energies as well as battery evolution.

A. Overall energy model

The collected energy data and corresponding spectrum analysis in Fig. 1 show the energy of a static coverage plan. It is relative to one flight of a series of flights for CPP exhibiting periodic behavior in a precision agriculture use case [25]. Assuming the primitive paths have approximately the same length and the aerial robot has a fixed ground speed, the data exhibits 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 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 $h : \mathbb{R}_{\geq 0} \to \mathbb{R}$ 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

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

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

$$C = (1/T)\Big[1 \; \overbrace{1 \quad 0 \quad \cdots \quad 1 \quad 0}^{2r}\Big], \quad (7)$$

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 computation 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 Γ_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 computation parameters respectively.

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

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

shifts the base frequency α_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 ρ columns are zeros and the remaining σ 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 the Kalman filter in Section IV-B, to refine the states $\mathbf{q}$ and model the energy of the aerial robot flying under diverse conditions.

Matrices A and C are constructed such that 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

$$\mathbf{q}_0 = \begin{bmatrix} a_0 & a_1/2 & b_1/2 & \cdots & a_r/2 & b_r/2 \end{bmatrix}', \quad (9)$$

i.e., h, y are both harmonic signals with the same frequencies [31].

$\hat{\mathbf{u}}$ in Eq. (8) is then a scale transformation

$$\hat{\mathbf{u}}(t) := \mathrm{diag}(\nu_i) c_i(t) + \tau_i, \quad (10)$$

where $\mathrm{diag}(x)$ is a diagonal matrix with items of a set x 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.

For ease of notation, we assume that the coverage time evolves linearly. Path parameters c_i^ρ can be transformed into a time measure with scaling factors

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

$\forall j \in [\rho]_{>0}$ 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. (11), computation parameters c_i^σ can be transformed into an instantaneous energy measure with

$$\nu_{i,j} = (g(\bar{c}_{i,j}) - g(\underline{c}_{i,j}))/(\bar{c}_{i,j} - \underline{c}_{i,j}), \quad (12a)$$
$$\tau_{i,j} = \underline{c}_{i,j}(g(\underline{c}_{i,j}) - g(\bar{c}_{i,j}))/(\bar{c}_{i,j} - \underline{c}_{i,j}) + g(\underline{c}_{i,j}), \quad (12b)$$

$\forall j \in [\rho+1, n]$. 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 [32], [33] or different techniques, such as regressional analysis [27], [34], [35], aiding the selection of hardware- or software-specific parameters. This section presents an energy model based on our early studies [27], [28], which relies on regressional analysis to quantify the computations energy of any configuration of computations c_i^σ within the bounds (see Definition II.1).

The model compromises an automatic modeling and profiling tool [27] named `powprofiler` distributed under the open-source MIT license. It is segmented into two layers. In the *measurement layer*, the tool measures a discrete set of computation parameters and infers the energy of the remaining in the *predictive layer* via a piecewise linear regression.

We 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, computation 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 computation 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 expressed in watts or overall expressed in joules. 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. (12), assuming the computations energy behaves linearly between $\underline{c}_i^\sigma$ and $\bar{c}_i^\sigma$, otherwise

$$\begin{aligned} g(c_i^\sigma) = {} & (\mathbf{g}(\lceil c_i^\sigma \rceil, \mathcal{T}_1) - \mathbf{g}(\lfloor c_i^\sigma \rfloor, \mathcal{T}_2)) \\ & (c_i^\sigma - \lfloor c_i^\sigma \rfloor)/(\lceil c_i^\sigma \rceil - \lfloor c_i^\sigma \rfloor) + \mathbf{g}(\lfloor c_i^\sigma \rfloor, \mathcal{T}_2), \end{aligned} \quad (13)$$

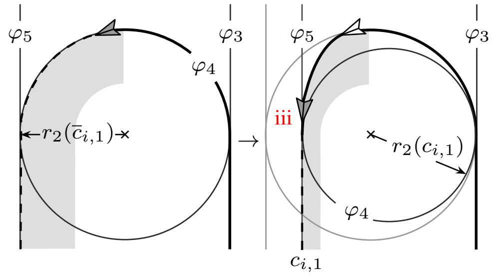


Fig. 4: Alteration of the path parameter $c_{i,1}$, the radius of the circle (i.e., the alteration of the plan in Fig. 1).

