Assured Contingency Landing Management for Advanced Air Mobility

Joseph Kim[1], Prashin Sharma[2], Ella Atkins[3], Natasha Neogi[4], Evan Dill[5], and Steven Young[6]

***Abstract*—Advanced Air Mobility (AAM) is quickly developing as a new air transportation system with increasing autonomy levels to support low-cost on-demand passenger and package transport. AAM must operate safely despite the potential to encounter hazards and experience anomalies and failures in-flight. It becomes especially important to have systematic auto-mitigation strategies to perform safe contingency actions in AAM flight operations, as pilots have limited Situational Awareness (SA) and limited time to make decisions when encountering failures/anomalies in complex airspace and terrain. This paper presents Assured Contingency Landing Management (ACLM) with an online landing strategy selection capability to decide between the following three options when a contingency landing is required: (1) Return-to-launch landing, (2) Land immediately at a nearby clear but unprepared site, and (3) Land at a prepared landing site within the aircraft's reachable footprint. Our presented algorithm shows a real-time auto-mitigation loop with multiple threads that run simultaneously to check controllability, reachability, and intermediate decisions to hold/ loiter or continue the flight plan as the landing strategy solution is being computed. A simulation case study demonstrates ACLM with a focus on safety-critical propulsion and battery system degradation scenarios to illustrate ACLM functionality.**

***Index Terms*—advanced air mobility, emergency planning, controllability, approximate footprint, reachability**

I. Introduction

Advanced Air Mobility (AAM) has the potential to revolutionize aviation with flexible on-demand passenger and package transportation. AAM will require increasing autonomy levels to support safe low-altitude mixed-use airspace operations especially near urban centers. AAM will meet new transport needs within urban areas and between populated regions to improve access and reduce transportation time compared to existing air or ground-based systems [1]. However, full realization of AAM will require consistent long-term development efforts [2]. AAM requires auto-mitigation strategies to perform safe contingency actions in emergencies. This is especially important for high-density low-altitude airspace where pilots have limited Situational Awareness (SA) and limited time to make decisions. Automated Assured Contingency Landing Management (ACLM) will enable a distressed AAM flight to determine the safest landing option from a fusion of pre-analyzed and real-time flight data to minimize reaction time and ensure safe landing opportunities are not lost due to decision delay.

This paper proposes a single-aircraft ACLM solution. We present sub-component logic flow and mathematical derivation to achieve Assurance Efficacy (AE) of different auto-mitigation responses. For this work, ACLM functions are derived for multicopter systems with more than four rotors so that the vehicle can maintain control authority even after losing one or two motors. Specific degradation models of safety-critical propulsion and battery modules are presented and used in case study simulations to demonstrate how different failure cases impact safe landing decisions. This paper is first to introduce our ACLM capability.

The remaining structure of this paper is organized as follows. Section II discusses related work in emergency flight planning and decision making. Section III summarizes multicopter dynamics, controllability, and reachability. Section IV defines ACLM requirements and states assumptions made in this work. An ACLM activation watchdog is defined to assure controllability and reachability constraints are satisfied. A strategy is proposed for landing site selection that capitalizes on pre-flight contingency planning to minimize or eliminate the need for online flight planning. Section V describes case study results for a multicopter system experiencing rotor failures, and Section VI concludes the paper.

II. Literature Review

AAM platforms must have fault tolerance capabilities to safely complete their missions. A fault-tolerant control scheme is proposed for multicopters in [3], utilizing a reduced controllability index to compute a constrained control allocation scheme. A baseline Nonlinear Dynamic Inversion (NDI) controller is presented and used to safely land the faulty multicopter. A data fusion architecture for tolerating sensor and software faults is presented in [4]. The proposed architecture provides detection and recovery services adapted to multi-sensor perception systems.

AAM platforms must be capable of safely reaching the selected emergency landing site. This can be formulated as the reachability problem in physical space. Hamilton-Jacobi reachability analysis [5], [6] is a verification method that computes a set of states from which the system can be driven to a target set while satisfying time-varying state constraints at all times. Ref. [7] utilizes Hamilton Jacobi Bellman (HJB) reachability analysis to determine the feasible landing region. Reachability in [8] is defined in terms of feasible post-failure trim states and transitions between them. The authors then

[1]Ph.D. Candidate, Robotics Institute, University of Michigan, USA
[2]Ph.D. Candidate, Robotics Institute, University of Michigan, USA
[3]Professor, Aerospace Engineering Department and Robotics Institute, University of Michigan, AIAA Fellow
[4]Research Engineer, NASA, AIAA Associate Fellow
[5]Research Engineer, NASA
[6]Research Engineer, NASA, AIAA Associate Fellow

use a database of feasible trim states and transitions to build landing plans. An approximate footprint is proposed in [9] for a fixed-wing aerial vehicle experiencing total loss of thrust thus constrained to a descending glide path. A three-dimensional maximum range Dubins path is proposed in [10]. This maximum range also defines an approximate reachable footprint for fixed-wing aircraft.

AAM also requires contingency planning to handle urgent landing or emergency scenarios. Procedures to manage lost link situations are proposed in [11] for remotely piloted aircraft. These procedures rely on historical traffic volume data to develop appropriate reactions; defined procedures are assessed in human-in-the-loop simulations to determine operator workload. A decision system is proposed in [12] to monitor a suite of metrics to assess the safety and feasibility of the mission and categorize flight states as Nominal, Off-Nominal, Alternate Land, and Land Now in a finite state machine. Changes in the monitored metrics serve as transitions in the state machine and trigger replanning. Our work builds on this previous work by applying controllability and reachability analysis to ACLM decisions cast in a procedure that exploits preflight planning to minimize online computations and associated response delay.

III. Preliminaries

This section reviews multicopter dynamics, multicopter controllability analysis with rotor failures, and classical reachability analysis. All are integral to ACLM.

A. Multicopter Dynamics

A multicopter with more than four motors provides redundant actuation to support continued flight following propulsion unit failure(s). Although this paper specifically models multicopters, ACLM logic can be applied to other air vehicle configurations. Multicopter dynamics are given by [13]:

$$\ddot{x} = (\cos\phi\sin\theta\cos\psi + \sin\psi\sin\phi) * \frac{u_f}{m} \tag{1a}$$

$$\ddot{y} = (\cos\phi\sin\theta\cos\psi - \sin\psi\sin\phi) * \frac{u_f}{m} \tag{1b}$$

$$\ddot{z} = (\cos\phi\cos\theta) * \frac{u_f}{m} - g \tag{1c}$$

$$\dot{p} = \frac{I_{yy} - I_{zz}}{I_{xx}} qr - \frac{J_r}{I_{xx}} q\Omega + \frac{1}{I_{xx}} T_\phi \tag{1d}$$

$$\dot{q} = \frac{I_{zz} - I_{xx}}{I_{yy}} pr + \frac{J_r}{I_{yy}} p\Omega + \frac{1}{I_{yy}} T_\theta \tag{1e}$$

$$\dot{r} = \frac{I_{xx} - I_{yy}}{I_{zz}} pq + \frac{1}{I_{zz}} T_\psi \tag{1f}$$

where x, y, z are multicopter center of mass position coordinates in a ground-fixed inertial frame, m is vehicle mass, and I_{xx}, I_{yy}, I_{zz} are the moments of inertia around body frame roll (x), pitch (y) and yaw (z) axes. ϕ, θ, ψ are the Euler roll, pitch, and yaw angles, p, q, r are angular rates in the body frame, and T_ϕ, T_θ, T_ψ are torques generated in the body frame. J_r is the moment of inertia of a single propeller, and Ω is propeller angular velocity with counter-clockwise (top-down view) positive by convention.

