Multi-Level Adaptation for Automatic Landing with Engine Failure under Weather Uncertainties

Haotian Gu* and Hamidreza Jafarnejadsani†
Stevens Institute of Technology, Hoboken, New Jersey, 07030

This paper addresses unmanned aerial vehicle (UAV) navigation and path planning under engine-out case for landing under severe weather using a Multi-Level Adaptation approach. This is a milestone in a novel autopilot framework, which enables a UAV under large uncertainties to perform safety maneuvers that are traditionally reserved for human pilots with sufficient experience. In addition, we present a high-fidelity simulation environment for fixed-wing aircraft to test and validate the approach under various uncertainties. For the proposed Multi-Level Adaptation autopilot framework, we present the design and analysis as well as the simulation results for the case of emergency landing due to engine failure under severe weather conditions, a challenging task for an autonomous aircraft.

I. Introduction

The technology of unmanned aerial vehicles (UAVs), which is moving towards full autonomous flight, requires operation under uncertainties due to dynamic environments, interaction with humans, system faults, and even malicious cyber attacks. Ensuring security and safety is the first step to make the solutions using such systems certifiable and scalable. In the following, we introduce an autopilot framework called “Multi-Level Adaptive Safety Control" (MASC) for the resilient control of autonomous UAVs.

A. MASC Architecture

In 2009, an Airbus A320 passenger plane (US Airways flight 1549) lost both engines minutes after take-off from LaGuardia airport in New York City due to severe bird strikes [1]. Captain Sullenberger safely landed the plane in the nearby Hudson River. Inspired by this story, we aim to equip UAVs with the capability of human pilots to determine if the current mission is still possible after a severe system failure. If not, the mission is re-planned so that it can be accomplished using the remaining capabilities. This is achieved by the proposed autopilot framework, MASC, which is capable of performing safe maneuvers that are traditionally reserved for human pilots.

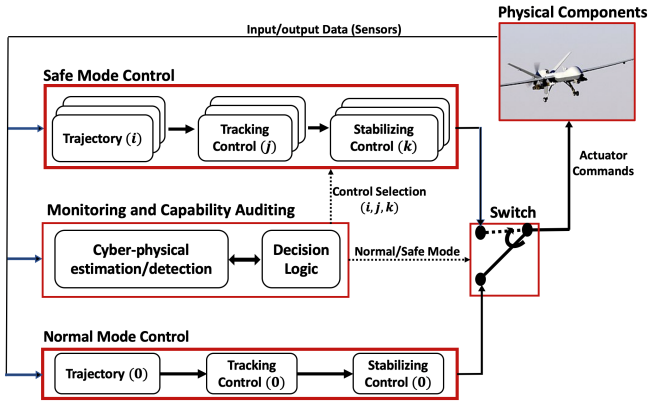

Fig. 1 Multi-Level Adaptive Safety Control (MASC) framework.

From a mission control architecture perspective, we aim to replace the traditional top-down, one-way adaptation, that starts with mission planning and cascades down to trajectory generation, tracking, and finally stabilizing controller, with an integrated top-down and bottom-up architecture that allows for two-way adaptation between planning and control to

*Ph.D. Student, Department of Mechanical Engineering, Hoboken, New Jersey, USA, AIAA Student Member.
†Assistant Professor, Department of Mechanical Engineering, Hoboken, New Jersey, USA, AIAA Member.

improve the stability and robustness of the system. To this end, we build the MASC architecture upon the Simplex fault-tolerant architecture [2–6], which is recognized as a useful approach for the protection of cyber-physical systems against various software failures. By integrating the MASC framework with the Simplex architecture, we aim to enable cyber-physical systems to handle large uncertainties originating from the physical world. The MASC framework, shown in Figure 1, consists of the following components:

- Normal Mode Controller: equipped with complex functionalities to operate the system under normal conditions.
- Safe Mode Controller with Multi-Level Adaptation: a simple and verified controller that ensures safe and stable operations of the system with limited levels of performance and reduced functionalities. The control architecture consists of three levels: i) mission re-planning and trajectory regeneration; ii) trajectory tracking and path following control; and iii) robust low-level adaptive controller.
- Monitoring and Capability Auditing: uses a model considering the cyber-physical nature of autonomous systems for estimation and fault detection. The model identifies the remaining capabilities of the system and its decision logic triggers a switch from Normal Mode to Safe Mode.

Under circumstances with large uncertainty such as engine failures, the proposed architecture re-plans/adapts the mission to the new constraints by i) auditing the remaining capability of the crippled aircraft and providing feedback to the other layers, ii) updating the flight envelope, iii) computing the reachable destinations and selecting the low risk one; iv) generating the flight path, and v) using a robust adaptive controller to track the path and stay within the flight envelope of the crippled UAV. Feedback provided from the lower layers to the higher layers such as the mission planner allows for the interaction of the MASC modules and as a result a two-way adaptation between planning and control.

In this paper, we address a challenging but easy to diagnose fault: a total engine failure. We show that our Multi-Level Adaptive approach enables the UAV to land safely after an engine failure and under crosswind conditions. It is worth mentioning that despite focusing on engine failure under weather uncertainties, the capabilities of MASC are not limited to this case, and the framework can be extended to the other scenarios. The monitoring and capability auditing module of the framework (cf. Figure 1 and Section II.B), which performs fault/failure detection and provides the necessary information required for a mission adaptation, allows for performing safe maneuvers under a wide range of faults. The autopilot can adapt to mission planning scheme under support of high fidelity simulation environment. Furthermore, MASC framework is not limited in UAV and can be used to develop any path planning algorithm for ground mobile robot.

B. Related Work: Engine-Out Emergency Landing

UAVs play a significant role in a wide range of industries, including defense, transportation, and agriculture, to name a few [7–9]. Engine failure is one of the most hazardous situations for UAVs [10, 11]. While engine failures are not common in passenger aircraft, an engine-out accident is more probable for a low-cost commercial UAV. The safety risks are even higher if a UAV crashes over a populated area endangering people and infrastructure on the ground.