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 [36] with varying complexity, accuracy, and ease of implementation ranging from accurate but costly physical models [37], to abstract models [29], [30] 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 [29], [30].

The battery SoC changes according to [38], i.e.,

$$\dot{b}(y(t)) = -k_b I(y(t))/Q_c, \tag{14}$$

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 constant nominal capacity measured in amperes per hour. k_b is a battery coefficient added to [38] 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 [29] and depends on the SoC [38].

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

$$I(y(t)) = (V - \sqrt{V^2 - 4R_r y(t)})/(2R_r). \tag{15}$$

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

$$\mathcal{Y}(t) := \{y \mid y \in [0, b\,Q_c V] \subseteq \mathbb{R}_{\geq 0}\}, \tag{16}$$

where $b\,Q_c V$, the maximum instantaneous energy consumption measured in watts, is derived from Eq. (14–15).

IV. Planning-Scheduling

This section solves the problem described in Section II-B. It provides a plan and re-plans-schedules such plan energy-wise.

Algorithm 1 Zamboni-like motion for CPP

1: **for all** $t \in \mathcal{T}$ **do**
2:   **if** $\mathbf{p} = \mathbf{p}_{\Gamma_l}$ in Definition II.3 **then return** Γ
3:   **if** $\mathbf{p} = \mathbf{p}_{\Gamma_i}$ **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. (18) 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.1–4

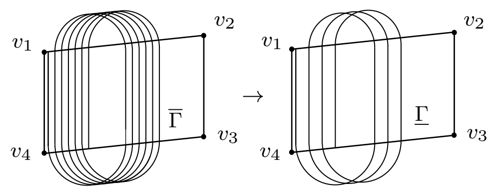


Fig. 5: Zamboni-like motion: $(\bar{\Gamma})$ with four primitive paths (Lines 9–14 in Algorithm 1) can be re-planned $(\underline{\Gamma})$ with r_2 in Eq. (17).

A. Coverage

There are various approaches in the literature to solve CPP problems (e.g., Section II-B). Those that ensure completeness are NP-hard [39] 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], [40] and aerial [23], [41], [42] 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 [43]–[45] and are generally unable to fly quick turns [46].

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* [41], but additionally allows variable CPP at the very core of this work. Although cover variability is already considered in the literature [23], it is limited to boustrophedon motion for rotary wings. The novel motion is termed *Zamboni-like motion* and is composed of four primitive paths (see Definition II.5): two lines φ_1, φ_2 and two circles φ_3, φ_4.

We 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 Γ 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, a complex covering is possible by, e.g., dividing $\mathcal{Q}^v$ into cells to be easily covered and subsequently covering each cell [1].

Algorithm 2 Coverage re-planning-scheduling

1: **for all** $t \in \mathcal{T}$ **do**
16: $\quad \mathbf{q}(\mathcal{K} \setminus \{t+N\}), c_i^\sigma(\mathcal{K}) \leftarrow$ 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. (19) 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. (14)
22: $\quad\quad k \leftarrow k + h$
23: $\quad t_b \leftarrow k - t$
24: $\quad t_r \leftarrow (\mathrm{diag}(\nu_i^\rho) c_i^\rho(t) + \tau_i^\rho)[\overbrace{1 \quad 1 \quad \cdots \quad 1}^{\rho}] - t$
25: $\quad$ **if** $t_r > t_b$ **then**
26: $\quad\quad c_i^\rho(t) \leftarrow$ find c_i^ρ with $t_r \in [0, t_b]$, otherwise take $\underline{c}_i^\rho$
27: $\quad \hat{\mathbf{q}}(t+h) \leftarrow$ estimate $\mathbf{q}$ in Eq. (4a) with energy sensor $\Upsilon(t)$
28: $\quad \hat{y}(t+h) \leftarrow$ derive y from Eq. (4b) with est. state $\hat{\mathbf{q}}(t+h)$

