A Health-Aware Framework for Online Replanning of Unmanned Aerial Vehicles Missions under Faulty Conditions

Timothy Darrah, [1] **Marcos Quiñones-Grueiro,** [1] **Gautam Biswas,** [1] **Jeremy Frank** [2]

[1] Vanderbilt University, Institute for Software Integrated Systems
[2] NASA Ames Research Center, Intelligent Systems Division
timothy.s.darrah@vanderbilt.edu, marcos.quinones@vanderbilt.edu, gautam.biswas@vanderbilt.edu, jeremy.frank@nasa.gov

Abstract

The use of Unmanned Aerial Vehicles (UAVs) has drastically increased over the last 10 years for a wide range of applications that include topological surveys, package delivery, and surveillance. Maintaining safe and reliable operations is of utmost importance to minimize the risk of loss or injury from unforeseen events, such as the occurrence of abrupt faults in uncertain environments. In this work, a novel *Health-Aware* framework for replanning under faulty conditions with system-level state of health information is presented. Previous works in this field do not account for *wear-and-tear* degradation in conjunction with abrupt faults. We describe the problem of integrating health-state predictions, execution, and replanning. Health-state information can provide additional constraints to a system during replanning, resulting in plans that do not violate safety or performance constraints. We demonstrate the use of a machine learning-based health-state prognostics system that provides highly accurate remaining flight time estimates to an on-board replanning agent. We show this machine-learning approach outperforms a standard discharge-based health-state prediction model in a simulated UAV domain.

1 Introduction

The use of autonomous systems such as Unmanned Aerial Vehicles (UAVs) in civilian and military operations has drastically increased over the last 10 years, requiring significant advances in health management technologies to maintain safe and reliable operations. Most of these technologies employ some form of *Fault-Adaptive Control* (Ahmed, Quinones-Grueiro, and Biswas 2023), which seeks to alter the parameters of the controller or switch controllers to accommodate the fault. These methods are focused on low-level control and are primarily used to maintain a safe trajectory. What they don't account for, however, is the health of the individual components and the subsequent health of the overall system after the fault occurrence. This recognition is what we mean by *Health-Aware*, where component and system *State of Health* (SOH) information is used to generate safer and more reliable plans when faulty conditions occur during a flight. In this context, (1) the system is operating in a degraded but functional state capable of satisfying system-level safety and performance constraints, and (2) an abrupt fault occurs in a component and its magnitude has been computed. The combination of the fault and the preexisting degraded state means the UAV is no longer able to finish its current mission while safely returning to base, but could potentially finish part of its existing mission or perhaps an alternate mission.

A motivating example is depicted in Figure 1, where a UAV operates in an urban environment with some approximate level of degradation (normal *wear and tear*) in its 8 motors and 6-cell battery. The UAV must reach a predetermined number of waypoints, stop at each for a time period, T_s, and return back to the starting location safely. On the left, no fault occurs and it's able to complete its flight. On the right, a fault has occured with no SOH estimation or replanning, and the UAV fails to compelte its mission.

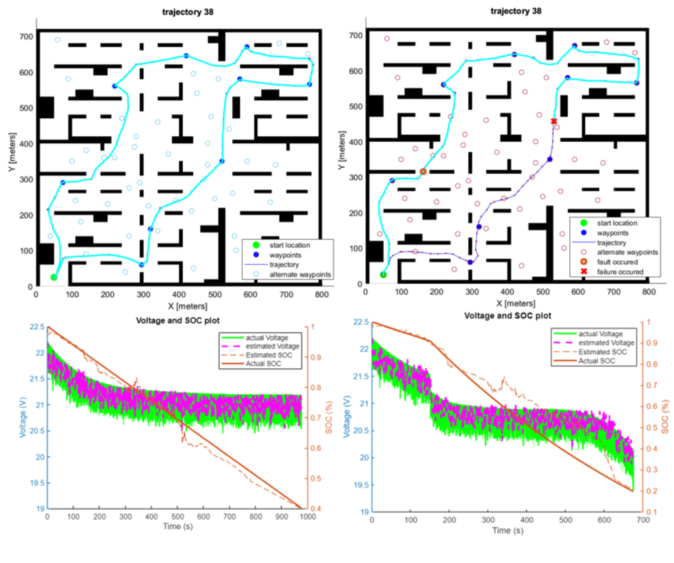

Figure 1: Nominal flight with no fault (L); failed flight with a battery cell loss (R).

1.1 Contribution

The key contribution of this work is the development of an online replanning framework that accounts for system-level degradation and the onset of an abrupt fault. Such a framework does not currently exist in the literature. Methods in use today rely on simple prediction models that do not utilize accurate health data about the system itself. Absent of accurate system-level health information, we cannot certify that the replanner will always generate safe solutions.

1.2 Problem Statement

We formulate this problem by first defining the UAV as a dynamical system model and characterize its usage-based degradation over time with individual component degradation models.

Definition 1 (Dynamical System Model)

$$\begin{aligned} \mathbf{x}_{n+1} &= f(\mathbf{x}_n, \mathbf{u}_n, \vartheta_n, \mathbf{w}_x) \\ \mathbf{y}_n &= h(\mathbf{x}_n, \mathbf{u}_n, \vartheta_n, \mathbf{w}_y), \end{aligned} \tag{1}$$

where $\mathbf{x}_n \in \Re^m$ denotes the state vector describing the dynamics of the system; $\mathbf{y}_n \in \Re^p$ represents the measured variables in the system at time step n; $\mathbf{u}_n \in \Re^o$ denotes the input; $\vartheta_n \in \Re^q$ is the set of component and system parameters; and $\mathbf{w}_x$ and $\mathbf{w}_y$ capture the uncertainties and disturbances associated with the system model and the system outputs. Equations f and h are the state update and system output functions, which together, characterize the system dynamics.