The relation between angular velocity in body frame and inertial frame is written as follows:

$$\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} 1 & \sin\phi\tan\theta & \cos\phi\tan\theta \\ 0 & \cos\phi & -\sin\phi \\ 0 & \frac{\sin\phi}{\cos\theta} & \frac{\cos\phi}{\cos\theta} \end{bmatrix} \begin{bmatrix} p \\ q \\ r \end{bmatrix}. \tag{2}$$

The mapping from the *n* rotor thrust forces f_i to total thrust (T) and torque (T_ϕ, T_θ, T_ψ) is:

$$\begin{bmatrix} T \\ T_\phi \\ T_\theta \\ T_\psi \end{bmatrix} = B * \begin{bmatrix} f_1 \\ \vdots \\ f_n \end{bmatrix} \tag{3}$$

where the control effectiveness matrix is given by:

$$B = \begin{bmatrix} \eta_1 & \eta_2 & \dots & \eta_n \\ l_1\eta_1\sin\gamma_1 & l_2\eta_2\sin\gamma_2 & \dots & l_n\eta_n\sin\gamma_n \\ -l_1\eta_1\cos\gamma_1 & -l_2\eta_2\cos\gamma_2 & \dots & -l_n\eta_n\cos\gamma_n \\ \eta_1 * d_1 & \eta_2 * d_2 & \dots & \eta_n * d_n \end{bmatrix} \tag{4}$$

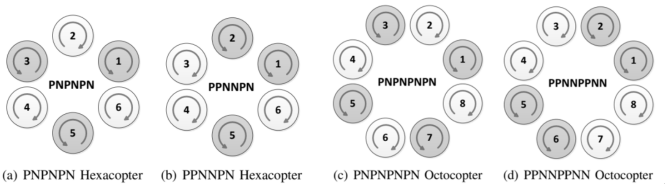
Above, l_i denotes multicopter arm length, η_i is propulsion unit health in the range $[0, 1]$ and γ_i indicates the angle from the x-axis to the i^{th} arm. d_i denotes the direction of propeller rotation.

B. Multicopter Controllability with Rotor Failure(s)

A multicopter is a nonlinear dynamic system with positive control inputs. That is, the rotors provide unidirectionally-constrained thrust. According to classical controllability theory for linear systems [14], [15], the Kalman rank test is not a sufficient condition to test the controllability of such a system with bounded control inputs. Instead, positive controllability was defined by Brammer [16] to provide necessary and sufficient conditions for analyzing controllability of a system with constrained inputs. Du et al [17] and Saied et al [13] proposed methods to evaluate multicopter controllability, defining Available Control Authority Index (ACAI) [18] and Small time Local Controllability (STLC), respectively. For ACLM multicopter analyses, we check the health of the rotors using Fault Detection and Identification (FDI) and Fault-Tolerant control as in [19], [20]. We follow the derived methods in [17] to test positive controllability of a generalized multicopter system. This process will allow us to check the controllability conditions in [14] in real-time with the proposed linearized state-space method.

The positive controllability test in [17] was derived with respect to a hover condition, suggesting a full controllability test on attitude and altitude control for *some combinations* of rotor failures. It is important to note that *"some combinations"* is subject to the multicopter configuration; one configuration can successfully maintain its controllability, while another cannot maintain it although both are subject to the same combination of rotor failures. Schneider, et al. investigated how the number and arrangement of rotors affect multicopter fault-tolerance [19]. Fig. 1 shows example hexacopter and

octocopter configurations from Schneider's paper. The fault-tolerant control strategy in [18] achieved positive position controllability at the cost of uncontrollable yaw. Although removing yaw control authority is undesirable, retaining position control can enable a multicopter to reach a designated landing site.



(a) PNPNPN Hexacopter (b) PPNNPN Hexacopter (c) PNPNPNPN Octocopter (d) PPNNPPNN Octocopter

Fig. 1: Multicopter configurations in fault-tolerant positive controllability analysis with hexacopter and octocopter configurations [19]. The proposed positive controllability test can also be applied in other multicopter configurations.

After linearizing the multicopter dynamics about a steady hover condition, the positive controllability theorem from [16] can be directly applied [17]. The necessary and sufficient conditions for positive controllability of a linearized multicopter system are given by [16], [17]:

1) C(A, B) = $\begin{bmatrix} B & A*B & \dots & A^n*B \end{bmatrix}$
2) There is no real eigenvector ν of A^T satisfying $\nu^T B u \leq 0$ for all $u \in U$ from [16]
   Equivalently, **ACAI** $\rho(G, \delta\Omega) > 0$ from [17]

Here ACAI stands for **Available Control Authority Index** [17]. The greater the ACAI, the more control authority is available. If $ACAI < 0$, the vehicle is not controllable.

After applying the controllability test procedure from [17], lookup tables can be generated offline to define ACAI for all combinations of rotor failures given a specific multicopter configuration. These tables can quickly indicate cases in which a multicopter must perform immediate flight termination, e.g., with parachute-assist. If a failure has occurred and the multicopter is still controllable, the landing strategy selection procedure is executed to define a safe ACLM solution. Figures 2-4 show examples of octocopter configurations and their positive controllability results with up to three rotor failures.

C. Reachability of energy-constrained multicopter dynamics

Reachability of the destination is determined based on an estimate of Battery End of Discharge (EOD) time. Algorithm (1) describes the function to compute landing site reachability for the ACLM solution. Battery EOD time is determined based on the maximum power usage of the multicopter for level flight, and is a conservative estimate. Flight plan duration is calculated by subtracting a safety margin from the EOD to provide a buffer.

IV. Assured Contingency Landing Management (ACLM)

A. ACLM Functions and Assumptions

ACLM is formally defined to specify a common framework and data structure for emergency landing management. The four main ACLM functions are:

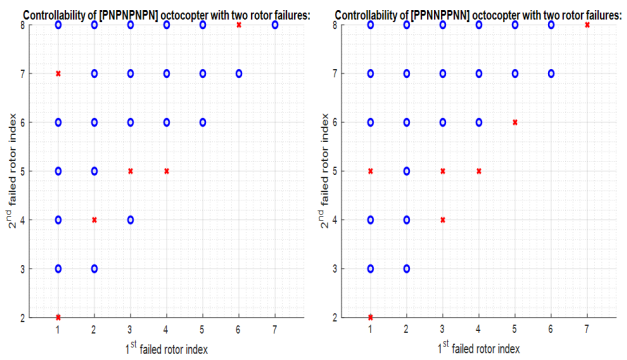


Fig. 2: Controllability of "PNPNPNPN" (left) and "PPNNPPNN" (right) octocopters, where P and N indicate clockwise (CW) and counter-clockwise (CCW) directions of rotation of a related DC motor, subject to two rotor failures. A blue circle denotes a controllable system even though the indicated rotors have failed. A red cross denotes an uncontrollable system due to failed rotors.

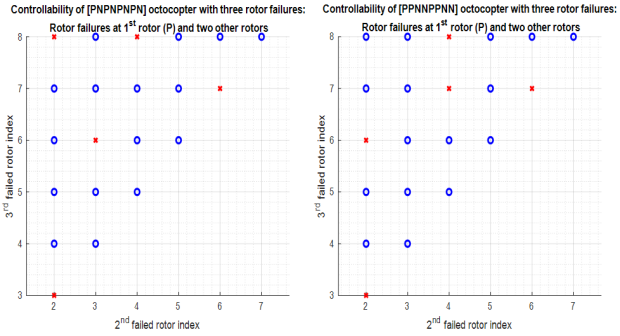


Fig. 3: Controllability of "PNPNPNPN" (left) and "PPNNPPNN" (right) octocopters subject to three rotor failures. Controllability is tested on the failure cases with first rotor failure at P (i.e. rotor index 1). Blue circles and red crosses denote controllable and uncontrollable systems, respectively, as in Fig 2.

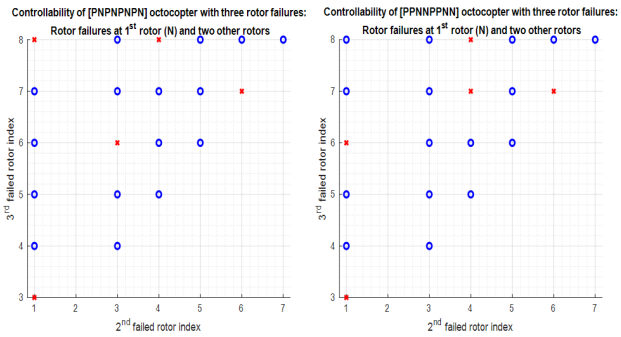


Fig. 4: Controllability of "PNPNPNPN" (left) and "PPNNPPNN" (right) octocopters subject to three rotor failures. Controllability is tested on the failure cases with first rotor failure at N (i.e. rotor index 2). Blue circles and red crosses denote controllable and uncontrollable systems, respectively, as in Fig 2.