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

$$r_2(c_{i,1}) := \sqrt{r^2 + c_{i,1}}, \tag{17}$$

is expressed as a 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

$$\varphi_{|\Gamma|+4} := (x - x_{\mathbf{p}_{|\Gamma|+3}} + r_2)^2 + (y - y_{\mathbf{p}_{|\Gamma|+3}})^2 - r_2^2, \tag{18}$$

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 $\varphi_1, \dots, \varphi_4$. The remaining Γ 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 φ_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 optimization as well as heuristics-related approaches [11], [12], [17], [21]. We similarly derive an optimal control problem and a greedy approach 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 $N \in \mathbb{R}_{>0}$ [47]. We utilize MPC to derive the trajectory of the computation parameters and the greedy approach with heuristics remaining coverage time for the path parameters.

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

$$\max_{\mathbf{q}(t), c_i(t)} l_f(\mathbf{q}(t_f), t_f) + \int_{t_0}^{t_f} l(\mathbf{q}(t), c_i(t), t)\, dt, \tag{19a}$$

$$\text{s.t. } \dot{\mathbf{q}} = f(\mathbf{q}(t), c_i(t), t), \tag{19b}$$

$$c_{i,j}(t) \in \mathcal{C}_{i,j},\ c_{i,\rho+k}(t) \in \mathcal{S}_{i,k}\ \forall j \in [\rho]_{>0},\ k \in [\sigma]_{>0}, \tag{19c}$$

$$\mathbf{q}(t) \in \mathbb{R}^m,\ y(t) \in \mathcal{Y}(t), \tag{19d}$$

$$\mathbf{q}(t_0) = \hat{\mathbf{q}}_0 \text{ given (last estimated state), and} \tag{19e}$$

$$b(t_0) = b_0 \text{ given}, \tag{19f}$$

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 cost function with the quadratic expression

$$l(\mathbf{q}(t), c_i(t), t) = \mathbf{q}'(t) Q \mathbf{q}(t) + c_i'(t) R c_i(t), \tag{20}$$

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 [47]

$$l_f(\mathbf{q}(T), T) = \mathbf{q}'(T) Q_f \mathbf{q}(T), \tag{21}$$

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

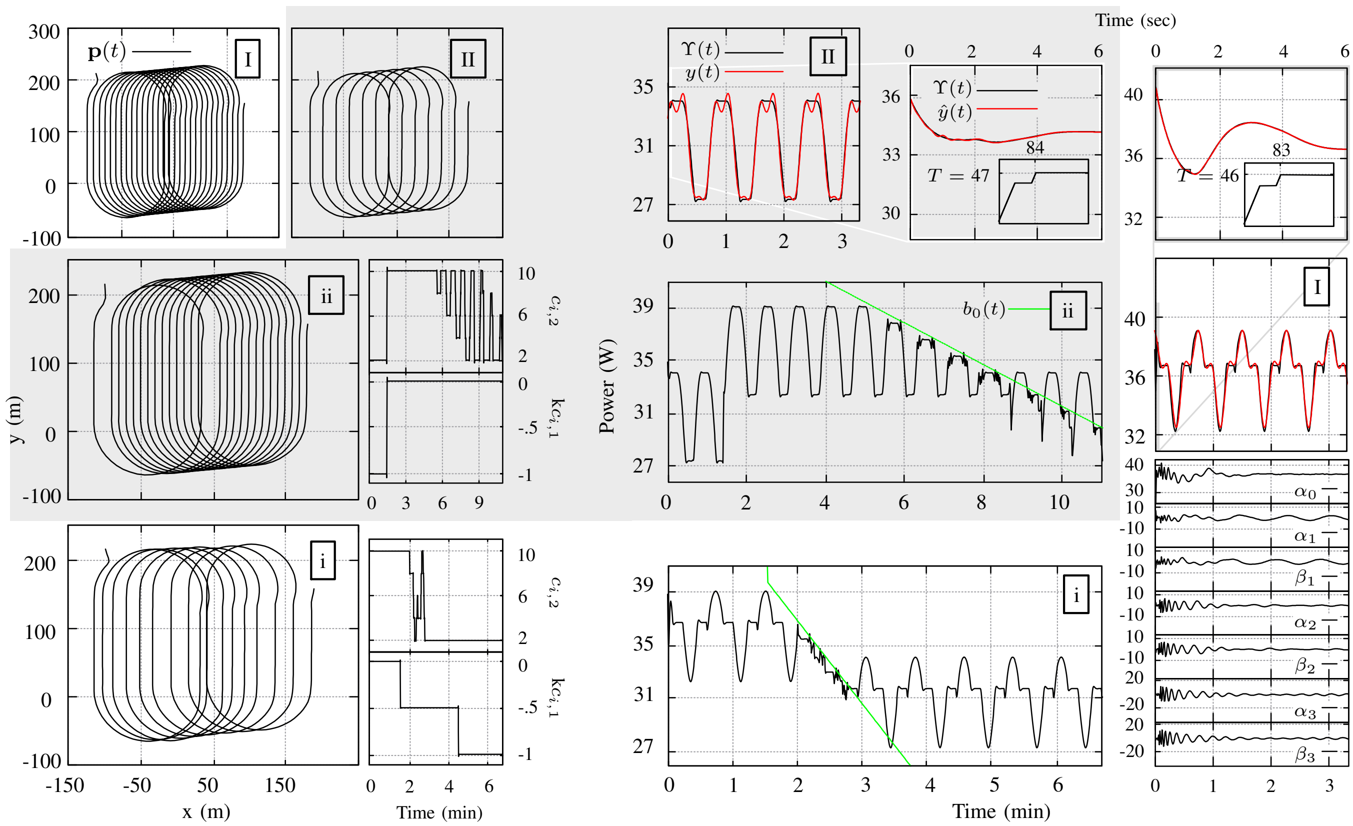
Furthermore, Eq. (19b) is the differential periodic energy model in Eq. (4). The model requires a value of the period T, which is simply the time needed to fly the four primitive paths in the Zamboni-like motion, i.e., the time elapsed between two positive evaluations of the condition on Line 5.