Definition 2 (Component Degradation Model)

$$\gamma_{in+1} = g(\gamma_{in}, \alpha_{in}, \mathbf{x}_n, \mathbf{w}_d) \tag{2}$$

where $\gamma_\mathbf{i} \in \Re^q$ is the set of degrading parameters for component i; $\alpha_{in} \in \Re^q$ define the degradation rates for the set of component degradation parameters; and $\mathbf{w}_d$ captures the uncertainty in the degradation models.

Each waypoint, $w_i \in \mathcal{W}$, $i = \{0, 1, ..., N\}$ is defined by 3-dimensional spatial coordinates in a global reference frame along with expected arrival time (t) and reward value (s):

$$w_i \quad = (x, y, z, t, s), \tag{3}$$

where any waypoint coordinates (x, y, z) are bounded by the area of operations (AO) of the UAV. The distribution of the system parameters, ϑ, and the degradation parameters, γ, are known. With the exception of take-off and landing procedures, the altitude, z, is fixed, which restricts the problem to the x-y plane, and the velocity set-point, v_s, is assigned prior to flight. A dynamically feasible reference trajectory for the vehicle ($\mathcal{T}$) is generated offline using Probabilistic Roadmaps (PRM) and the minimum jerk method based on v_s, $\mathcal{W}$, and the AO (Darrah et al. 2022a).

The set of system performance and safety constraints, $\wp$, maps a performance parameter to a Boolean domain $\{\top, \bot\}$ where each constraint is described using a linear temporal logic (LTL), and takes the form

$$\wp_i : \mathbf{y}_t, \vartheta_t \to \{\top, \bot\} \tag{4}$$

Equation 4 returns true if at least one of the performance constraint functions returns false at time t, signifying that a constraint has been violated. Together, the trajectory, $\mathcal{T}$, the associated set of waypoints, $\mathcal{W}^\mathcal{T}$, the stopping period, t_s, and the system level safety and performance constraints, $\wp$, comprise a mission plan, $\mathcal{MP}$:

$$\mathcal{MP} \quad = (\mathcal{T}, \mathcal{W}^\mathcal{T}, t_s, \wp) \tag{5}$$

Under nominal conditions, the UAV executes $\mathcal{MP}$ succesfully; under faulty conditions, a fault detection, isolation, and identification (FDII) agent detects (1) that a fault has occured, (2) that component or subsystem n has failed $\mathbf{f_n}$, and (3) the magnitude of the fault $||\overrightarrow{\mathbf{f_n}}||$ (Bregon et al. 2014). The diagnosis recomputes the system and degradation parameters of the dynamical system model, and a new state is computed. The updated state and degradation estimates are used by the remaining useful life (RUL) predictor to determine the maximum time remaining for the flight. This value is then used by the replanner to define a subset of waypoints that can be visited and allow the UAV to safely return to base.

1.3 Paper organization

The rest of the paper is organized as follows: A literature review of prognostics and replanning is covered in Section 2. The online System Level Prognostics and replanning framework is discussed in Section 3. The experiments that demonstrate the framework are discussed in Section 4. The results of the replanning experiments are detailed in Section 5, and the conclusion and future direction is discussed in Section 6.

2 Related Work

The Health Aware framework brings together two fields of study, that of prognostics and online replanning. Separately, there is an abundance of literature on the two subjects. However, within the field of prognostics, it is only recently that prognostics at the system-level has been discussed, and the area is still relatively new. Online replanning has been a topic of study for many years, including contingency planning, i.e. planning under faulty conditions. There is no actual verifiable publications that discuss online replanning under faulty conditions while accounting for system-level degradation due to nominal use (such as general *wear and tear*). Therefore, these two areas of study are briefly described in the disjoint manner with which they are presented in the literature.

2.1 System Level Prognostics

System Level Prognostics is a methodology for estimating the future performance of a system comprised of multiple interacting components that degrade simultaneously, with respect to a set of predefined safety and performance constraints. An event such as *End of Life* (EOL) occurs when one or more of these constraints is violated (Equation 4), and the time of that event less the current operational running time of the system is known as the RUL.

RUL must account for uncertainty, external influences, component damage accumulation, the interactions among these components, and the effects of multiple component degradation on system performance. Components interact with one another, so individual component degradation is no longer an isolated function of inputs and environment, but includes the complexities of feedback loops and interactions with other components (Darrah et al. 2021). System EOL represents an estimated point in time when the system performance drops below pre-specified thresholds. System failure is expressed as the union of component failure conditions in conjunction with the violation of one or more system level performance constraints. In general, prognostics methodologies can be divided into three basic categories (Sikorska, Hodkiewicz, and Ma 2011; Daigle, Saha, and Goebel 2012): model based, data driven, and hybrid approaches, discussed below.

1. **Model based:** Model-based prognostics approaches use domain knowledge about a system and its failure modes through the use of physics-based models. Model-based prognosis is generally divided into two sequential problems: a joint state-parameter estimation problem, in which, using the model, the health of a system or component is determined based on the observations; and a prediction problem, in which, using the model, the state-parameter distribution is simulated forward in time to compute *End of Life* (EOL) and subsequently *Remaining Useful Life* (RUL).