To mitigate the risks due to engine failure, numerous approaches for planning and control are proposed in the literature. An engine-out aircraft can be landed via adaptive flight planner (AFP) presented in this work[12]. This path planar for lost of thrust case performs the two main flight-planning tasks required to get a disabled aircraft safely on the ground: select a landing site, and construct a post failure trajectory that can safely reach that landing site. An adaptive trajectory generation scheme with a certain presumed best glide ratio and bank angle for turns is proposed in [13]. Additionally, a trajectory planning based on flight envelope and motion primitives is proposed in [14] for damaged aircraft. The reachable set for auto-landing is calculated by using optimal control theory in [15]. Although their architecture is comprehensive, published algorithms and results have focused on constructing hierarchical control architecture rather than high-level flight plan and autopilot design. In addition, most of the existing studies do not address emergency landing under additional environmental uncertainties, such as severe weather conditions. Instead, only simplified models of aircraft and the environment are considered in their simulation.

RRT is popular sample based path planning algorithm designed to efficiently search nonconvex, high-dimensional spaces by randomly building a space-filling tree. The algorithm creates a search tree, containing a set of nodes and the connecting paths edges set. This paper[16] builds a path planning scheme upon the optimal sampling-based algorithm RRT to generate landing trajectory in real-time and also examines its performance for simulated engine failures occurring in mountainous terrain. However, RRT-based algorithms are computationally demanding for planning large-scale smoother path[17]. The demand increases with dimensions of the searched state-space. Another motion planning for emergency landing is based on Artificial Potential Field (APF) [18]method and greedy search in the space of motion primitives. In APF[19], UAV is moving under the control of potential fields around the c-space. The UAVs

path is calculated on the basis of the resultant fields from the initial point to the target point. However, the conventional APF[20] encounter a trap of a local minimum when the attractive force and repulsive force reach a balance, which means that the UAV stops moving towards target causing the UAVs to crash when approaching to landing area. Also, These algorithm for large scale path planning such as emergency landing is not time efficient. In addition, Those path planning scheme is general method which can not adapt to varied airplane flight phase task. Most simulation conduct the emergency landing animation without meeting flight dynamic constraints and guaranteeing path smoothness.

Emergency landing due to an engine failure under severe weather conditions is a challenging task even for an experienced pilot. Our proposed approach, referred to as MASC framework, provides the autopilot with the agility required to compensate for uncertainties by adaptations in planning and control. The framework can be employed on dependable computing architectures for the safety control design. Also, we propose a computationally low-cost trajectory generation and tracking control approach, which can be computed in low latency on a low-cost real-time embedded platform on-board of most UAVs. Moreover, we test the MASC in a high-fidelity flight simulation environment, and we are able to achieve successful landings under a wide range of initial conditions (i.e., initial altitude, distance, and orientation relative to the landing site), stormy weather, and turbulence without the need to re-tune the autopilot parameters.

The contribution of this paper is two-fold. First of all, a novel, resilient and lightweight planning and control framework is proposed for the autopilot of autonomous UAVs to handle large uncertainties that arise out of engine failures using the Multi-Level Adaptive Safety Control autopilot. We extend the standard Simplex fault-tolerant architecture discussed in [2, 3] to address faults originated from the physical world in addition to software faults. Second, the proposed autopilot framework is integrated and implemented into a high-fidelity software-in-the-loop flight simulation environment for further verification and validation under various uncertainties.

This paper is organized as follows. The components of the Multi-level Adaptive Safety Control (MASC) Framework are presented in Section II. Particularly, Monitoring and Capability Auditing is discussed in Section II.B, while the safe mode control with Multi-Level Adaptation is developed in Section II.C. Section IV describes the high-fidelity software-in-the-loop (SIL) simulation environment for a fixed-wing aircraft and presents the simulation results. Finally, Section V concludes the paper.

II. Multi-level Adaptive Safety Control (MASC)

This section presents the components of the Multi-Level Adaptive Safety Control (MASC) framework.

A. Dynamic System Model

The capability auditing module has a set of stored expected models $\mathcal{D} = \{\mathcal{D}_1, \ldots, \mathcal{D}_N\}$ where the triple

$$\mathcal{D}_j = \{A_j, B_j, \Theta_j\} \tag{1}$$

represents the plant matrices $(A_j,\ B_j)$, and the uncertainty set Θ_j. In particular, each model is represented as

$$\mathcal{D}_j : \begin{cases} \dot{x}(t) = A_j x(t) + B_j(u(t) + f_j(x(t), t)), \\ y(t) = Cx(t), \quad x(t_0) = x_0, \end{cases} \tag{2}$$

where $x(t) \in \mathbb{R}^n$ is the state vector, and $y(t) \in \mathbb{R}^q$ is the available output measurement. The term $f_j \in \Theta_j,\ \forall (x, t) \in \mathbb{R}^n \times [0, \infty)$, represents unknown system uncertainties and disturbances subject to local Lipschitz continuity assumption. Control input $u(t)$ is the robust low-level adaptive controller that stabilizes the model $\mathcal{D}_j$ with guaranteed robustness margins for *apriori* given bounds on the uncertainties.