Eq. (19c) are the parameters constraints sets in Definition II.1. Eq. (19d) are the state and output constraints in Eq. (16) that evolve the battery model in Eq. (14). Eq. (19e) is the state guess estimated via state estimation (the very first estimate is given). Eq. (19f) is the initial battery SoC from, e.g., flight controller.

Line 16 in Algorithm 2 contains a transcribed version of the OCP in Eq. (19) into a nonlinear program (NLP) that can be solved with available NLP solvers [47]. Its solution leads to both trajectories of computation 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): the set $\mathcal{K}$ is used for the numerical simulation, whereas $\mathcal{T}$ is for re-planning, meaning that h tunes the precision and the frequency of re-planning in $\mathcal{K}$, $\mathcal{T}$ respectively. The functions l_d, b_d are the discretized versions of Eq. (20) and Eq. (14), with, e.g., Runge-Kutta or Euler methods.

Lines 17–23 estimate the time needed to completely drain the battery, exploiting the SoC already predicted previously on Line 16. The path parameters and thus the coverage is then re-planned accordingly on Lines 24–26 using the heuristics with the scaling factors from Eq. (11). Concretely, these lines implement the greedy approach by decreasing the path parameters of a given value δ_i or similarly increasing the parameters when $t_r \leq t_b$ within the bounds (this latter analogous case is not shown explicitly in Algorithm 1 but implemented in Sec. V). Lines 27–28 estimate the energy model's state with current energy sensor reading Υ, with, e.g., Kalman filter.

Algorithm 2 implements Eq. (19) for the purpose of energy-aware re-planning-scheduling of Γ from Algorithm 1, i.e, Lines 16–28 continue after Line 15 in Algorithm 1.



(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.