   These approaches use first principles physics of the system dynamics to develop parameterized degradation models of a component (Li and Lee 2005; Kacprzynski et al. 2004). The primary assumption is that accurate mathematical models are available, which can be the case for individual components, but typically not for complex *Cyber Physical Systems* (CPS). Model-based approaches have the advantage of being able to represent physical understanding into the monitor which also helps with explainability, and, choosing which features to monitor such that a functional mapping from the feature to the health of the asset is straight forward as well (Luo et al. 2003). Both state and degradation parameters need estimated first, and several techniques have been proposed for state and parameter estimation in nonlinear systems. These include least squares estimation (LSE) (Smyth et al. 2002), Extended Kalman Filters (EKF) (Corigliano and Mariani 2004; Mariani and Corigliano 2005), Uncented Kalman Filters (UKF) (Chatzi and Smyth 2009), and Particle Filters (Arulampalam et al. 2002; Jha, Dauphin-Tanguy, and Ould-Bouamama 2016).

2. **Data-driven:** Instead of a model of the system, sensor measurements are used to learn the damage accumulation function, and require run-to-failure training data (An, Kim, and Choi 2015). Data driven approaches are further divided into statistical methods (Si et al. 2011), (Tsui et al. 2015) or neural networks (Zhang et al. 2019), (Rezaeianjouybari and Shang 2020). Approaches such as Gaussian Processes (Liu et al. 2013), Support Vector Machines (SVM) (Chen et al. 2013), or Bayesian techniques (Youn and Wang 2012) fall under statistical methods. Recurrent Neural Networks (RNN) (Wu, Ding, and Huang 2020), (Dong, Li, and Sun 2017) or Convolutional Neural Network (CNN) (Li, Ding, and Sun 2018) are the most common deep learning architectures used.

   Deep learning approaches typically fall under RNNs or CNNs. RNNs are capable of accounting for temporal dependencies by connecting the output of layer n to layer $n - 1$. However, they suffer from the famous *vanishing gradient* problem, which is where the gradients tend towards zero during propagation through multiple partial derivative calculations during backpropogation. LSTM networks are typically used in favor of the original formulation of the RNN. LSTMs introduce the concept of *gates* and a *memory cell*. Bi-Directional LSTMs are a further enhancement to these types of networks that allow for future information to pass backwards to update the gradient during training (Darrah et al. 2022b).

3. **Hybrid approaches:** Hybrid approaches combine model-based and data-driven techniques to utilize the best of both approaches for accurate RUL estimations (Chao et al. 2021). Typically, model-based methods are used for state and parameter estimation, then a data-driven model (such as a neural network) is used for RUL prediction. Hybrid methods outperform either method individually (Liao and Köttig 2014), and there are numerous ways these methods can be constructed. This is the approach used in the Health Aware framework, and discussed in greater detail in Section 3.

2.2 Replanning

Online replanning assumes that an agent is initially following a predefined plan and due to new information must change the plan (Bonet and Geffner 2011). With an updated knowledge base, a new plan can be computed that would allow it to achieve a subset of its objectives (Komarnitsky and Shani 2016). In the domain of CPS, an inviolable objective will always be to maintain safe operations and minimize risk of failure. A vast majority of the algorithms to solve these types of problems are well known, but their implementation in these systems is not a trivial task. In a static environment, everything can be computed offline without worry of processor limitations or computational complexity. In a dynamic environment, a complex system such as a UAV must repeatedly make these calculations and trigger a replanning mode if it detects a failure along its current trajectory. Quiñones-Grueiro et al. (2021) developed a multi-objective cost function was implemented to assess risk, and a path search algorithm was utilized offline to generate a new trajectory and communicated to the UAV.

Krishnan and Manimala (2020) considered both the minimal path length and the minimal risk of collision for designing a path-planning algorithm suitable for real-time applications. They, however, do not consider the state of health of the UAV in the optimization problem. Real-time path planning for UAVs in the context of obstacles as dynamic traffic and geofences is explored by Chatterjee and Reza (2019). They use RRTs and build on top of the DAIDALUS software by NASA to implement maneuver computation for collision avoidance. To save computation time, the authors propose heuristics for early termination of tree generation for RRTs. Zammit and Van Kampen (2020) compared an A* graph search algorithm with limited look-ahead and intermediate goals, against RRTs for path planning. In their scenario of traversing through openings in walls laid out in a single principal direction, A* outperforms the more random exploration of RRT. This may be because the heuristics for intermediate goal points and the cost to goal are monotonic with respect to the optimal trajectory and its true cost.

Fickert et al. (2021) proposed a replanning technique based on A* as above but include multiple initial states in the search space to account for replanning time. Typically

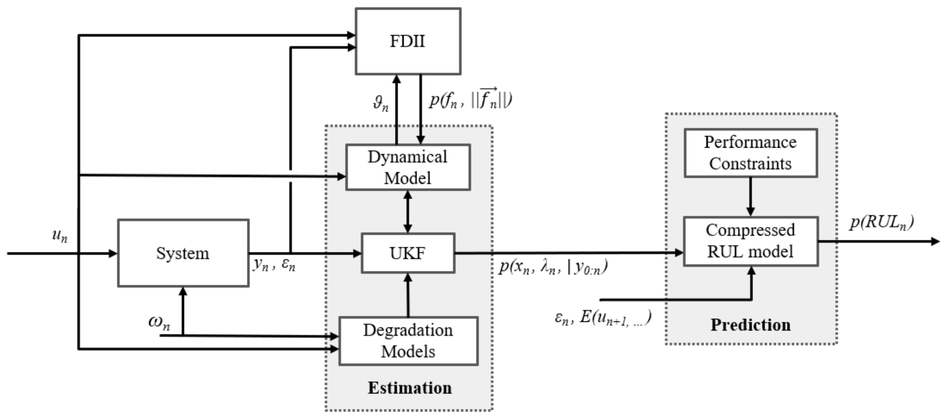

Figure 2: Online Architecture for Health-Awareness.