Remark 1 *It is worth to notice that $\mathcal{D}$ is a set of nominal/representative fault models. That is, it does not need to be perfect, and any modeling error is expressed as $f_j \in \Theta_j$. Given a nominal model, any model mismatch and external disturbance will be dealt by the safe mode controller described in Section II.*

B. Monitoring and Capability Auditing

Monitoring and Capability Auditing is an integral part of the MASC framework (shown in Figure 1), which performs the task of fault detection and isolation (FDI), i.e., to notice the existence of fault, and to further identify the fault model. Since the autopilot is characterized as a cyber-physical system, regardless of the location of the faulty elements, the

effect of the fault is always reflected on the physical world. Leveraging the measurement of physical state, we employ a model-based FDI approach used in control literature [21][22]. To this end, one can derive the discrete model of the system in (2) as

$$\mathcal{D}_j : \begin{cases} x_{k+1} = A_{j,d}\, x_k + B_{j,d}\, (u_k + d_{j,k}) + w_k, \\ y_k = C\, x_k + v_k, \quad x_0 = x_0, \end{cases} \tag{3}$$

where $w_k \sim \mathcal{N}(\mu_w, \Sigma_w)$ and $v_k \sim \mathcal{N}(\mu_v, \Sigma_v)$ are Gaussian noises, and unknown input $d_{j,k}$ represents model uncertainty $f_j(x(t), t)$ and discretization error. A generic form of a FDI is given by

$$O_j : \begin{cases} \hat{x}_{k+1} = \bar{A}_{j,d}\, \hat{x}_k + \bar{B}_{j,d}\, u_k + \bar{H}_{j,d}\, y_k, \\ \hat{y}_k = \bar{C}\, \hat{x}_k, \quad \hat{x}_0 = 0, \\ r_k = \bar{M}(\hat{y}_k - y_k), \end{cases} \tag{4}$$

where $\hat{x}_k$ is the estimation of system states x_k, $\hat{y}_k$ is the observer output, and the residual signal r_k is a quantified representation of the model discrepancy. Finally, $\bar{A}_{j,d}$, $\bar{B}_{j,d}$, $\bar{C}$, and $\bar{M}$ are observer matrices to be designed [21**?**].

The capability auditing module employs the residual r_k of the FDI in (4) and possibly hardware redundancy in the UAV's sensors and actuators to detect and identify the fault models and accordingly updates the current model of UAV with possible physical damages. Therefore, the identification of the new model is critical for stabilization of the UAV, and it should be prioritized computationally, while the mission re-planning algorithm can take longer to converge to a feasible trajectory.

In the particular case of engine malfunction , monitoring the sensors such as engine's RPM indicator, accelerometer provides sufficient information for the monitoring and capability auditing module to detect the engine failure. Then, the module activates the planing and control task specifically designed for emergency landing within the safe mode control module.

Having identified the faults and updated the system model $\mathcal{D}_j$, the second most important task of the monitoring and capability auditing module is to determine safe flight envelopes for the mission re-planing. Specifically, for protecting the flight envelope during the emergency landing, it is very crucial to maintain the forward airspeed around the optimal speed V_{opt} and the corresponding best slope γ_{opt} that are recommended by the aircraft manufacturer. The optimal speed ensures maximum gliding distance without stalling the aircraft. In addition, the following constraints are considered for motion planning:

$$\begin{aligned} V_{\min} < V < V_{\max}, \quad & p_{\min} < p < p_{\max}, \\ \theta_{\min} < \theta < \theta_{\max}, \quad & q_{\min} < q < q_{\max}, \\ \phi_{\min} < \phi < \phi_{\max}, \quad & r_{\min} < r < r_{\max}, \end{aligned} \tag{5}$$

where V, θ, and ϕ are the forward airspeed, pitch angle, and roll angle, respectively. Also, p, q, and r are roll, pitch, yaw rates, respectively.

C. Safe Mode Control with Multi-Level Adaptation

While the detection of engine failure is straightforward, the planning and control for motor malfunction emergency glide landing is a challenging task, particularly under severe weather conditions. Therefore, the main part of this paper is devoted to the development of the Landing Area selection module for being capable of optimal landing zone selection. A Multi-Level Adaptation approach is developed for the real-time planer to handle large uncertainties and disturbances (engine-out landing, severe weather conditions, etc.), which otherwise require intervention by a human operator. The proposed MASC framework will have reduced functionality, light weighted computation, but enhanced robustness and simplicity for certification. In addition, MASC uses data from only a subset of available sensors with high reliability. We refer to three different levels of adaptation:

– *Low-Level Controller Adaptation* adapts the low-level control laws for maintaining flight stability using robust adaptive control techniques;

– *Path Following Control Adaptation* modifies the control commands to the low-level controller to satisfy the safety constraints (cf. safety envelope (6));

– *Mission Adaptation* computes the reachable sets permitted by the remaining capabilities within the available potential and kinetic energy, selects the safe emergency landing site, and generates the flight trajectory.

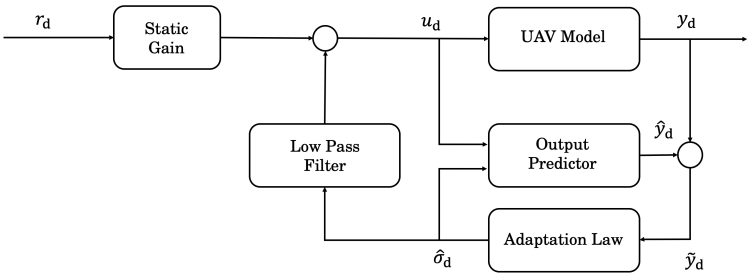

Fig. 2 The architecture of output-feedback $\mathcal{L}_1$ adaptive controller. The reference command $r_{\mathrm{d}}[\cdot]$ represents the desired roll and pitch angles that are given by the path following control module. The discrete-time signal $u_{\mathrm{d}}[\cdot]$ represents the computed elevator deflection, δ_a, and aileron deflection, δ_e, that will be the control commands to the UAV. The sampling time T_{s} of the low-level adaptive controller is 0.01 seconds.

D. Low-Level Controller Adaptation

A robust adaptive control law is used to stabilize the longitudinal and lateral dynamics of the fixed-wing aircraft and to track the commanded roll and pitch angles. The design of the controller requires the knowledge of an approximate plant model. This information, modeled as flight dynamics $\mathcal{D}_j$ in (2), is updated by the monitoring and capability auditing module. Notice that an accurate model is unnecessary as the controller can compensate for bounded model uncertainties.