Algorithm 1: EOD based reachability test

Input: EOD,t_{sf},t_{rem}
Output: R_{EOD} Flag
// EOD = Battery End of Discharge (s)
// t_{sf} = Safety Margin (s)
// t_{rem} = Remaining flight time (s)
if $(EOD \geq t_{rem} + t_{sf})$ **then**
    R_{EOD} = 1
else
    R_{EOD} = 0
end

Function 1: *Controllability and Reachability (C&R) Watchdog* verifies controllability and current landing site reachability based on the operational condition of vehicle systems and updated time until battery EOD.

Function 2: *Continue/Hold Selector* decides whether to continue executing the current flight plan or hold/loiter while a safe ACLM solution is defined. A new hold/loiter command will only be issued when a C&R Watchdog warning is active and the cost of continuing the current plan is higher than a hold/loiter. A hold/loiter would typically be issued when the current overflight area is not densely populated and traffic conflicts in the hold/loiter area are unlikely.

Function 3: *Landing Strategy Selection* retrieves and executes an emergency landing flight plan when a watchdog warning is first issued. A pre-computed contingency landing plan to a prepared site is executed when possible. Otherwise, a flight plan is generated to a nearby unprepared site that minimizes risk to people and property.

Function 4: *Offline Flight Planning* constructs a mission-specific flight plan database as part of pre-flight activities. As described below, plans to prepared landing sites near each nominal flight plan segment are generated and stored for in-flight use as needed.

We make the following simplifying assumptions for this work:

- AAM aircraft operate well above complex terrain and infrastructure, so contingency flight plans can be defined as direct paths from the initial state to the destination at a constant altitude or descending flight path angle that is dynamically feasible and guarantees obstacle clearance.
- There exist a sufficient number of potential emergency landing sites to enable at least one safe landing option throughout each flight.
- Failures possible in this work will involve only propulsion and battery systems. Nearby traffic will move aside to give priority to the distressed aircraft so emergency landing plans do not need to be deconflicted with other traffic.

Figure 5 shows the overall functionality and system flow of ACLM. ACLM input includes the fault/exception flag vector $F(t)$, the rotor health parameters $\eta_r(t)$, the rotor thrust commands $f_r(t)$, the time until battery system End of Discharge (EOD) t_{EOD}, and the current state vector $x(t)$.

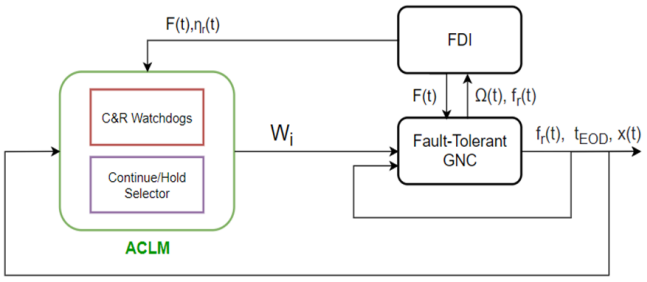


Fig. 5: Assured contingency landing management (ACLM).

ACLM output is a waypoint-based flight plan $\mathcal{W}_i$ defined by an n waypoint sequence:

$$\mathcal{W}_i = \{w_{i,1}, \cdots, w_{i,j}, \cdots, w_{i,n}\} \tag{5}$$

where each waypoint $w_{i,j}$ is given by:

$$w_{i,j} = \{\sigma_j, \lambda_j, h_j, t_j, \theta_j\} \tag{6}$$

Here σ_j denotes waypoint latitude, λ_j denotes waypoint longitude, h_j is the cruising altitude, t_j is the required travel time, and θ_j is the heading angle. Note that i indicates the database index of the selected landing flight plan, and j indicates the waypoint index. The output flight plan $\mathcal{W}_i$ becomes the reference trajectory the fault-tolerant controller follows to a safe landing.

B. C&R Watchdog Function

The C&R watchdog runs at each control cycle only if the contingency warning flag is clear (i.e., $W_{reachable} == 0$). Once the warning flag is set, landing strategy selection is activated to identify a safe emergency landing solution. The watchdog function confirms controllability and current landing site reachability as discussed in Section III. Fig. 6 shows C&R watchdog logic flow. Positive controllability is first tested. Flight termination is initiated if $C(t) < 0$, meaning that the controllability matrix is not full rank or the system does not have enough control authority to follow a prescribed flight plan. Otherwise, reachability $R(t)$ is confirmed based on the current location and updated estimate of battery EOD time. If $R(t) == 1$, no warning is issued and the current flight plan continues. If $R(t) == 0$, the C&R watchdog returns a warning flag ($W_{reachable} = 1$) and spawns the landing strategy selection thread to find an emergency landing solution as described in Subsection IV-D.

C. Continue/Hold Selector Thread

Typically ACLM will run in milliseconds due to its ability to rapidly retrieve and execute a safe contingency landing plan computed pre-flight. However, in the worst case (e.g., no prepared site is within the distressed aircraft's footprint), real-time planning will need to occur to select an unprepared landing site and build a feasible trajectory to that site. When online planning is required, ACLM execution time will be fast but nontrivial. The Continue/Hold Selector thread runs in

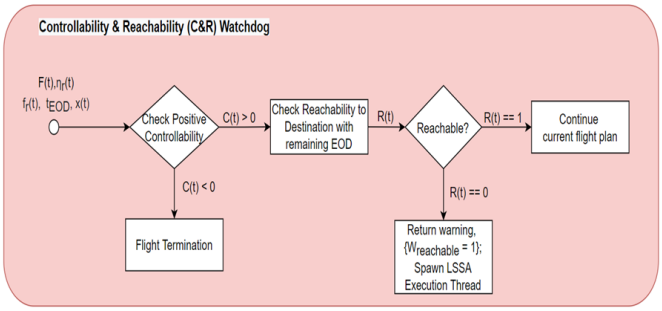


Fig. 6: Controllability and reachability watchdog logic.

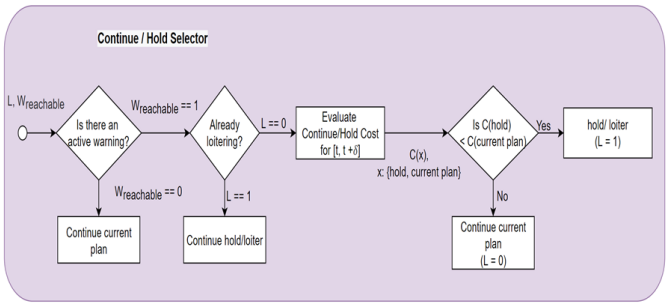


Fig. 7: Continue/Hold Selector Diagram.

parallel to manage this case. In particular, the selector checks whether it is safest to hold/loiter or continue executing the current flight plan while the safe landing solution is generated. Inputs to the selector thread are the loitering status L and the warning flag for reachability $W_{reachable}$.

When $W_{reachable} == 0$, the current flight plan is acceptable and can continue nominally to completion. If $W_{reachable} == 1$ and the vehicle is already loitering (L == 1), the loitering continues. Otherwise, the costs of loitering versus continuing the current flight path are evaluated over the near-term time horizon $[t, t+\delta]$, and the minimum cost option is selected or continued. Fig. 7 shows the Continue/Hold Selector logic. The cost function deciding whether to continue/hold is given by:

$$\begin{aligned} \mathcal{C}^i(\cdot) &= \sum\{\alpha_k * \mathcal{C}_k\} \\ &= \alpha_1 * \mathcal{C}_{grd}(\cdot) + \alpha_2 * \mathcal{C}_{air}(\cdot) + \alpha_3 * \ \mathcal{C}_{UL}(\cdot) \end{aligned} \tag{7}$$

where

$$\mathcal{C}_{UL}(\cdot) = \frac{1}{U_i} \tag{8}$$

The superscript i in the cost function indicates hold/loiter ($i = 1$), or continue ($i = 2$). α_i are cost function weights that typically sum to 1. $\mathcal{C}_{grd}$ is the ground occupancy risk (i.e., population density). It is defined as being immediately under the vehicle for the "loiter" action or the average population density over the current flight path for the near-term horizon in the "continue" action. $\mathcal{C}_{air}$ is the air traffic risk (i.e., airspace density). It is defined at the exact place where the vehicle is loitering for the "loiter" action or for the current flight path for the near-term horizon in the "continue" action. We assume that both $\mathcal{C}_{grd}$ and $\mathcal{C}_{air}$ are a-priori map information uploaded before the flight begins (provided by the UTM service providers). $\mathcal{C}_{UL}$ is the unprepared landing site proximity (i.e., inverse of the total number of unprepared landing sites U_i near the current location). We use the term **"unprepared landing site"** to refer to roads, rooftops, open fields, parking lots, etc., that are clear but not marked or protected. In contrast, **"prepared landing site"** refers to a published runway or vertiport.