an initial state is manually selected during online replanning that attempts to account for replanning time. However, their approach does not account for faults or degradation. Ure et al. (2013) proposed the first and one of only a few works that discuss Health-Aware planning. They assigned a capability score based on the health of the system and use a Markov Decision Process to solve the replanning problem. Their approach, however, does not account for abrupt faults that occur mid-flight. Kita et al. (2023) discussed uncertainty with travel times for an office delivery robot and proposed an adaptive parameter update method to better estimate the probability distribution of the travel time. Like numerous previous works, they did not account for *wear-and-tear* degradation or abrupt faults. Balaban et al. (2013) developed a mobile robotics test platform for prognostics based decision making studies. This work is most closely related to the Health-Aware framework presented here. However, the authors did not implement a deep learning model for prognostics and did not have multiple degrading components in the system.

3 Methodology

The Health-Aware Framework is comprised of four key elements: (1) a state-space dynamical system model; (2) individual component degradation models; (3) a system-level Remaining Useful Life (RUL) model; and (4) a replanning agent. The first two components were defined in the problem statement in Section 1. The system-level RUL model is defined as

Definition 3 (System Level Remaining Useful Life Model)

$$t_r = \mathcal{F}(\mathbf{x}_n, \gamma_n, \epsilon_n, E(u_{n+1}, ...)), \tag{6}$$

where $\mathcal{F}$ represents a Bi-Directional Long-Short Term Memory (Bi-LSTM) network with inputs consisting of the system state, $\mathbf{x}_n$, degradation parameters, γ_n, control and position errors, ϵ_n, and expected future usage, $E(u_{n+1}, ...)$. Several publications discuss various architectures for RUL estimation, all which achieve relatively decent accuracy. We have chosen the Bi-LSTM architecture due to success in previous experiments and its simplicity (only 2 layers).

3.1 Replanning

There are a number of approaches to solution generation such as dynamic programming, evolutionary programming, mixed integer linear programming, or graph-based methods such as A*. The replanning function is framed as an orienteering problem (Golden, Levy, and Vohra 1987) and defined by

Definition 4 (Replanning Function)

$$\mathbf{G} = (\mathcal{W}, \mathbf{D}, \mathbf{H}, \mathbf{R}),$$
$$\mathcal{MP}' = \mathcal{G}(\mathbf{G}, w_f, T_s, t_r, w_g) \tag{7}$$

where $\mathbf{G}$ is a graph with waypoints, $w \in \mathcal{W}$, and edges are the travel time between any two pair of waypoints in a distance matrix $\mathbf{D}$, the travel time from a given waypoint to the goal location is the heuristic stored in a vector $\mathbf{H}$, and the reward value for each waypoint is stored in a reward vector $\mathbf{R}$. $\mathcal{G}$ is a function that takes as input the graph, $\mathbf{G}$, the starting wayoint after the failure, w_f, the stopping time at each waypoint, T_s, the remaining flight time, t_r, and the goal location, w_g, which is the same as the start location. A new mission plan, $\mathcal{MP}'$ is generated with a new trajectory. Once a tentative solution is generated, the algorithm checks to see if it is feasible (i.e. does exceed the remaining flight time), and then prioritizes by total reward, r_t. If no feasible trajectory is found, the algorithm returns $None$, otherwise, it returns the new trajectory, $\mathcal{W}^{\mathcal{T}'}$. The waypoints on the original trajectory have a reward value of 5, and all other waypoints have a value of either 0 or 1. Rewards are used to prioritize the selection of original waypoints over alternate waypoints where possible. Flight-time calculations are captured with g. The algorithm is provided below in Algorithm 1.

The algorithm takes as input the graph, $\mathbf{G}$, the starting waypoint after fault has been identified, w_f, a vector of stopping times at each waypoint, T_s, the time remaining until failure, t_s, and the goal waypoint, w_g. In Line 2, the visited waypoints, V, is assigned an empty set, and in Line 3, the priority queue, PQ, is assigned a tuple consisting of the reward for the initial waypoint, R^{w_f}, the stopping time at the current waypoint, $T_s^{w_f}$, the waypoint itself, w_f, and an empty set for the path. In Line 5, the tuple is popped and represents the total reward, r_t, the total path cost, g, the current waypoint, w_i, and the current trajectory, $\mathcal{W}^{\mathcal{T}'}$. Lines 6 and 7 are the stopping condition that will return the new trajectory. In lines 9 and 10, the current waypoint is added to the visited set and to the new trajectory. If the path cost plus

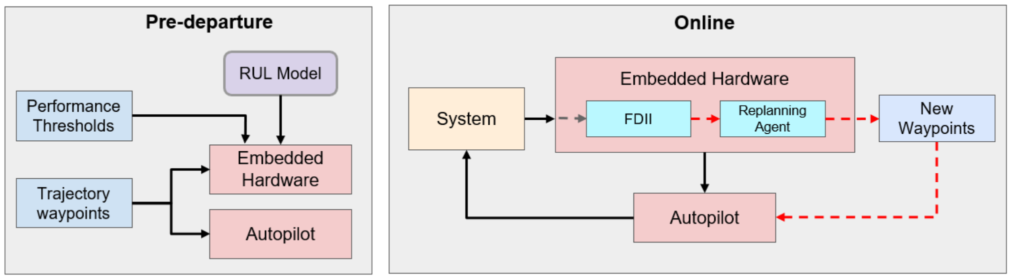

Figure 3: Replanning framework. Offline mission plan generation (L); online monitoring and replanning (R)

the estimated distance to the goal waypoint is less than the time remaining (line 11), then the neighbors of the current waypoint, w_i^N, are evaluated. The path cost is updated to include the travel time between the current waypoint and its neighbor and the stopping time at the neighbor (line 14). In Line 15 the reward is updated, then Line 16 does the same check as Line 11 for the neighbor, and adds it to the priority queue in Line 17.