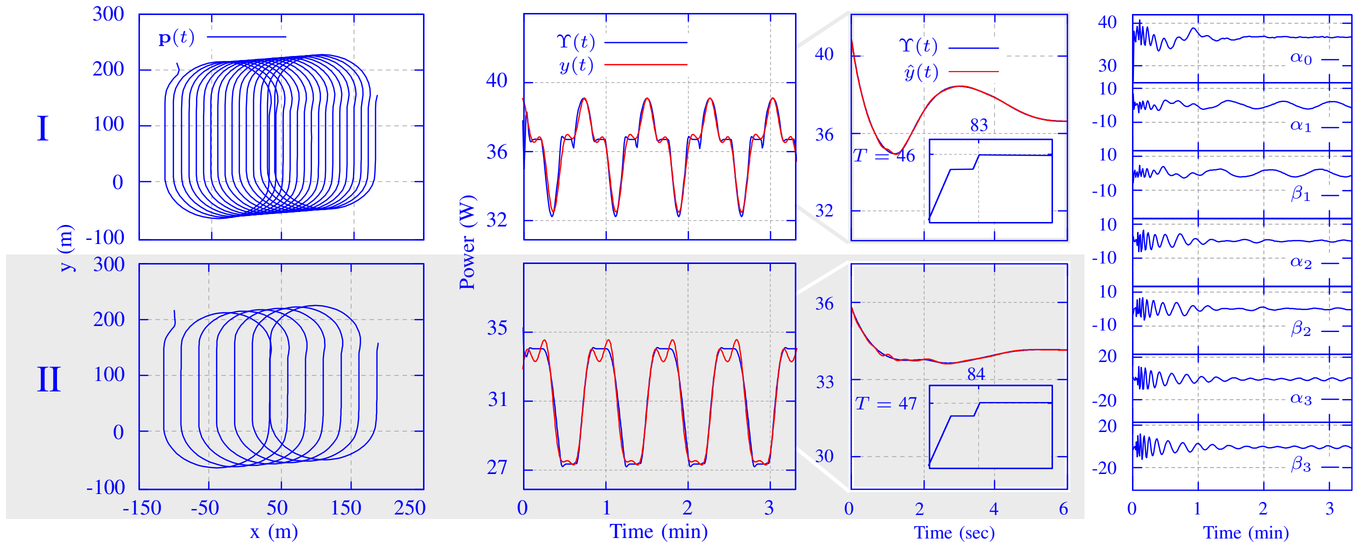
V. Numerical Simulations

Numerical simulations of Algorithms 1–2 in this section are implemented in MATLAB (R) and are extended with the computations energy model on NVIDIA (R) Jetson Nano (TM) heterogeneous computing hardware. These simulations complement early data of physical flights of a static coverage plan with the open-source Paparazzi flight controller. The computing hardware carries a camera as a peripheral and is evaluated independently of the aerial robot with `powprofiler` (see Section III-B). The scheduler, implemented using the Robot Operating System (ROS) middleware, 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) [48], also implemented using ROS. The planner varies the path parameter $c_{i,1}$ in Eq. (17) 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, $c_i := \begin{bmatrix} c_{i,1} & c_{i,2} \end{bmatrix}', \forall i$, along δ_i in the greedy approach.

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 as described in this letter.