D. Landing Strategy Selection Thread

The landing strategy selection thread is spawned when the C&R Watchdog first sets the landing site reachability warning flag (i.e., $W_{reachable==1}$) and the aircraft remains controllable. Per Fig. 8, landing strategy selection takes the fault/exception flag vector $F(t)$, the rotor health parameters $\eta_r(t)$, the rotor thrust commands $f_r(t)$, the time until battery system End of Discharge (EOD) t_{EOD}, and the current state vector $x(t)$ as inputs. It then uses the offline flight planning database in Sec. IV-D1 to look up a viable existing contingency flight plan to a prepared landing site. Once the potential candidate flight plans are generated, the plans are then checked for their reachability with given flight conditions (i.e., battery EOD and thrust vectors). If there exists at least one reachable flight plan as an emergency landing option, then the plan is executed. If there are multiple flight plans all satisfying reachability conditions to new destinations, then the plans are evaluated based on their cost as shown in Equation 9, and the minimum cost plan is chosen as a landing strategy solution.

$$\mathcal{C}_{LSS} = \beta * d_{travel} + \gamma * h_L \tag{9}$$

where β and γ are weight factors, and d_{travel} and h_L are the distance traveled from the mid/endpoint to the landing site and the height of the landing site, respectively.

The Warning flag, $W_{reachable}$ is cleared once the landing strategy selection thread successfully generates a solution. Fig. 8 shows the diagram of the landing strategy selection thread.

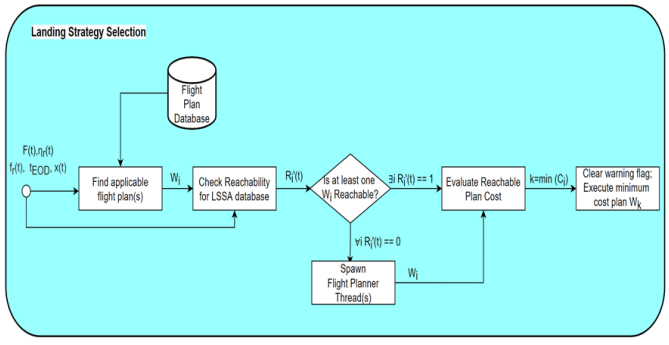


Fig. 8: Landing strategy selection logic flow.

1) Offline Flight Planning: Before an AAM operation executes, a flight plan is generated for the mission either from the operator or through the Air Traffic Controller (ATC). Offline flight planning uses this flight plan, $\mathcal{W}_{nom}$, to generate the potential landing sites in preparation for in-flight emergency

events. Fig. 9 shows the offline flight planning diagram with their inputs and outputs as follows:

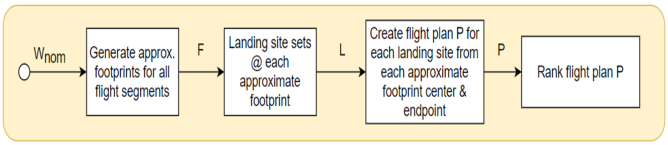


Fig. 9: Offline flight planning.

$$\mathcal{W}_{nom} = \{w_{nom,1}, \cdots, w_{nom,i}, \cdots, w_{nom,n}\} \tag{10}$$

$$w_{nom,i} = \{\sigma_i, \lambda_i, h_i, t_i, \theta_i\} \tag{11}$$

where $\mathcal{W}_{nom}$ and $w_{nom,i}$ are defined by equations 5-6 as seen in the nominal flight trajectory.

$$\mathcal{F} = \{F_1, \cdots, F_i, \cdots, F_n\} \tag{12}$$

$$F_i = \{d_i, \sigma_i, \lambda_i\} \tag{13}$$

$\mathcal{F}$ is the footprint dataset, and each element F_i contains the approximate footprint radius, d_i, and the latitude and longitude (σ_i, λ_i) of the mid-point of $[w_i, w_{i+1}]$ where the footprint is generated.

$$\mathcal{L} = \{L_1, \cdots, L_i, \cdots, L_n\} \tag{14}$$

$$L_i = \{i, c\} \tag{15}$$

$\mathcal{L}$ is the set of potential landing sites at each approximate footprint, F_i, where L_i has flight segment number, i, and landing site index, c. Then, the landing site data is saved as a look-up table thereby minimizing the search time for the potential candidate.

$$\mathcal{P} = \{P_1, \cdots, P_i, \cdots, P_n\} \tag{16}$$

$$P_i = \{P_{i,1}, \cdots, P_{i,k}, \cdots, P_{i,m}\} \tag{17}$$

$$P_{i,k} = \{P_{mid}, P_{end}\} \tag{18}$$

$\mathcal{P}$ denotes the flight plan set for all approximate footprints, where the set P_i contains individual flight plans for each landing site at each approximate footprint. $P_{i,k}$ contains two flight plans: (1) the flight plan starting at the midpoint of $[w_i, w_{i+1}]$ and (2) the flight plan starting at the endpoint at w_{i+1}. This allows a vehicle to select a flight plan that is closer to the current vehicle location in the corresponding flight segment i. Equation 9 is used to rank the flight plans, and the data is saved as a look-up table for the landing strategy selection. A visual representation of the offline flight plan database is shown in Fig. 10.

2) Flight Planner Thread: It might be the case, however, that no solution exists in the offline flight planning database. When none of the flight plans are reachable, then it is important to search for alternative viable landing options at unprepared landing sites (i.e., rooftops, roads, etc.). Then, a new flight planner thread is generated with a smaller online footprint based on the real-time $[t_{EOD}, x(t)]$ as the center of this new approximate footprint. Section IV-D2a below shows how the online footprint approximation of an energy-constrained multicopter is calculated. Note that Castagno et al. developed a risk-based planning approach for sUAS landing on rooftops using LIDAR, OpenStreetMap data, census, and terrain height sources in urban areas [21]. Such a method can be integrated into the flight planner thread to search for potential landing places at unprepared landing sites.

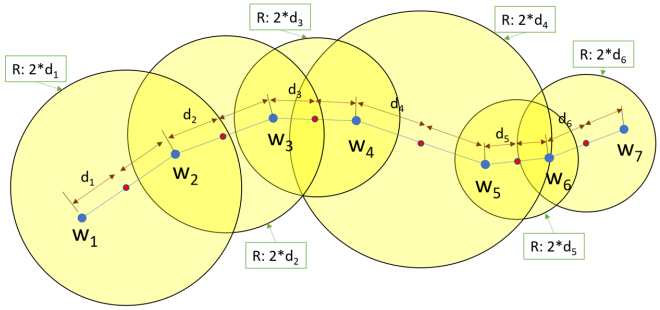


Fig. 10: Visual representation of offline flight plan database. Blue dots indicate flight segments in $\mathcal{W}_{nom}$. Red dots indicates the midpoint between the flight segments. Yellow circles are the approximate footprints with their radii, $R_i = 2 * d_i$, where the prepared landing sites are pre-selected and saved as dataset L_i. The corresponding flight plans $P_{i,k}$ are generated at the midpoint of $[w_i, w_{i+1}]$ and endpoint at w_{i+1}. The flight plans are ranked based on Equation 7 and used in the landing strategy selection thread.