Let $r_{\mathrm{d}}[\cdot]$ represent the given discrete-time reference commands of the desired roll and pitch angles in the lateral and longitudinal dynamics, respectively. To achieve the desired dynamics and track the referenced roll and pitch angles, we use an output-feedback $\mathcal{L}_1$ adaptive sampled-data (SD) controller developed in [23]. Figure 2 depicts the architecture of the low-level controller. $\mathcal{L}_1$ robust adaptive controller ensures uniform performance bounds with quantifiable robustness margins [24–26]. Also, the SD controller, which is in discrete-time, can be easily implemented in embedded software. The reference command $r_{\mathrm{d}}[\cdot]$ is provided by the path following control module that will be described in the following Section.

E. Path Following Control Adaptation

Monitoring and Capability Auditing will evaluate mission feasibility, given the states and the model of a damaged UAV. Large uncertainty mitigation requires mission adaptation and selection of a new trajectory that is still feasible, given the remaining capabilities. For large uncertainties outside the design bounds, i.e., $f_j \notin \Theta_j$ in (2), the control inputs can saturate, which drives the system to unsafe states. Modification of the reference command $r_{\mathrm{d}}[i]$ based on the updated objectives is another layer of defense for maintaining safety by *satisfying flight envelope constraints*. Therefore, we consider a control structure that consists of a path following controller, where the generated reference commands to the low-level controller are limited by saturation bounds to maintain the closed-loop system within operational safety envelope.

Let the reference command be constrained to a convex polytope as a safe operational region, defined by the set

$$\mathcal{R} = \{r_{\mathrm{d}} \in \mathbb{R}^q \,|\, \|W r_{\mathrm{d}}\|_\infty \leq 1\}\,, \tag{6}$$

where $W = \mathrm{diag}\{\mathrm{r}_{\mathrm{max}_1}^{-1}, \ldots, \mathrm{r}_{\mathrm{max}_q}^{-1}\}$, and the positive constants $\mathrm{r}_{\mathrm{max}_i}$'s are the saturation bounds on the reference commands. Then, the weighted reference command is bounded by

$$\|W r_{\mathrm{d}}[i]\|_\infty \leq 1, \quad i \in \mathbb{Z}_{\geq 0}.$$

In this paper, the reference command $r_{\mathrm{d}}[i],\ i \in \mathbb{Z}_{\geq 0}$, which is generated by the path following control law. We expect to stabilize the system $\frac{\partial x}{\partial t} = f(x,u)$ at each equilibrium point x_{ss}, which is similar as reference command $r_{\mathrm{d}}[i]$ of flight system. We define steady-state problem: find the steady-state control u_{ss} s.t. $0 = f(x_{\mathrm{ss}}, u_{\mathrm{ss}})$,where

$$\begin{aligned} x_{\mathrm{ss}} &= x - x_\delta \\ u_{\mathrm{ss}} &= u - u_\delta \end{aligned} \tag{7}$$

So,

$$\frac{\partial x_\delta}{\partial t} = f(x_{\mathrm{ss}} + x_\delta, u_{\mathrm{ss}} + u_\delta) \stackrel{\mathrm{def}}{=} f_\delta(x_\delta, u_\delta) \tag{8}$$

At each equilibrium point,

$$\begin{aligned} f_\delta(0,0) &= 0 \\ u_\delta = \gamma(x_\delta) \Rightarrow u &= u_{\text{ss}} + \gamma(x - x_{\text{ss}}) \end{aligned} \tag{9}$$

Since designed autopilot is proportional integral derivative controller, we define state feedback stabilization solution is to find

$$\begin{aligned} u &= \gamma(x) \\ [\gamma(0) &= 0] \end{aligned} \tag{10}$$

given:

$$\begin{aligned} \frac{\partial x}{\partial t} &= f(x,u) \\ [f(0,0) &= 0] \end{aligned} \tag{11}$$

s.t. each stabilized point is asymptotically stable equilibrium point of $\frac{\partial x}{\partial t} = f(x, \gamma(x))$.

To convert desired heading angle to desired roll angle, we embed a proportional controller between the mission planner and autopilot. Since the response speed is not so fast enough through controlling aileron to achieve desired heading angle, we consider transferring from the control goal of desired heading angle to roll angle so that speed up controller system response. We use proportional controller to regulate fixed wing UAV heading direction through controlling its roll angle. This controller requires define p gain and desired heading angle. The desired roll angle is given by

$$\phi = \text{atan2}\left(p_{\text{gain}} \times (\gamma - \gamma_{\text{ref}}) \times V, V_{\text{g}}\right) \tag{12}$$

where γ is the current heading angle of the aircraft in the horizontal plane. γ_{ref} denotes the desired heading angle output from carrot chasing based mission planner, given in radians. V is forward velocity of airplane in simulation and V_{g} is component of velocity along the direction of gravity.

F. Mission Adaptation

In the MASC framework, we present a mission planner in the Safe Control Mode that generates the landing trajectory to a safe landing site. The planner also evaluates mission feasibility considering the damage of the aircraft and environmental constraints. In the case of a fault/failure detection, once the capability auditing provides the new/altered model $\mathcal{D}_j$, two concurrent steps will be taken: *i)* a j^{th} low-level controller is activated based on its ability to stabilize the system around a pre-calculated $r_{\text{d}}^{th}[\cdot]$ reference command, and *ii)* MASC initiates the re-planning of the mission.

The first step is taken to ensure that the system does not violate its stability/safety bounds while the mission is being re-planned. Once the mission is re-planned, it is fed into the path following controller, which then accordingly alters the $r_{\text{d}}^{th}[\cdot]$ reference command provided to the j^{th} low-level controller to execute the new mission.

In the second step, re-planning of the mission, it is crucial to compute the *reachable area*, which is defined as all the spatial points the aircraft is capable of reaching given the dynamic constraints, potential and kinetic energy, and available fuel if the engine is still partially working. To this end, provided the information and new constraints by the monitoring and capability auditing module, a set of candidate locations are initially considered. This initial set may include near airports and empty lands. Leveraging the updated model of UAV, MASC evaluates the feasibility of safe landing in the candidate areas and identifies the most likely safe location for emergency landing. This safety-and-time-critical evaluation process is performed through simulating, in parallel and on the UAV's embedded board, the emergency landing in the potential reachable areas low. Any violation of the safety constraints (for instance (5) and (6)) during the evaluation process rules out a candidate area as a safe reachable area.