Algorithm 1: Constrained Routing with Rewards

```
 1: procedure ROUTING($\mathbf{G}, w_f, T_s, t_r, w_g$)
 2:     $V \leftarrow \{\}$
 3:     $PQ \leftarrow (R^{w_f}, T_s^{w_f}, w_f, \{\})$
 4:     while $PQ \neq \varnothing$ do
 5:         $r_t, g, w_i, \mathcal{W}^{\mathcal{T}'} \leftarrow$ POP($PQ$)
 6:         if $w_i = w_g$ then
 7:             return $\mathcal{W}^{\mathcal{T}'} + w_i$
 8:         if $w_i \notin V$ then
 9:             $V \leftarrow V + w_i$
10:             $\mathcal{W}^{\mathcal{T}'} \leftarrow \mathcal{W}^{\mathcal{T}'} + w_i$
11:             if $g + H_{w_i} < t_r$ then
12:                 for $n \in w_i^N$ do
13:                     if $n \notin V$ then
14:                         $g \leftarrow g + D_{w_i,n} + T_s^n$
15:                         $r_t \leftarrow r_t + T_r^n$
16:                         if $g + H_n < t_r$ then
17:                             PUSH($PQ, (r_t, g, n, \mathcal{W}^{\mathcal{T}'})$)
18:     return None
```

3.2 Online Architecture for Health Awareness

The online Health-Aware architecture is depicted in Figure 2, consisting of an estimation scheme to estimate the system state, $\mathbf{x}$, and degradation parameters, γ; a prediction scheme to predict the remaining flight time, t_r, (RUL), and a fault detection, identification, and isolation (FDII) scheme to detect faults and their magnitude, f and $||\overrightarrow{\mathbf{f_n}}||$, which are used to recompute state and degradation parameters of the system. The inputs, $\mathbf{u}$, consist of the velocity profile, wind conditions, and motor control signals, and are fed to the system, the dynamical model of the system, the degradation models of the components within the system, and the FDII monitor. The uncertainties and noise, ω, are also input to the system and degradation models as Gaussian distributions. The FDII block also receives the system parameters, ϑ, and outputs fault information back to the dynamical system model to update the parameters of the system if a fault is detected. There is a two-way link between the dynamical system model and the UKF, as the UKF estimates the system parameters and updates them as necessary as well. This is a central part of the Health-Aware architecture. As these estimates are continuously calculated online, the system state and degradation parameters are passed to the RUL model in the prediction step. The RUL model also receives a vector of errors, ϵ, which are comprised of position errors, arrival time errors, and control errors. In addition, the RUL model takes the future expected load, $E(u_{n+1}...)$, and the performance constraints, which is a set of Boolean statements on both system-level and component-level parameters, such as cumulative position error (system-level), or battery state of charge (component-level). Together, the composition of these pieces form the online architecture for Health-Awareness.

3.3 Online Replanning

The online replanning framework is presented in Figure 3, consisting of a pre-departure component (L), and an online component (R). During pre-departure, the mission plan, $\mathcal{MP}$ is generated which loads the initial trajectory and performance thresholds onto the embedded hardware and autopilot. The RUL model is also loaded onto the embedded hardware at this time. During the online phase, the Health-Awareness architecture performs monitoring, estimation, and prediction tasks. If at anytime the remaining flight time, t_r, is less than the original trajectory flight time, the replaning agent is activated and computes a new mission plan. The autopilot receives the new waypoints and then executes the updated trajectory.

Together, the architecture for health awareness and online replanning framework are used for online health-aware replanning. This methodology is demonstrated in the following section.

4 Experiments

We demonstrate the benefits of the Health-Aware replanning framework by showing the effects of replanning for a UAV while undergoing usage-based degradation and an abrupt fault. We show what happens when a UAV has access to system-level SOH information and when it does not. To satisfy the first part, we select a UAV that has completed between 40 and 60 flights, having flown at least 100km, with the relevent degradation parameters shown in Table 1.

The abrupt fault we implement is the loss of a battery cell, and for a LiPo 6S battery this results in the total charge capacitance being reduced by $\frac{1}{6}$. An exemplary trajectory is

Parameter	description	Initial	Actual
Q	Battery Capacitance	22.0 aH	20.95 aH
R_0	Battery resistance	.0011 Ω	.0043 Ω
R_{m1}	Motor 1 resistance	.2371 Ω	.2674 Ω
R_{m2}	Motor 2 resistance	.2370 Ω	.2709 Ω
R_{m3}	Motor 3 resistance	.2371 Ω	.2799 Ω
R_{m4}	Motor 4 resistance	.2372 Ω	.2703 Ω
R_{m5}	Motor 5 resistance	.2369 Ω	.2727 Ω
R_{m6}	Motor 6 resistance	.2371 Ω	.2671 Ω
R_{m7}	Motor 7 resistance	.2369 Ω	.2806 Ω
R_{m8}	Motor 8 resistance	.2374 Ω	.2684 Ω

Table 1: System degradation parameters

shown in Figure 1, with the AO, trajectory, and waypoints depicted for a successful flight in the top left. Alternate waypoints are depicted that could be considered during replanning. The bottom left depicts the battery state of charge and output voltage for a nominal flight. The right half of the figure shows what happens when a fault occurs with no replanning at all, where the UAV ultimately fails mid-flight, and the effects of the abrupt loss of a battery cell are evident.