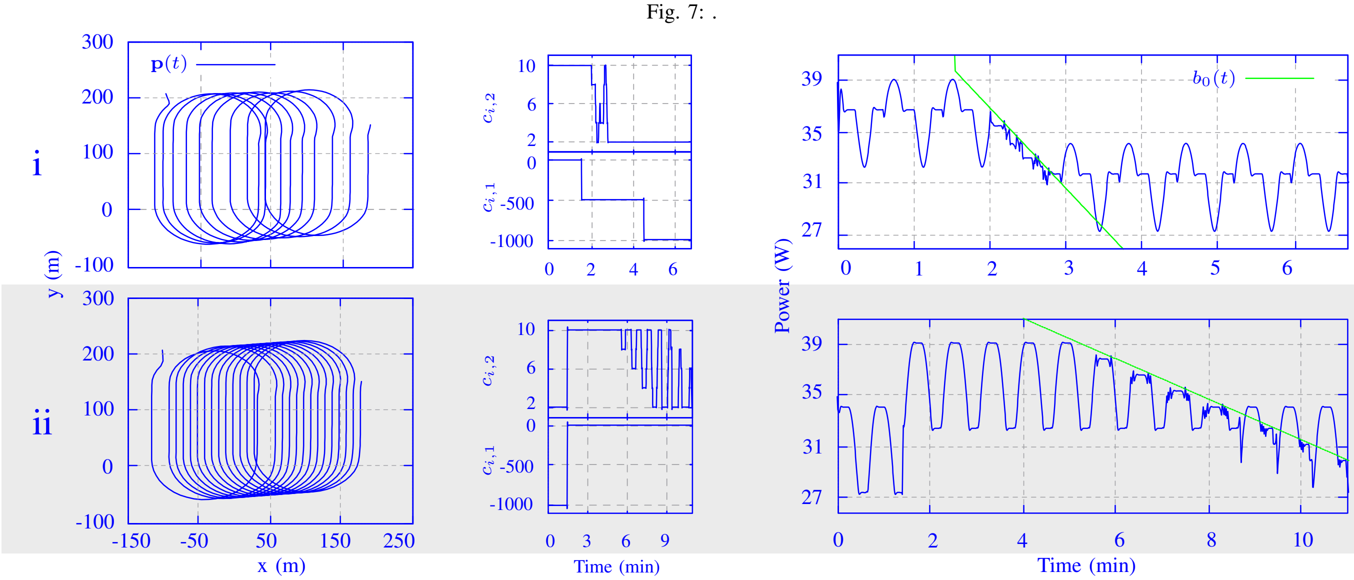
Fig. 6a illustrates the same plan Γ under different conditions. Flights I–i have a constant wind speed of five meters per second, a wind direction of zero degrees, and initial parameters $c_{i,1}, c_{i,2}$ values of zero and ten (i.e., full r_2 and detection). Flights II–ii (see added gray background for clarity) are 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). The initial values of path and computation parameters are chosen to represent the highest and lowest configurations in the search space in I–i and II–ii respectively, modeling the behavior of the best- and worst-case scenario. Different search strategies are possible by, e.g., running an ideal instance of planning-scheduling prior to the flight.

Fig. 6b illustrates first the power (Υ on Line 27 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 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. (16)). Flight ii simulates the opposite scenario: the lowest configuration of parameters and no battery defects. The path parameter increases as soon as the algorithm has estimated enough data and the computation parameter decreases matching the battery discharge rate. For both cases, scaling factors are derived empirically similarly to δ_i set to two hundred fifty, the horizon N is set to six seconds as in relevant literature [49], [50], order r is three (see Fig. 1), and the matrices Q, R, Q_f are chosen such that the cost is merely squared control. h is set to one-hundredth of a second and to one second in $\mathcal{K}, \mathcal{T}$ respectively to allow sufficient precision and re-planning online. The figure further details the energy model's estimate (see detail view for I–II)



(a) Trajectories without re-planning (b) Energies, detail for first instants, period T evolutions (a) States $(\alpha_0, \dots)$ evol. for I

Fig. 7: .



(a) Re-planned trajectories (b) Parameters $(c_{i,1}, c_{i,2})$ evol. (c) Energies and batteries evol.

Fig. 8: .

on an initial slice of the model ($\hat{y}$), power (Υ), and period (T). The bottom detail of I illustrates the evolutions of the state $\mathbf{q}$ in time, concluding that approximately two periods suffice for a consistent estimate.

Output MPC on Line 16 relies on a software framework for nonlinear optimization called CasADi [51], and the popular NLP solver IPOPT [52]; both are open-source.

VI. Conclusions and Future Directions

This 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 stage the aerial robot flies a path and runs the computations, allowing extensibility in terms of constructs and approaches.

Indeed, we are currently extending the results to a standard flight controller to enable physical experiments. The guidance on the coverage, coverage with variable altitude, and distributed planning-scheduling merit additional 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 [53], in line with relevant literature [17], [21]. Further directions include the use of a purely optimization-based technique, e.g., MPC derives both the path and computation parameters trajectories and the study of different energy models. Amongst others, these include aperiodic energy models, different linear combinations of the variations of parameters, and stage-dependent energy models.