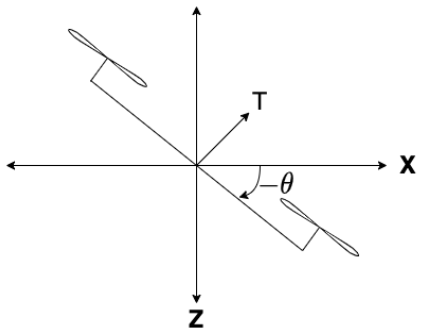


Fig. 11: 1-D motion diagram

a) Online Footprint Approximation of an Energy-constrained Multicopter System: This section provides an analytical solution that determines the approximate footprint for the multicopter given the End of Discharge (EOD) time for batteries in real time. We consider longitudinal level flight for a multicopter as shown in Figure 11. The equations of motion are defined as follows:

$$\begin{aligned} m\ddot{x} &= u - d_x\dot{x} \\ mg &= T\cos\theta \end{aligned} \tag{19}$$

where $u = -T\sin\theta$, T = thrust (N) and θ = pitch angle (rad). The state space form of equation (19) with position, $x_1 = x$, and velocity, $x_2 = \dot{x}_1$, is as follows:

$$\begin{aligned}\dot{x}_1 &= x_2 \\ \dot{x}_2 &= \frac{1}{m}u - \frac{d_x}{m}x_2\end{aligned} \tag{20}$$

The aim is to determine the minimum control effort input, hence the cost functional is defined as follows:

$$J = \frac{1}{2}\int_{t_0}^{t_f} u^2\, dt \tag{21}$$

where $t_0 = 0$, $t_f = EOD(s)$.

The Pontryagin Minimum Principle (PMP) is used to determine the optimal control input u^* [22]. Given the consideration of simple, longitudinal, level flight dynamics using the PMP provides an analytical solution. This solution can be computed in milliseconds.

The Hamiltonian is defined as follows:

$$H = \frac{1}{2}u^2 + [\lambda_1\ \lambda_2]\begin{bmatrix} x_2 \\ \frac{1}{m}u - \frac{d_x}{m}x_2 \end{bmatrix} \tag{22}$$

The obtained solution for optimal states (x_1^*,x_2^*), co-states (λ_1^*, λ_2^*), and controls (u^*) is given by the following set of equations:

$$x_1^*(t) = -\frac{C_1}{d_x^2}t + \frac{C_2}{2d_x^2}\exp\left(\frac{d_x}{m}t\right) - \frac{C_3}{d_x}m\exp\left(\frac{-d_x}{m}t\right) + C_4 \tag{23}$$

$$x_2^*(t) = -\frac{C_1}{d_x^2} + \frac{C_2}{2d_x m}\exp\left(\frac{d_x}{m}t\right) - C_3\exp\left(\frac{-d_x}{m}t\right) \tag{24}$$

$$u^*(t) = -\frac{C_1}{d_x} + \frac{C_2}{m}\exp\left(\frac{d_x}{m}t\right) \tag{25}$$

$$\lambda_1^*(t) = C_1 \tag{26}$$

$$\lambda_2^*(t) = \frac{mC_1}{d_x} - C_2\exp\left(\frac{d_x}{m}t\right) \tag{27}$$

In order to determine the maximum distance covered by the hexacopter in the forward and backward directions, we formulate the problem as a fixed final time and free final state problem to determine the constants C_1, C_2, C_3, C_4. The following boundary conditions are used for determining the maximum distance traveled forward:

$$x_1(t_0) = x_0,\ x_2(t_0) = v_{init},\ x_2(t_f) = v_{init},\ \lambda_2(t_f) = 0 \tag{28}$$

To determine the maximum distance traveled backward, given an initial forward velocity, the following boundary conditions are used:

$$x_1(t_0) = x_0,\ x_2(t_0) = v_{init},\ x_2(t_f) = -v_{init},\ \lambda_2(t_f) = 0 \tag{29}$$

The initial and final velocities are kept the same to represent a constant velocity profile. This results in a constant current draw from the battery and the battery EOD value decreases at a constant rate to reach the final position for level flight. Given the boundary conditions, the system of linear equations is solved to obtain the C_i values. In the implementation, the exp is replaced by the n^{th} order MacLauren Series expansion of the exponential. For the current implementation, we take $n = 5$.

Once the optimal control ($u^*(t)$) is calculated for steady level flight, we can extract the T and θ values by rearranging terms in (19) and (25) as shown in the following equations:

$$\theta_{opt}(t) = atan^{-1}\left(\frac{-u(t)}{mg}\right) \tag{30}$$

$$T_{opt}(t) = \frac{mg}{cos(\theta_{opt}(t))} \tag{31}$$

Algorithm 2 determines the approximate footprint for the multicopter given the End of Discharge (EOD) time for batteries in real time. To determine the boundary of the footprint, since the multicopter can instantaneously accelerate along the x, y axis, an analysis similar to longitudinal level flight can be conducted for lateral level flight and would provide similar results. Hence, the limit along the lateral direction is assumed to be the same as the limit for the longitudinal/forward direction as shown in Figure 12.

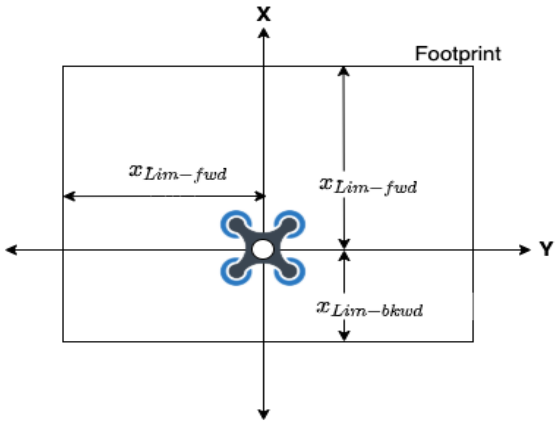


Fig. 12: Footprint of a multicopter

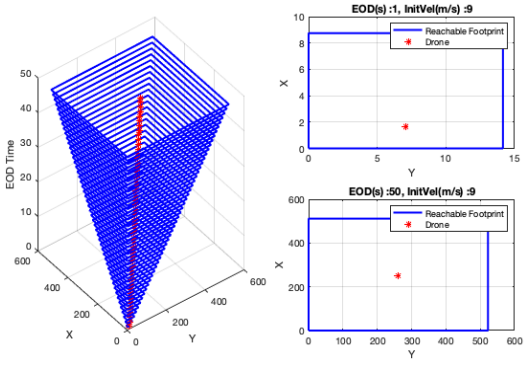


Fig. 13: (Left): Variation in footprint of a multicopter for different EOD values and initial velocity of 9 m/s, (Right Top): Footprint for EOD value of 1 sec, (Right Bottom): Foortprint for EOD value of 50 sec.

Algorithm 2: Online Footprint Approximation

Input: v_{init},EOD,T_{Max}
Output: Footprint Limits
// v_{init} = Current Velocity (x,y)
// EOD = Battery End of Discharge (s)
// T_{Max} = Maximum Total Thrust (N)
Assign the boundary conditions as per eqn (28),(29)
Calculate C_i values for forward and backward motion
Determine u^*_{fwd},u^*_{bkwd} using eqn (25)
Determine [$\theta_{opt-fwd}$, $T_{opt-fwd}$] and [$\theta_{opt-bkwd}$, $T_{opt-bkwd}$] using eqn (31),(30)
Bound the [θ_{opt} , T_{opt}] with $\theta_{min/max}$ obtained from available T_{Max} for level flight
Determine u_{bound} using eqn (19) and bounded values of θ_{bound},T_{bound}
Simulate system given by eqn (20) until EOD using u_{bound}
Determine footprint limits based on the position at end of simulation

Algorithm 3: Real-time Reachability

Input: $\mathcal{F}_{RT}$,P_{curr},P_{ELSite},v_{init},ψ
Output: R_{RT}
// $\mathcal{F}_{RT}$ = Real Time foot print boundaries
// P_{curr} = Coordinates of Current Position (x,y)
// P_{ELSite} = Coordinated of Emergency Landing site (x,y)
// v_{init} = Velocity along the x,y-axis
// ψ = Heading angle
// H = Homogeneous transformation from world frame to drone frame
$P^*_{ELSite} = H \times P_{ELSite}$
if P^*_{ELSite} *is within* $\mathcal{F}_{RT}$ **then**
  R_{RT} = 1
else
  $R_{RT} = 0$
end

b) Real Time Reachability: Once we obtain the approximate footprint boundaries as detailed in the previous section, the next step is to determine if the emergency landing site is reachable. Algorithm (3) performs the check to determine if the emergency landing site is within the approximate footprint. If the condition is satisfied, the real-time reachability flag is set to true. The following assumptions guarantee that if the emergency landing site is within the footprint it is reachable in the real world:

- There exists an obstacle-free straight-line path to the emergency landing site.
- The multicopter executes a level or descending flight to the emergency landing site. If the multicopter had to execute an ascent it would require more energy thus violating the validity of the EOD assumption in determining the footprint approximation.