The remaining useful life calculation is the primary input to the replanner. Current industry standards use the battery discharge rate to determine max flight time, and by this method the remaining useful life can be calculated by

Definition 5 (Discharge-based Remaining Useful Life)

$$t_r = t_m - t_m \cdot \frac{(\Sigma i_c)}{Q}, \tag{8}$$

where t_r is the time remaining, t_m is the max flight time (typically provided by the manufacturer), Σi_c is the cumulative current consumed, and Q is the total charge capacitance. Under ideal conditions ignoring degradation, this is a perfectly acceptable method of calculating remaining flight time. However, degradation is a natural phenomenon that occurs in all complex systems and therefore must be accounted for. A functional mapping between system operation and system state of health must be incorporated into a replanning agent to provide more accurate flight time estimates. This is shown in the right hand side of Figure 2.

4.1 RUL Model

The details of the RUL model development process can be found in previous publications (Darrah et al. 2022a,b), however the dataset is briefly discussed for context. This is a run to failure dataset, meaning there are multiple UAVs that execute trajectories assigned at random until a safety or performance threshold is violated. They fly under varying wind conditions with stochastic degradation profiles in 9 components, consisting of 8 motors with one degrading parameter (internal resistance) and 1 battery with 2 degradation parameters (total charge capacitance and internal resistance). There are 184 UAVs with a total of 17,629 flights. Of this, 88 UAVs were selected for the experiment, 66 for training, 11 for testing, and 11 for validation. The telemetry data consists of 80 features sampled at 1 hz for a total of 19,876,367 records. Features sampled include battery data such as current, voltage, and estimated state of charge; motor data such as current and RPM; errors; and body data such as position, velocity, acceleration, orientation, and angular velocity.

Parameter	description	Value
L	layers	2
c_{L1}	layer 1 cells	80
c_{L2}	layer 2 cells	40
d	dropout rate	20%
d_r	recurrent dropout	25%
l_1	l_1 regularization	$1e^5$
l_2	l_2 regularization	$1e^5$
b	batch size	64

Table 2: Bi-LSTM parameters.

When accounting for the entire test set, the model has an RMSE of 8.3%, expressed as a percentage of predicted minutes until failure. When discarding the outlier (Figure 4, middle-bottom row), the model RMSE is 2.6%. The results of the RUL model are shown in Figure 4 for 10 test UAVs. From these results we can be confident in accurate RUL estimates. The model architecture is a 2-layer Bi-LSTM with the following parameters shown in Table 2.

5 Results

The results of the replanning experiment that demonstrates the Health-Aware framework are shown in Figure 5. The planner on the left receives a poor quality estimate from the naive remaining flight time calculation and comes up with a plan that is too optimistic and overestimates the RUL. Due to this, it attempts to reach two waypoints from the original trajectory, but ultimately fails before finishing the flight. It would have received a total reward of 24 points, as opposed to 16 points received by the Health-Aware replanner. However, the Health-Aware replanner had a more accurate RUL estimate that accounted for system degradation and estimated a shorter remaining flight time. It opted to not attempt to reach the the last waypoints (circled in red) and chose alternate waypoints instead due to the fact reaching the goal is an inviolable constraint.

6 Conclusion

In this work, we introduce novel *Health-Aware* replanning framework that incorporates system-level SOH information. By using a prognostics model to during online replanning, we observe better flight time estimates and as a result a safer trajectory is generated online that allows the UAV to reach the goal location without failure.

This was a simple demonstration to show the viability of the approach but there is plenty of research opportunities to extend this work. Some initial areas to explore would be to predict travel time and power consumption on a segment-by-segment basis as it relates to current wind conditions. Edges must then be dynamically computed as opposed to statically defined in a matrix. Additionally, only one fault was considered in this work, the loss of a single battery cell. Other faults could be implemented such as the loss of a motor, an electronic speed controller (ESC) switching fault, or a parasitic load fault, to name a few. Also, only one trajectory was used in this demonstration. There are currently 53 different trajectories available for selection in the simulation,

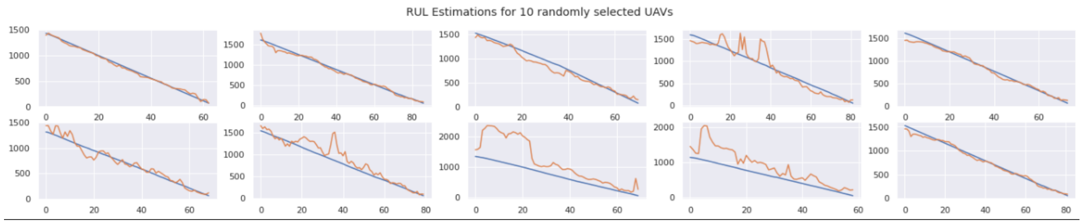

Figure 4: RUL Estimation results for 10 UAVs in the test dataset.