References

[1] H. Choset, "Coverage for robotics–A survey of recent results," *Ann. of Mathematics and Artif. Intell.*, vol. 31, pp. 113–126, 2001. 1, 2, 5

[2] E. Galceran and M. Carreras, "A survey on coverage path planning for robotics," *Robot. and Autonomous Syst.*, vol. 61, no. 12, pp. 1258–1276, 2013. 1, 2, 5

[3] T. Cabreira, L. B. Brisolara, and P. R. Ferreira Jr., "Survey on coverage path planning with unmanned aerial vehicles," *Drones*, vol. 3, no. 1, 2019. 1

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

[5] V. Puri, A. Nayyar, and L. Raja, "Agriculture drones: A modern breakthrough in precision agriculture," *J. of Statist. and Manage. Syst.*, vol. 20, no. 4, pp. 507–518, 2017. 1

[6] 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 *Conf. Series: Earth and Environmental Science*, vol. 275, no. 1. IOP Publishing, 2019, p. 012022. 1

[7] 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 Int. Conf. on Advanced Robot.* IEEE, 2005, pp. 492–497. 1, 2

[8] 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 *Int. Conf. on Intell. Robots and Syst.*, 2019, pp. 237–243. 1

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

[10] 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 Int. Conf. on Robot. Comput.* IEEE, 2019, pp. 212–219. 1, 2

[11] J. Brateman, C. Xian, and Y.-h. Lu, "Energy-effcient scheduling for autonomous mobile robots," in *IFIP Int. Conf. on Very Large Scale Integration*. IEEE, 2006, pp. 361–366. 1, 2, 6

[12] W. Zhang and J. Hu, "Low power management for autonomous mobile robots using optimal control," in *46th Conf. on Decis. and Control*. IEEE, 2007, pp. 5364–5369. 1, 2, 6

[13] Y. Mei, Y.-H. Lu, Y. C. Hu, and C. G. Lee, "Energy-efficient motion planning for mobile robots," in *Int. Conf. on Robot. and Automation*, vol. 5. IEEE, 2004, pp. 4344–4349. 1

[14] M. Wahab, F. Rios-Gutierrez, and A. El Shahat, "Energy modeling of differential drive robots," in *Southeast Conf.* IEEE, 2015. 1

[15] C. H. Kim and B. K. Kim, "Energy-saving 3-step velocity control algorithm for battery-powered wheeled mobile robots," in *Int. Conf. on Robot. and Automation*. IEEE, 2005, pp. 2375–2380. 1

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

[17] P. Ondrúška, C. Gurău, L. Marchegiani, C. H. Tong, and I. Posner, "Scheduled perception for energy-efficient path following," in *Int. Conf. on Robot. and Automation*. IEEE, 2015, pp. 4799–4806. 1, 2, 6, 8

[18] S. Sudhakar, S. Karaman, and V. Sze, "Balancing actuation and computing energy in motion planning," in *Int. Conf. on Robot. and Automation*. IEEE, 2020, pp. 4259–4265. 1, 2

[19] 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," *J. of Low Power Electron. and Applications*, vol. 7, no. 2, p. 10, 2017. 1

[20] 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 Int. Conf. on Advanced Robot.* IEEE, 2019, pp. 773–778. 1

[21] 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 Robot. and Automation Letters*, vol. 3, no. 3, pp. 1840–1847, 2018. 1, 2, 6, 8

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

[23] C. Di Franco and G. Buttazzo, "Energy-aware coverage path planning of UAVs," in *Int. Conf. on Autonomous Robot Syst. and Competitions*. IEEE, 2015, pp. 111–117. 2, 5

[24] ——, "Coverage path planning for UAVs photogrammetry with energy and resolution constraints," *Journal of Intelligent & Robotic Systems*, vol. 83, no. 3, pp. 445–462, 2016. 2

[25] 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 Int. Conf. on Robot. Comput.* IEEE, 2020, pp. 135–142. 2, 3

[26] D.-K. Ho, K. Ben Chehida, B. Miramond, and M. Auguin, "Towards a multi-mission QoS and energy manager for autonomous mobile robots," in *2nd Int. Conf. on Robot. Comput.* IEEE, 2018, pp. 270–273. 2

[27] A. Seewald, U. P. Schultz, E. Ebeid, and H. S. Midtiby, "Coarse-grained computation-oriented energy modeling for heterogeneous parallel embedded systems," *Int. J. of Parallel Program.*, vol. 49, no. 2, pp. 136–157, 2021. 2, 4