As mentioned earlier, the mission planner generates the parametrized motion trajectory, $z_{\text{m}_\text{d}}[i]$, for the new mission, known as the offline motion planning task. In this paper, the optimal landing area selection is based on carrot chasing method[27] which is motion offline planning for determining the landing zone. The trajectory consists of segments of basic maneuvers (i.e., straight line and loiter) for a time horizon of $\Delta t = 5s$ that are pieced together in real-time from the initial point to the landing site. By introducing a virtual target point (VTP) on the path, the UAV updates its heading direction toward the VTP to follow a desired path. The plant model following controller is UAV Guidance model which is reduced-order model for a closed-loop system including UAV dynamics and autopilot. The model approximates the

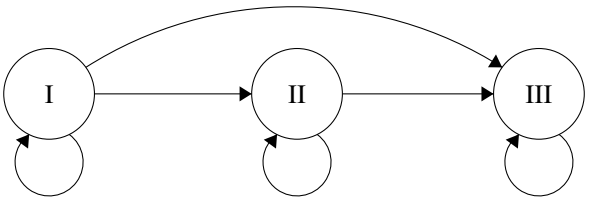

Fig. 3 Landing Mission Phase Transition Diagram. Phase I: Cruising, Phase II: Loitering, and Phase III: Approach.

behavior of a closed-loop system consisting of an autopilot controller and a kinematic UAV model for 3D motion. In offline path evaluation for landing zone determination part, the autopilot subscribe aircraft status from previous second to calculate aileron and elevator control command to chase the VTP so that construct a trajectory planning full loop.

To improve the resilience of the online motion planning in the presence of failures and external disturbances such as crosswind accompanied turbulence, this paper select the carrot chasing based guidance logic to do online trajectory following. The carrot chasing based method is more robust to the effect of disturbances[28]. It is by using an pseudo target moving along the desired flight path and meanwhile generates a desired heading angle using the reference point[29]. This lightweight nonlinear guidance logic [29] consumes the least control effort and takes shorter time to track the paths. For each trajectory segment, the flight path angle γ and the yaw rate $\dot{\psi}$ are constant. Formally, knowing the step response of a high fidelity motion controller gives information on the stability of such a system and on its ability to reach one stationary state when starting from another. We test the step response of high fidelity autopilot by feeding the step signal to this critically damped system, it needs $5s$ to stablize around the reference value. For emergency glide landing, the mission planner is updated with the true state of the aircraft every $\Delta t = 5s$. If the time step is smaller than the autopilot controller needed to reach stationary state, it may have extreme effects on the component itself and other portions of the overall system dependent on this component. In addition, The x-plane under the management of motion controller cannot act until the component's output settles down to some vicinity of its final state, which will delay the overall system response. Then, the aircraft state for the next step can be predicted using the following model

$$\begin{aligned} x[k+1] &= x[k] + \bar{V}_{g_x}[k]\cos(\psi[k]) \times \cos(\gamma[k]) \times \Delta t \\ y[k+1] &= y[k] + \bar{V}_{g_y}[k]\sin(\psi[k]) \times \cos(\gamma[k]) \times \Delta t \\ z[k+1] &= z[k] + \bar{V}_{g_z}[k]\sin(\gamma[k]) \times \Delta t \end{aligned} \tag{13}$$

where c is a constant. Also, V_{g_x}, V_{g_y}, and V_{g_z} are x, y, and z components of the aircraft's velocity with respect to the ground, respectively. In addition, $\Delta\psi[k]$ and $\Delta\gamma[k]$ are the inputs of the system in (13) generated by the mission planner at each step.

Remark 2 *The simultaneous use of ground velocity, which is available via GPS measurement, along with the airspeed improves the one-step prediction model in the presence of windy and turbulent weather conditions. Therefore, the planning algorithm of this paper is wind-aware which actively compensates for weather uncertainties, accordingly.*

The landing mission is divided into three Phases: *Phase I: Cruising, Phase II: Loitering, and Phase III: Approach* as illustrated in Figure 3. By default, the emergency glide landing starts by cruising to the center of loitering circle denoted by (x_l, y_l) near the landing site. After the aircraft reaches close enough to the loitering center, i.e., $\sqrt{(x_l - x)^2 + (y_l - y)^2} < R_l$, the mission enters the loitering phase. While loitering, the aircraft losses altitude in a spiral trajectory. When the cut-off altitude, z_a, is reached, i.e., $z < z_a$, the mission enters into the approach phase. Notice that the mission is allowed to progress in one direction similar to a directed acyclic graph (DAG), and it is possible that the first and second phases are skipped altogether depending to the initial states of the aircraft as illustrated in Figure 3.

The cost function is given by

$$J = \begin{cases} J_{\text{cruising}}, & \text{if Phase I,} \\ J_{\text{loitering}}, & \text{if Phase II,} \\ J_{\text{approach}}, & \text{if Phase III,} \end{cases} \tag{14}$$

where the time step k is dropped from the notation for brevity. In the following, we specify the nonlinear guidance logic and corresponding cost functions for each phase.