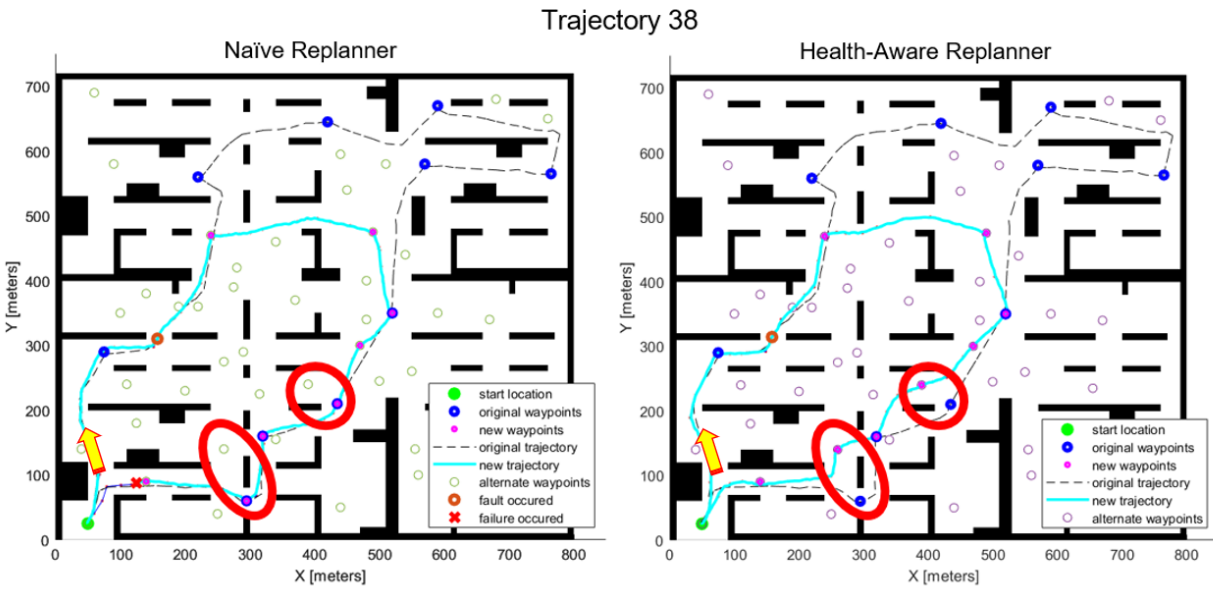

Figure 5: Naive replanner (L), with a 1,702 meter long trajectory; Health-Aware replanner (R), with a 1,608 meter long trajectory. Red circles show different waypoints selected. The UAV travels clockwise in this particular trajectory.

and other trajectories should be used as well. Other UAVs with different levels of usage and degradation should also be used, here, only one was used.

7 Acknowledgement

This work is supported by NASA Fellowship OSTEM 20-154.

References

Ahmed, I.; Quinones-Grueiro, M.; and Biswas, G. 2023. Adaptive fault-tolerant control of octo-rotor UAV under motor faults in adverse wind conditions.

An, D.; Kim, N. H.; and Choi, J.-H. 2015. Practical options for selecting data-driven or physics-based prognostics algorithms with reviews. *Reliability Engineering & System Safety*, 133: 223–236.

Arulampalam, M. S.; Maskell, S.; Gordon, N.; and Clapp, T. 2002. A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking. *Signal Processing, IEEE Transactions on 50.2, (2002) 174-188*.

Balaban, E.; Narasimhan, S.; Daigle, M.; Roychoudhury, I.; Sweet, A.; Bond, C.; and Gorospe, G. 2013. Development of a Mobile Robot Test Platform and Methods for Validation of Prognostics-Enabled Decision Making Algorithms. *International Journal of Prognostics and Health Management*, 4.

Bonet, B.; and Geffner, H. 2011. Planning under Partial Observability by Classical Replanning: Theory and Experiments. In *IJCAI*.

Bregon, A.; Daigle, M.; Roychoudhury, I.; Biswas, G.; Koutsoukos, X.; and Pulido, B. 2014. An event-based distributed diagnosis framework using structural model decomposition. *Artificial Intelligence*, 210: 1–35.

Chao, M. A.; Kulkarni, C.; Goebel, K.; and Fink, O. 2021. Aircraft Engine Run-to-Failure Dataset under Real Flight Conditions for Prognostics and Diagnostics. *Data*.

Chatterjee, A.; and Reza, H. 2019. Path Planning Algorithm to Enable Low Altitude Delivery Drones at the City Scale. In *2019 International Conference on Computational Science and Computational Intelligence (CSCI)*, 750–753. IEEE.

Chatzi, E. N.; and Smyth, A. W. 2009. The unscented Kalman filter and particle filter methods for nonlinear structural system identification with non-collocated heterogeneous sensing. *Structural control and health monitoring 16, no. 1 (2009) 99-123*.

Chen, X.; Shen, Z.; He, Z.; Sun, C.; and Liu, Z. 2013. Remaining life prognostics of rolling bearing based on relative features and multivariable support vector machine. *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science*, 227.

Corigliano, A.; and Mariani, S. 2004. Parameter identification in explicit structural dynamics: performance of the extended Kalman filter. *Computer Methods in Applied Mechanics and Engineering 193, no. 36 (2004) 3807-3835*.

Daigle, M.; Saha, B.; and Goebel, K. 2012. A comparison of filter-based approaches for model-based prognostics.

Darrah, T.; Biswas, G.; Frank, J.; Quinones-Grueiro, M.; and Teubert, C. 2022a. A data-centric approach to the study of system-level prognostics for cyber physical systems: application to safe UAV operations. *Journal of Surveillance, Security and Safety*, 3(2).

Darrah, T.; Lovberg, A.; Frank, J.; Biswas, G.; and Quinones-Gruiero, M. 2022b. Developing Deep Learning Models for System Remaining Useful Life Predictions: Application to Aircraft Engines. In *Annual Conference of the PHM Society*.

Darrah, T.; Quiñones-Grueiro, M.; Biswas, G.; and Kulkarni, C. 2021. Prognostics Based Decision Making for Safe and Optimal UAV Operations. In *AIAA Scitech 2021 Forum*.