[28] A. Seewald, U. P. Schultz, J. Roeder, B. Rouxel, and C. Grelck, "Component-based computation-energy modeling for embedded systems," in *SIGPLAN Int. Conf. on Syst., Program., Lang., and Applications: Software for Humanity*. ACM, 2019, pp. 5–6. 2, 4

[29] H. Hinz, "Comparison of lithium-ion battery models for simulating storage systems in distributed power generation," *Inventions*, vol. 4, 2019. 2, 5

[30] 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. 2, 5

[31] A. Seewald, *Energy-aware coverage planning and scheduling for autonomous aerial robots*, 1st ed. Syddansk Universitet, 2021. 4

[32] A. Marowka, "Energy-aware modeling of scaled heterogeneous systems," *Int. J. of Parallel Program.*, vol. 45, no. 5, pp. 1026–1045, 2017. 4

[33] T.-J. Yang, Y.-H. Chen, and V. Sze, "Designing energy-efficient convolutional neural networks using energy-aware pruning," in *Conf. on Comput. Vision and Pattern Recognit.* IEEE, 2017, pp. 5687–5695. 4

[34] 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 Int. Conf. on Parallel Processing*. IEEE, 2014, pp. 371–380. 4

[35] 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 Int. Conf. on Parallel Processing*. IEEE, 2012, pp. 48–57. 4

[36] R. Rao, S. Vrudhula, and D. Rakhmatov, "Battery modeling for energy aware system design," *Comput.*, vol. 36, no. 12, pp. 77–87, 2003. 5

[37] 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," *J. of Power Sources*, vol. 237, pp. 310–324, 2013. 5

[38] A. Hasan, M. Skriver, and T. A. Johansen, "Exogenous Kalman filter for state-of-charge estimation in lithium-ion batteries," in *Conf. on Control Technol. and Applications*. IEEE, 2018, pp. 1403–1408. 5

[39] E. M. Arkin, S. P. Fekete, and J. S. Mitchell, "Approximation algorithms for lawn mowing and milling," *Comput. Geometry*, vol. 17, no. 1, pp. 25–50, 2000. 5

[40] S. M. LaValle, *Planning algorithms*. Cambridge Univ. Press, 2006. 5

[41] J. Araújo, P. Sujit, and J. Sousa, "Multiple UAV area decomposition and coverage," in *Symposium on Comput. Intell. for Security and Defense Applications*. IEEE, 2013, pp. 30–37. 5

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

[43] M. Dille and S. Singh, "Efficient aerial coverage search in road networks," in *Guid., Navigation, and Control Conf.* AIAA, 2013, pp. 1–20. 5

[44] R. Mannadiar and I. Rekleitis, "Optimal coverage of a known arbitrary environment," in *Int. Conf. on Robot. and Automation*. IEEE, 2010, pp. 5525–5530. 5

[45] A. Xu, C. Viriyasuthee, and I. Rekleitis, "Efficient complete coverage of a known arbitrary environment with applications to aerial operations," *Autonomous Robots*, vol. 36, no. 4, pp. 365–381, 2014. 5

[46] 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. 5

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

[48] M. Ullah, A. Mohammed, and F. Alaya Cheikh, "PedNet: A spatio-temporal deep convolutional neural network for pedestrian segmentation," *J. of Imaging*, vol. 4, no. 9, p. 107, 2018. 7

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

[50] T. Stastny and R. Siegwart, "Nonlinear model predictive guidance for fixed-wing UAVs using identified control augmented dynamics," in *Int. Conf. on Unmanned Aircr. Syst.* IEEE, 2018, pp. 432–442. 7

[51] 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. 8

[52] A. Wächter and L. T. Biegler, "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming," *Mathematical Program.*, vol. 106, no. 1, pp. 25–57, 2006. 8

[53] A. Seewald, "Beyond traditional energy planning: The weight of computations in planetary exploration," in *IROS Workshop on Plan. Exploration Robots*. ETH Zürich, Dept. of Mech. and Process Eng., 2020, p. 3. 8