1. Phase I

During the cruising phase, the aircraft cruises to the center of the loitering circle. The desired heading angle is given by

$$\begin{aligned} x[k+1] &= (R_u + \delta) \times \cos(\theta) \\ y[k+1] &= (R_u + \delta) \times \sin(\theta) \\ \psi_d &= \text{atan2}((y[k+1] - y[k]), (x[k+1] - x[k])) \end{aligned} \tag{15}$$

where

$$\begin{aligned} \theta &= atan2((W_{\text{ip1}}(2) - W_{\text{i}}(2)), (W_{\text{ip1}}(1) - W_{\text{i}}(1))) \\ R_u &= \sqrt{(path_g^2 - (path_g \times sin(\theta - \theta_{\text{u}}))^2)} \end{aligned}$$

where

$$\begin{aligned} path_{\text{g}} &= \sqrt{(x[k] - W_{\text{i}}(1))^2 + (y[k] - W_{\text{i}}(2))^2)} \\ \theta_{\text{u}} &= atan2((y[k] - W_{\text{i}}(2)), (x[k] - W_{\text{i}}(1)) \end{aligned}$$

The angle θ is the line-of-sight (LOS) formed by reference straight line with respect to the x-y coordinate frame angle for phase I III. θ denotes the angle in the Euclidean plane, given in radians, between the positive x-axis and the ray to the point (W_{ipl}, W_{i}) which connects engine malfunction position and loiter center. $path_{\text{g}}$ is Euclidean distance of current coordinates and start of reference path. R_{u} is the distance of start of reference path and current particle projected on the straight reference path. The airspeed should remain close to the best gliding speed V_{opt} while airplane cruises to the landing site. Then, the cost function can be expressed as

$$J_{\text{cruising}} = w_\psi \left|\psi - \psi_{\text{des}}\right| + w_{\text{V}} \left|V - V_{\text{opt}}\right| \tag{16}$$

where w_ψ and w_{V} are the weights for cost terms corresponding to heading error and speed error, respectively.

2. Phase II

In the loitering phase, the aircraft loiters near the landing site in spiral trajectory to lose any excessive altitude for the final approach.

$$\begin{aligned} x[k+1] &= o_x + r \times \cos(\theta + \delta) \\ y[k+1] &= o_y + r \times \sin(\theta + \delta) \\ \psi_d &= \text{atan2}((y[k+1] - y[k]), (x[k+1] - x[k])) \end{aligned} \tag{17}$$

where

$$\theta = atan2((o_y - y[k]), (o_x - x[k]))$$

o is global coordinates of loiter center. Also, θ will gradually approach to the chord tangent angle of loiter as the moving trajectory converging to the reference circle. In addition, δ is look ahead distance in (15) and in (17). With an increase in δ, the UAV is able to settle on the path quickly, reducing the cross-track error[30]. The cost function for this phase is given by

$$J_{\text{loitering}} = w_\psi \left|\psi - \psi_{\text{des}}\right| + w_{\text{V}} \left|V - V_{\text{opt}}\right|, \tag{18}$$

where ψ_{des} is given by the carrot-chasing algorithm for circular path with radius $R_{\text{c}} < R_{\text{l}}$.

3. Phase III

In the approach phase, the aircraft aligns itself with the runway and tracks the desired flight path angle for the final approach. The trajectory point renew scheme is same as Phase I. The difference is θ denotes the angle in the Euclidean plane, given in radians, between the positive x-axis and the ray to the point (W_{ipl}, W_{i}) which connects (x_{u}, y_{u}) and and airport coordinates. ψ_{f} is the runway direction. If the ground distance of the aircraft to the landing site is larger than a constant R, i.e., $\sqrt{(x-x_{\text{f}})^2+(y-y_{\text{f}})^2} > R$,

$$x_{\text{u}} = x_{\text{l}} + R_{\text{l}} \times cos(\psi_{\text{f}} - \pi)$$

$$y_{\text{u}} = y_{\text{l}} + R_{\text{l}} \times sin(\psi_{\text{f}} - \pi)$$

where R_{l} is loiter diameter. ψ_{f} is heading angle of runway. (W_{ipl}, W_{i}) connects the (x_{u}, y_{u}) start of the reference line and landing position coordinates in Euclidean plane. Then the cost function for the approach phase is given by

$$J_{\text{approach}} = w_{\psi} \left|\psi - \psi_{\text{des}}\right| + w_{\text{V}} \left|V - V_{\text{opt}}\right| + w_{\Gamma} \left|\Gamma - \Gamma_{\text{f}}\right|, \tag{19}$$

where the approach angle Γ is given by

$$\Gamma = -\operatorname{atan2}(z - z_{\text{f}}, \sqrt{(x-x_{\text{f}})^2+(y-y_{\text{f}})^2}. \tag{20}$$

If $\sqrt{(x-x_{\text{f}})^2+(y-y_{\text{f}})^2} \leq R$, the cost function for approach phase is given by

$$J_{\text{approach}} = w_{\psi} \left|\psi - \psi_{\text{des}}\right| + w_{\text{V}} \left|V - V_{\text{f}}\right| + w_{\gamma} \left|\gamma - \gamma_{\text{f}}\right|.$$

The desired heading angle ψ_{des} in (19)-(20) is given by the carrot-chasing algorithm for straight-line following algorithm.

III. Software-in-the-loop Implementation

Multi-Level Adaptive Safety Control includes two modules, one is Monitoring and Capability Auditing will evaluate mission feasibility for engine malfunction airplane. Another is online mission planning and navigation to adapt to emergency case. This section presents a software-in-the-loop (SIL) simulation scheme to evaluate and validate the proposed MASC framework for online path planning and navigation under the emergency case.

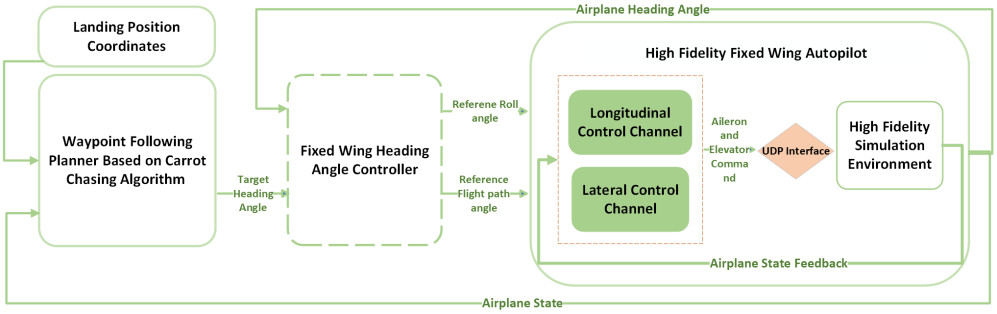

Fig. 4 MASC Autopilot(Online Navigation)