An example of different footprints for an initial velocity of 9 m/s and EOD varying from 1 sec to 50 sec is shown in Figure 13. It can be seen from the figure that with higher EOD values the multicopter has sufficient time to decelerate and go backward, however at lower EOD values the distance that can be traveled backward is reduced.

V. Case Study Simulations and Result

Case studies tested our proposed ACLM on the constructed environment with prepared landing sites (i.e., vertiports) and unprepared landing sites (i.e., rooftops). Section V-A describes the environment and flight plan database construction. Section V-B shows the results of the case studies.

A. Offline Planning

a) Environment modeling: Prepared landing sites are generated arbitrarily near the random nominal flight plan, $\mathcal{W}_{nom}$, with a width and length of 10 [m]. The heights of the prepared landing sites vary from 0 [m] (i.e., ground level) up to 20 [m] so that the height factor can be accounted for in Equation 9 in order to find the best viable emergency landing solution. A total of 30 prepared landing sites are generated to test the ACLM. Fig. 14 shows the constructed environment with prepared landing sites and the nominal flight trajectory, $\mathcal{W}_{nom}$, with its cruise altitude at 40 [m].

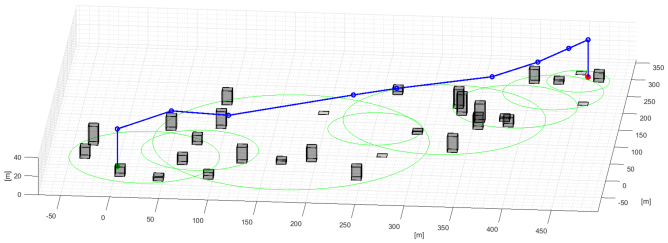


Fig. 14: Visualization of the 3-D offline environment construction of the flight plan. The nominal flight trajectory, $\mathcal{W}_{nom}$, is shown in blue with the origin and destination indicated by green and red solid dots. Approximate footprints at each midpoint between $w_{nom,i}$ and $w_{nom,i+1}$ are shown via the green circles. As each flight segment has a different length, the size of the approximate footprint changes accordingly. Prepared landing sites are shown in grey with varying heights.

b) Offline Emergency Flight Plan Database: The offline emergency flight plan database is created from the stored environment dataset. At each flight segment, $[w_{nom,i}, w_{nom,i+1}]$, flight plans P_i were generated at the midpoint and endpoint and saved as a data structure encapsulating Equations 10-18. Emergency in-flight decision-making in low-altitude airspace requires fast computation. By having a pre-stored dataset with viable landing options, a multicopter can quickly execute the flight plan to land at an emergency site with sufficient EOD. Fig. 15 shows the constructed offline emergency flight plan database for the environment given in Fig. 14. Cyan and red

lines indicate flight plans at the midpoint and endpoint at each flight segment.

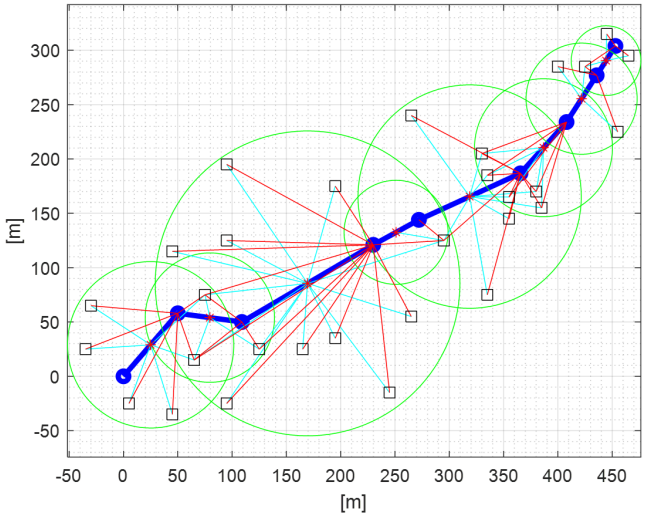


Fig. 15: Visualization of 2-D offline database with corresponding flight plans, $\mathcal{P}$. Nominal flight trajectory, $\mathcal{W}_{nom}$, is shown in blue. Approximate footprints at each midpoint between $w_{nom,i}$ and $w_{nom,i+1}$ are shown via the green circles. Prepared landing sites are shown as black squares. Flight plans from each midpoint (shown in red stars) to the landing sites at each approximate footprint is shown in cyan. Flight plans from each endpoint to the landing sites in that approximate footprint is shown in red.

B. Landing Strategy Selection Case Studies

Three case studies were performed to test ACLM, describing different solutions using: 1) an offline flight plan database and 2) online flight plan threads at various locations. All solutions were generated at the condition where a single rotor failure is detected through C&R Watchdog thread. While an emergency landing solution is computed, the Continue/Hold Selector thread maintains the vehicle either in cruise or hold/loiter. In this simulation, it is assumed that once a rotor fails, all other rotors experience no degradation in their health until the flight terminates.

a) Solution Using Offline Flight Plan Database: The landing strategy selection thread uses an offline flight plan database to generate multiple emergency landing solutions. The thread requires several inputs as explained in Section IV-D, one of which is the estimated EOD at the current vehicle position. To test our landing strategy selection algorithm, a simple linear relation was assumed between the distance traveled and the EOD while a multicopter travels at constant speed level flight ($v = 10\ m/s$). Fig. 16 shows the values of EOD vs. distance traveled used in this simulation. We assume that there is a 30% sudden decrease in EOD when experiencing a rotor failure. This is due to the increase in current drawn to compensate for the failed rotor and its resultant loss of thrust. The total distance traveled in the nominal flight mission (i.e., without a rotor failure) is 657 [m]. The first case study scenario has a rotor failure after the multicopter travels 238 [m], as shown in Fig. 16 with a vertical dashed line, with the EOD drop shown in black.

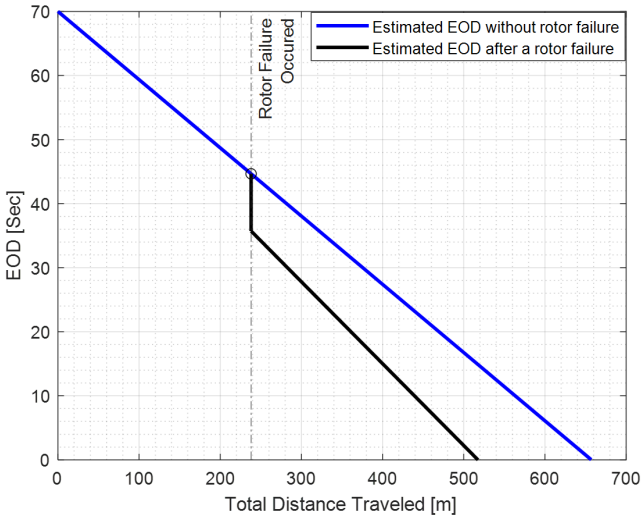


Fig. 16: EOD vs. total distance traveled. A linear relation was assumed between EOD and total distance traveled for a multicopter maintaining level flight at constant speed. With a rotor failure, it is assumed that there is 30% drop in EOD. Rotor failure occurred after flying 238 [m] for case study 1.