Dong, D.; Li, X.-Y.; and Sun, F.-Q. 2017. Life prediction of jet engines based on LSTM-recurrent neural networks. In *2017 Prognostics and System Health Management Conference (PHM-Harbin)*, 1–6.

Fickert, M.; Gavran, I.; Fedotov, I.; Hoffmann, J.; Majumdar, R.; and Ruml, W. 2021. Choosing the Initial State for Online Replanning. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(14): 12311–12319.

Golden, B. L.; Levy, L.; and Vohra, R. 1987. The orienteering problem. *Naval Research Logistics (NRL)*, 34(3): 307–318.

Jha, M. S.; Dauphin-Tanguy, G.; and Ould-Bouamama, B. 2016. Particle filter based hybrid prognostics for health monitoring of uncertain systems in bond graph framework. *Mechanical Systems and Signal Processing 75 (2016): 301-329*.

Kacprzynski, G. J.; Sarlashkar, A.; Roemer, M. J.; Hess, A.; and Hardman, W. 2004. Predicting remaining life by fusing the physics of failure modeling with diagnostics. *JOm 56, no. 3, (2004): 29-35.*

Kita, A.; Suenari, N.; Okada, M.; and Taniguchi, T. 2023. Online Re-Planning and Adaptive Parameter Update for Multi-Agent Path Finding with Stochastic Travel Times.

Komarnitsky, R.; and Shani, G. 2016. Computing Contingent Plans Using Online Replanning. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*.

Krishnan, P. S.; and Manimala, K. 2020. Implementation of optimized dynamic trajectory modification algorithm to avoid obstacles for secure navigation of UAV. *Applied Soft Computing*, 90: 106168.

Li, C. J.; and Lee, H. 2005. Gear fatigue crack prognosis using embedded model, gear dynamic model and fracture mechanics. *Mechanical systems and signal processing 19, no. 4, (2005): 836-846.*

Li, X.; Ding, Q.; and Sun, J.-Q. 2018. Remaining useful life estimation in prognostics using deep convolution neural networks. *Reliability Engineering & System Safety*, 172.

Liao, L.; and Köttig, F. 2014. Review of Hybrid Prognostics Approaches for Remaining Useful Life Prediction of Engineered Systems, and an Application to Battery Life Prediction. *IEEE Transactions on Reliability*, 63(1): 191–207.

Liu, D.; Pang, J.; Zhou, J.; Peng, Y.; and Pecht, M. 2013. Prognostics for state of health estimation of lithium-ion batteries based on combination Gaussian process functional regression. *Microelectronics Reliability*, 53(6): 832–839.

Luo, J.; Namburu, M.; Pattipati, K.; Qiao, L.; Kawamoto, M.; and Chigusa, S. 2003. Model-based prognostic techniques [maintenance applications]. In *Proceedings AUTOTESTCON 2003. IEEE Systems Readiness Technology Conference.*, 330–340. Ieee.

Mariani, S.; and Corigliano, A. 2005. Impact induced composite delamination: state and parameter identification via joint and dual extended Kalman filters. *Computer methods in applied mechanics and engineering 194.50 (2005) 5242-5272.*

Quiñones-Grueiro, M.; Biswas, G.; Ahmed, I.; Darrah, T.; and Kulkarni, C. 2021. Online decision making and path planning framework for safe operation of unmanned aerial vehicles in urban scenarios. *Aerospace Journal (to appear)*.

Rezaeianjouybari, B.; and Shang, Y. 2020. Deep learning for prognostics and health management: State of the art, challenges, and opportunities. *Measurement*, 163.

Si, X.-S.; Wang, W.; Hu, C.-H.; and Zhou, D.-H. 2011. Remaining useful life estimation–A review on the statistical data driven approaches. *European Journal of Operational Research, 213(1), (2011): 1-14.*

Sikorska, J. Z.; Hodkiewicz, M.; and Ma, L. 2011. Prognostic modelling options for remaining useful life estimation by industry. *Mechanical Systems and Signal Processing 25, no. 5 (2011): 1803-1836.*

Smyth, A. W.; Masri, S. F.; Kosmatopoulos, E. B.; Chassiakos, A. G.; and Caughey, T. K. 2002. Development of adaptive modeling techniques for non-linear hysteretic systems. *nternational Journal of Non-Linear Mechanics, 37(8), (2002) 1435-1451.*

Tsui, K. L.; Chen, N.; Zhou, Q.; Hai, Y.; and Wang, W. 2015. Prognostics and Health Management: A Review on Data Driven Approaches. *Mathematical Problems in Engineering*.

Ure, N. K.; Chowdhary, G.; How, J. P.; Vavrina, M. A.; and Vian, J. 2013. Health Aware Planning under uncertainty for UAV missions with heterogeneous teams. In *2013 European Control Conference (ECC)*, 3312–3319.

Wu, Q.; Ding, K.; and Huang, B. 2020. Approach for fault prognosis using recurrent neural network. *Journal of Intelligent Manufacturing*, 31.

Youn, B.; and Wang, P. 2012. *A Generic Bayesian Framework for Real-Time Prognostics and Health Management (PHM)*.

Zammit, C.; and Van Kampen, E.-J. 2020. Comparison of A* and RRT in real–time 3D path planning of UAVs. In *AIAA Scitech 2020 Forum*, 0861.

Zhang, L.; Lin, J.; Liu, B.; Zhang, Z.; Yan, X.; and Wei, M. 2019. A Review on Deep Learning Applications in Prognostics and Health Management. *IEEE Access*, 7.