A. UDP Receiver and Sender modules for the SIL

The SIL architecture for the MASC autopilot, shown in Figure 4, has four primary components: i) nonlinear logic based mission planner, ii) proportional heading angel regulation scheme, iii)high-fidelity physical simulation environment (in X-Plane), and iv) MASC autopilot (in MATLAB/Simulink). This section is introduction of iii) and iv) which are shown in Figure 4. MASC autopilot contains a mid-level autopilot and plant model. It will be simulated in the X-Plane flight simulation program. X-Plane has been certified by the Federal Aviation Administration (FAA) as a simulation software to train pilots. X-Plane adopts the User Datagram Protocol (UDP) to communicate with the third-party software and external processes. Unlike the Transmission Control Protocol (TCP), UDP assures that

data packages will arrive completely and orderly. Also, the communication via UDP can achieve high-speed data traffic compared to other protocols by efficient use of bandwidth, which is an advantageous characteristic for the simulation under consideration. Correspondingly, MATLAB/Simulink supports the UDP communication through DSP System Toolbox. This toolbox can query an application using UDP to send real-time data from the Simulink model to the corresponding channel in X-Plane. Also, the UDP object allows performing byte-type and datagram-type communication using a UDP socket in the local host.

For the implementation, we consider designing subscriber and publisher to guarantee communication between the X-Plane and MATLAB/Simulink in real time. For subscriber, we used two Simulink blocks: an embedded MATLAB function and a byte unpack. For publisher, we use a byte pack and encoder, and both are linked via a bus module in Simulink. The mid-level autopilot calculates the aileron and elevator movement command to navigate the X-Plane to approach reference flight path angle and heading angle at each time step. The autopilot publishes the calculated aileron and elevator movement order to the simulator by sender interface. Meanwhile this autopilot subscribes the real-time state of the X-Plane to form a control loop through receiver. This autopilot will take the target roll angle and desired flight path angle and continuously accept the state feedback from simulator at the same time. Through shrinking the error between the real time state and reference state value to control the airplane to stablize around the reference state variable. Inside autopilot, the reference flight path angle and roll angle converted to aileron and elevator moving command to navigate the airplane to follow the reference path to move.

B. Mission Planner for the MASC framework

The SIL architecture for the MASC autopilot, shown in Figure 4, has four primary components. This section introduces i) nonlinear logic based mission planner, ii) proportional heading angle regulation scheme. The nonlinear guidance logic based mission planner includes a simplified particle model of the aircraft that is used for predicting the states of aircraft in a short horizon for trajectory planning connecting the initial coordinates of the aircraft right after engine failure (latitude x_0, longitude y_0, altitude z_0, heading ψ_0, forward speed V_0) to the desired final state at the selected landing site.

The nonlinear guidance logic is a waypoint following algorithm for calculating desired head angle based on real-time airplane position coordinates feedback. Since through moving aileron to approach the reference roll angle at each discrete time step is not straightforward. We design the fixed-wing heading controller converts the control targets from the heading angle to the roll angle. The mathematical model of the heading angle regulation scheme is based on Proportional controller.

C. Robustness Analysis And Verification

To validate the feasibility of online path planning and navigation framework MASC, we performed software in the loop simulation as follows. The aeroplane model in simulation is Cessna 172SP.

1. Simulation Configuration In Sunny Weather

The airspeed of the aircraft remains well above the stall speed, which is around 30 m/s. The positions of aircraft are in random initial conditions detailed in Table I. Throughout Simulation I, the weather condition is set to Clear,no wind, i.e., the best weather condition, in X-Plane®. The unified airport coordinate is North 21822m, East $-9751.8m$, Height 140m. The heading direction of the aircraft runway is 24.18deg.

Trial	North(m)	East(m)	Height(m)	Heading(Deg)
1	13163	-7164.9	3000	78.5
2	13353	-14380	4000	110.3
3	23429	-6675.6	5000	85.6
4	20719	-11652	3000	256.74
5	21323	-11021	2000	69.594

Table 1 Starting Positions With Engine Out

2. Result And Discussion

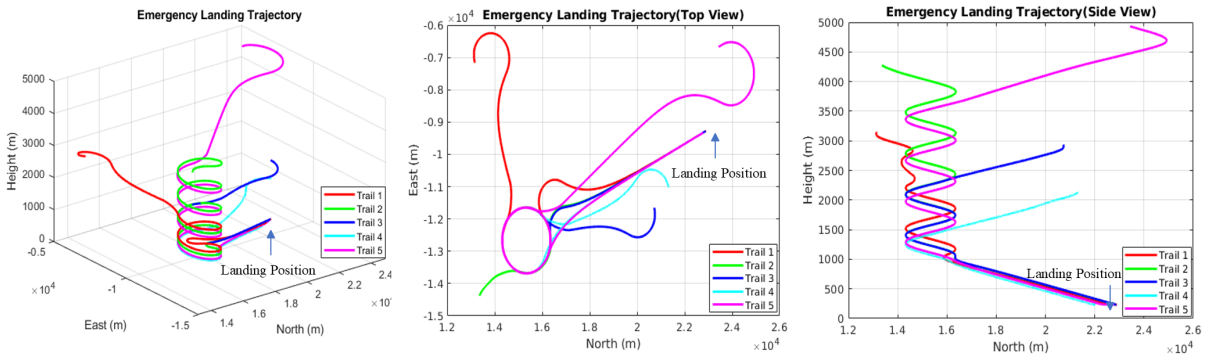

Fig. 5 Simulation I

The conduction of automatic landing process under sunny weather is shown here(simulation video: Landing Process Visualization). In this emergency landing simulation, we randomly initiate this process from five different positions. The final landing airport we configured in X-Plane is LaGuardia airport. The results for Simulation I can be seen in Figures 5 illustrating three different viewpoints of the real time landing trajectory. As we can see from the results, the MASC for emergency landing during engine failure is able to plan path online to safely navigate the aircraft to configured airport from any random initial conditions.