After a rotor fails, the landing strategy selection thread identifies the flight segment (i.e., 3^{rd} flight segment) wherein the vehicle is currently located. Then, the algorithm uses the offline flight plan database for that flight segment along with its corresponding flight plans. Fig. 17 shows the landing strategy selection of case study 1. Since the vehicle is currently located before the midpoint of the third flight segment, flight plans starting at the midpoint were used to find viable landing sites. Since this rotor failure instance occurred at an early phase of the flight operation, the landing strategy selection threads determined that all prepared landing sites inside the approximate footprint are reachable (as shown in green). Using Equation 9, the best landing site was determined as shown in magenta.

For case study 2, we investigate a rotor failure occurring in 3^{rd} flight segment after its midpoint. The rotor failure was simulated after the multicopter travels 318 [m]. Fig. 18 shows the EOD with a 30% drop and the reduced maximum distance the multicopter can travel. Our landing strategy selection algorithm revealed that 4 out of 15 prepared landing sites are not reachable. Using Equation 9, the cost of each reachable prepared landing site was calculated, and the minimum cost landing site was selected as the emergency landing site. Fig. 19 shows the results of case study 2.

b) Solution Using Online Flight Planner Thread: When the landing strategy selection algorithm cannot generate a viable solution using an offline flight plan dataset, the online flight planner thread calculates an alternative emergency landing solution by searching unprepared landing sites. A case study was created for this scenario, testing the Algorithms 2-3 in an online flight planner. For this particular case, a

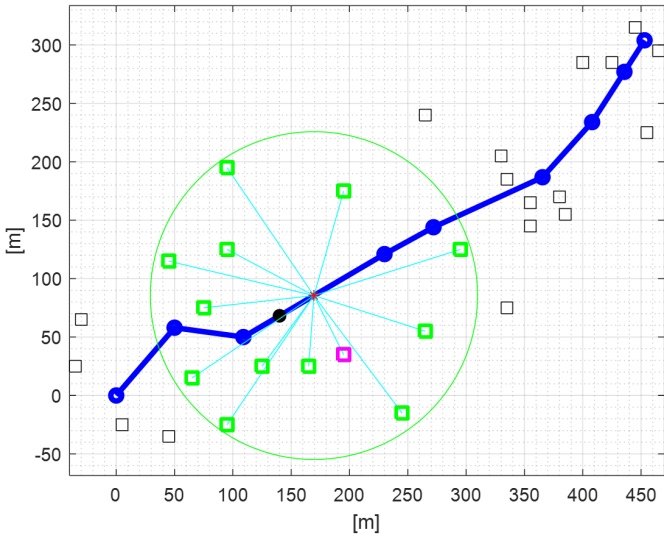


Fig. 17: Case study 1–Rotor failure in the third flight segment $[w_{nom,3}, w_{nom,4}]$. Green circle indicates the offline approximate footprint of the third flight segment. Red star indicates the midpoint of the flight segment. Current position of the multicopter is shown as a solid black dot located before the **midpoint** of the third flight segment. All prepared landing sites for the third flight segment, P_3, are reachable as shown in solid green. The best emergency landing location (shown in magenta) is selected based on the minimum cost calculated via Equation 9.

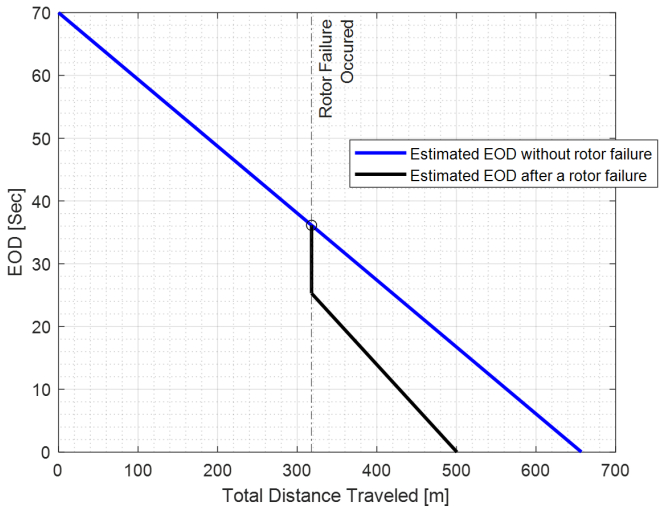


Fig. 18: EOD vs. total distance traveled for case study 2. Same assumption was made as in Fig. 16. Rotor failure occurred after travelling 318 [m].

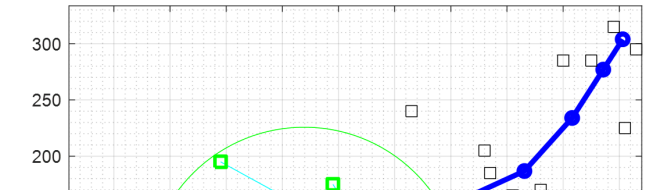


Fig. 19: Case study 2–Rotor failure in the third flight segment $[w_{nom,3}, w_{nom,4}]$. Green circle indicates the offline approximate footprint in the third flight segment. Red star indicates the midpoint of the flight segment. Current position of the multicopter is shown as a solid black dot located **after the midpoint** of the third flight segment. Reachable prepared landing sites for third flight segment, P_3, are shown in solid green. Unreachable prepared landing sites for the same flight segment are shown in solid red. The best emergency landing location (shown in magenta) is selected based on the minimum cost calculated via Equation 9.

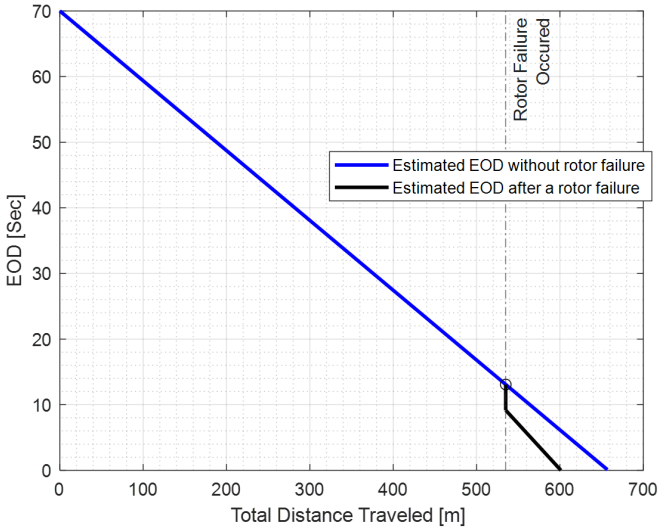


Fig. 20: EOD vs. total distance traveled for case study 3. Same assumption was made as in Fig. 16. Rotor failure occurred after travelling 535 [m].

rotor failure was triggered after the multicopter travels 535 [m]. Figures 20-21 show EOD vs. distance traveled and the landing strategy selection using the offline flight database for case study 3. As indicated by the red squares in Fig. 21, there is no reachable prepared landing site in this instance.

Landing strategy selection using an online flight planner is shown in Fig. 22. With the remaining flight time estimated from Fig. 20, an online footprint was generated for this energy-constrained multicopter that maintains its cruise speed at 10 [m/s]. For this particular case, three unprepared landing sites (shown in magenta) were found inside the online footprint approximation (shown via a red square). Then, the landing strategy selection cost equation was used to find the

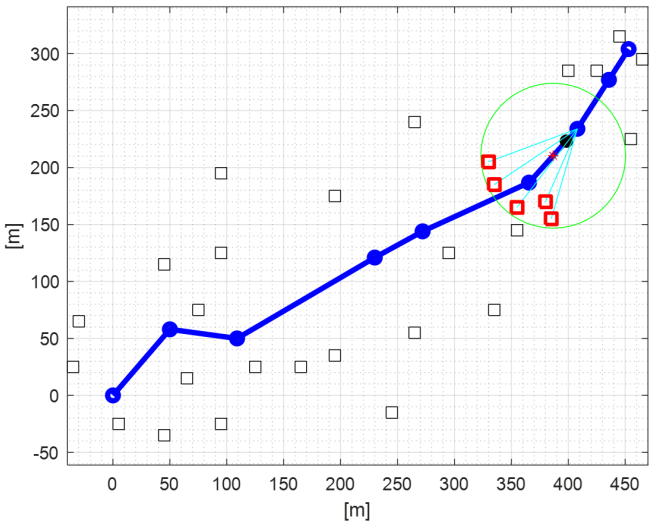


Fig. 21: Case study 3–Rotor failure in the sixth flight segment $[w_{nom,6}, w_{nom,7}]$. Green circle indicates the offline approximate footprint in the flight segment. Red star indicates the midpoint of the flight segment. Current position of the multicopter is shown in solid black dot located **after the midpoint** of the sixth flight segment. Flight plans are shown in cyan. As shown in red, none of the landing sites are reachable based on the EOD estimate.

minimum cost solution, as indicated in sold green.

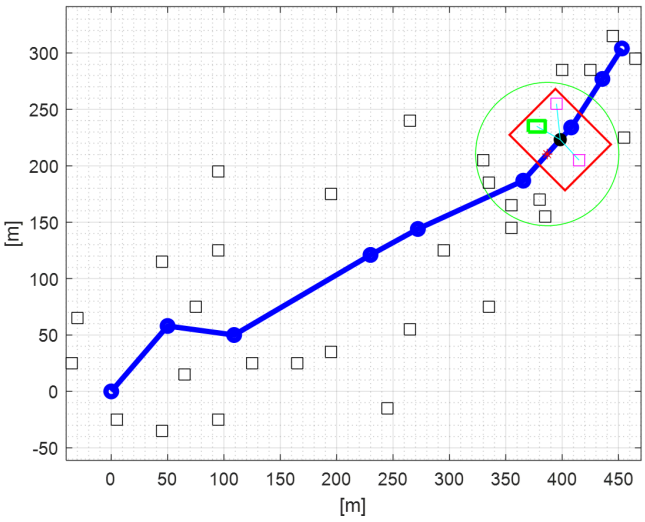


Fig. 22: Case study 3 with **online flight planner**–Rotor failure at the vehicle location in the sixth flight segment $[w_{nom,6}, w_{nom,7}]$. Green circle indicates the offline approximate footprint at the flight segment. Current position of the multicopter is shown by a red star. Online flight planner (shown via red rectangle) searches the landing site solutions at the solid black circle (a future location) so that the multicopter has a landing solution by the time it reaches the solid black circle. Unprepared landing sites are shown in magenta. The best unprepared landing site (shown in solid green) is found as the minimum cost of $[d_{travel}, h_{UL}]$ from Equation 9.

VI. Conclusion and Future Work

The proposed offline and online ACLM solutions, as well as their watchdog sub-threads, can support situational awareness (SA) and emergency decision-making for AAM missions experiencing an in-flight rotor failure in a low-altitude airspace. A C&R watchdog monitors the controllability of the multicopter, with and without the rotor failures, and checks the reachability of the energy-constrained system. The Continue/Hold Selector thread constantly provides commands to the multicopter to either loiter or continue on its current flight path. This helps the multicopter maintain flight while computing offline/online emergency landing solutions. Landing strategy selection is a sub-thread in the C&R watchdog that generates reachable landing site solutions using an offline flight plan database or an online flight planner. Case studies demonstrated how ACLM can generate safe landing options for a multicopter experiencing a rotor failure. Algorithms and definitions from this paper can contribute to a solid foundation of assured contingency landing management for AAM missions flying in low-altitude airspace.

Future work will extend our algorithms to meet different types of constraint requirements due to vehicle dynamics (e.g., quadplane etc.). Future endeavors will also perform Monte Carlo simulations and evaluate the run time characteristics of the ACLM framework.

References

[1] J. Holden and N. Goel, "Fast-forwarding to a future of on-demand urban air transportation [white paper]," 2016, available online: https://www.uber.com/elevate.pdf/ (accessed on 17 June 2021). [Online]. Available: https://www.uber.com/elevate.pdf

[2] A. Mathur, K. Panesar, J. Kim, E. M. Atkins, and N. Sarter, "Paths to autonomous vehicle operations for urban air mobility," in *AIAA Aviation 2019 Forum*, 2019, p. 3255.

[3] H. Z. I. Khan, J. Rajput, and J. Riaz, "Reconfigurable control of a class of multicopters," in *2020 European Control Conference (ECC)*. IEEE, 2020, pp. 1632–1637.

[4] H. Hamadi, "Fault-tolerant control of a multirotor unmanned aerial vehicle under hardware and software failures," Ph.D. dissertation, Compiègne, 2020.

[5] M. Chen and C. J. Tomlin, "Hamilton–jacobi reachability: Some recent theoretical advances and applications in unmanned airspace management," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 1, pp. 333–358, 2018.

[6] S. Bansal, M. Chen, S. Herbert, and C. J. Tomlin, "Hamilton-jacobi reachability: A brief overview and recent advances," in *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*. IEEE, 2017, pp. 2242–2253.

[7] A. K. Akametalu, C. J. Tomlin, and M. Chen, "Reachability-based forced landing system," *Journal of Guidance, Control, and Dynamics*, vol. 41, no. 12, pp. 2529–2542, 2018.

[8] P. F. A. Di Donato, S. Balachandran, K. McDonough, E. Atkins, and I. Kolmanovsky, "Envelope-aware flight management for loss of control prevention given rudder jam," *Journal of Guidance, Control, and Dynamics*, vol. 40, no. 4, pp. 1027–1041, 2017.

[9] E. M. Atkins, I. A. Portillo, and M. J. Strube, "Emergency flight planning applied to total loss of thrust," *Journal of aircraft*, vol. 43, no. 4, pp. 1205–1216, 2006.

[10] P. F. Di Donato and E. M. Atkins, "Optimizing steady turns for gliding trajectories," *Journal of Guidance, Control, and Dynamics*, vol. 39, no. 12, pp. 2627–2637, 2016.

[11] H. Lee, S.-H. Park, H.-T. Lee, B. Park, and J.-H. Han, "Lost c2 link contingency procedures for seoul tma and assessment on safety and controller workload," in *2020 AIAA/IEEE 39th Digital Avionics Systems Conference (DASC)*. IEEE, 2020, pp. 1–6.

[12] J. E. Baculi and C. A. Ippolito, *Onboard Decision-Making for Nominal and Contingency sUAS Flight*, 2019.

[13] M. Saied, H. Shraim, B. Lussier, I. Fantoni, and C. Francis, "Local controllability and attitude stabilization of multirotor uavs: Validation on a coaxial octorotor," *Robotics and Autonomous Systems*, vol. 91, pp. 128–138, 2017.
[14] E. D. Sontag, "Kalman's controllability rank condition: from linear to nonlinear," in *Mathematical system theory*. Springer, 1991, pp. 453–462.
[15] Q. Quan, *Introduction to multicopter design and control*. Springer, 2017.
[16] R. F. Brammer, "Controllability in linear autonomous systems with positive controllers," *SIAM Journal on Control*, vol. 10, no. 2, pp. 339–353, 1972.
[17] G.-X. Du, Q. Quan, B. Yang, and K.-Y. Cai, "Controllability analysis for multirotor helicopter rotor degradation and failure," *Journal of Guidance, Control, and Dynamics*, vol. 38, no. 5, pp. 978–985, 2015.
[18] G.-X. Du, Q. Quan, and K.-Y. Cai, "Controllability analysis and degraded control for a class of hexacopters subject to rotor failures," *Journal of Intelligent & Robotic Systems*, vol. 78, no. 1, pp. 143–157, 2015.
[19] T. Schneider, G. Ducard, K. Rudin, and P. Strupler, "Fault-tolerant control allocation for multirotor helicopters using parametric programming," *rN*, vol. 1, p. r2, 2012.
[20] H. Z. I. Khan, J. Rajput, and J. Riaz, "A robust nonlinear dynamic inversion control of a class of multicopters," in *2020 17th International Bhurban Conference on Applied Sciences and Technology (IBCAST)*. IEEE, 2020, pp. 264–270.
[21] J. Castagno, C. Ochoa, and E. Atkins, "Comprehensive risk-based planning for small unmanned aircraft system rooftop landing," in *2018 International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE, 2018, pp. 1031–1040.
[22] D. S. Naidu, *Optimal control systems*. CRC press, 2002.