3. Simulation Configuration Under Windy Weather And Accompanied Turbulence

Windy Weather And Turbulence Simulation Configuration: sets the initial conditions of the aircraft always to that of the first trial in Table I. The unified airport coordinate is North 21822m, East $-9751.8m$, Height 140m. The heading direction of the aircraft runway is $24.18deg$. Then for each trial in simulation II, various wind conditions, various turbulence, from mild to severe, are set in X-Plane® according to Table II.

Trial	Wind Direction(deg)	Wind Speed(kts)	Turbulence(deg)	Gust Speed Increase(kts)	Total Wind Shear(deg)
1	20	14	10	10	10
2	0	3	8	14	8
3	4	5	10	8	14
4	14	7	12	22	8
5	27	12	4	9	4

Table 2 Weather Condition

4. Result And Discussion

The conduction of automatic landing process under windy weather plus turbulence is shown here (simulation video: Landing Process Visualization)

To further demonstrate the robust performance of MASC under large wind and turbulence uncertainties and generalize it in varied weather conditions, we implement the emergency landing task in different severe weather conditions listed in Table II. We configure the Wind direction, Wind speed, Turbulence, Gust speed increase, and Total wind shear randomly in high fidelity environment to approximate the real flight weather condition before conducting the landing task. It is hard to put the airplane always in the same start position for each trial. So, we assume every start coordinate of the landing procedure is the same and also ignore the small difference of start position. Figures 6 includes the real-time trajectory that illustrates Landing Procedure during Simulation II. As can be seen in the results, MASC can

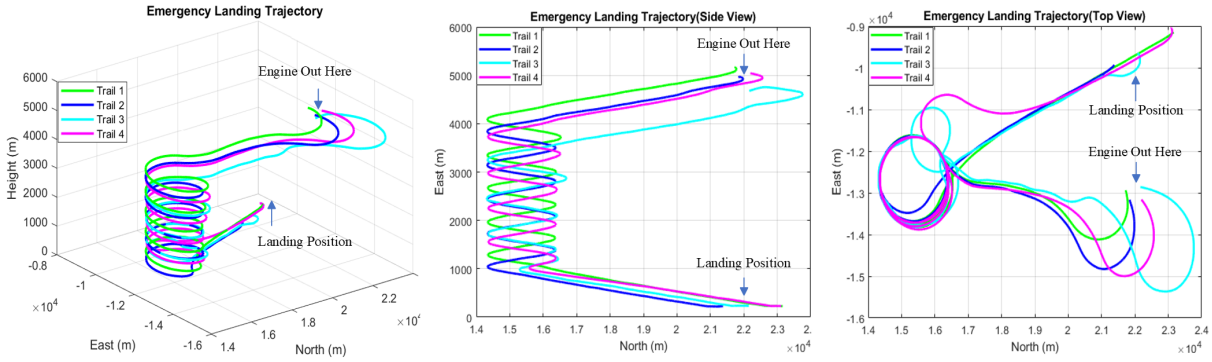

Fig. 6 Simulation II

navigate the engine out airplane to the configured airport in each running test in severe weather. So, MASC is capable to plan online and navigate the engine out airplane to land safely accompanied large wind and turbulence uncertainties.

IV. Offline Path Planning and Airport Selection

This section presents offline path planning module to validate the proposed MASC framework for feasibility evaluation of reachable areas for the emergency landing in an engine out scenario. It includes a programmed carrot chasing algorithm, a Fixed Wing heading controller and a UAV Guidance model. An emergency flight management architecture was presented in Figure 7 to compute feasible emergency landing trajectory to evaluate feasibility of each airport in response to the degrade engine preference in real time. The control laws are implemented in MATLAB/Simulink. This implementation is for mission planner to evaluate the feasibility of back up landing positions and then conduct general planning in regular flight.

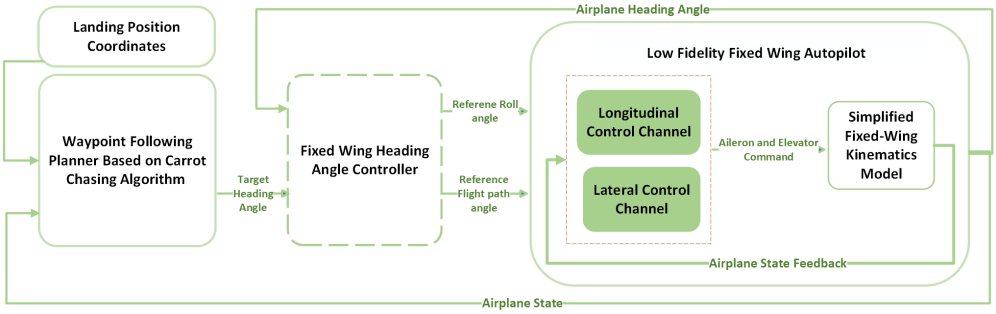

Fig. 7 MASC Autopilot(Offline Landing Path Prediction)

A. Safe landing trajectory generation

We programmed a path following algorithm based on the carrot chasing method in the very left block to calculate the look-ahead point, the desired course desired heading angle, and look ahead flag from the current UAV position using the pose, the set of waypoints, and the look-ahead distance. An embed a mathematical model is designed for discretizing land trajectory to serve mission planner. The mission planner accepts a set of waypoints of reference trajectory to calculate the expected path for feasible landing site. The Look ahead distance is a positive scalar that controls how closely the UAV follows the path. A smaller look ahead distance improves path tracking but can lead to oscillations in the path. The global coordinates of engine out position in simulation is North $18821m$, East $-17850m$, Height $2038m$. Configured landing position coordinated is airport in Table 1 labeled 1. The red dash line shown in Figures 8 is path planned offline.

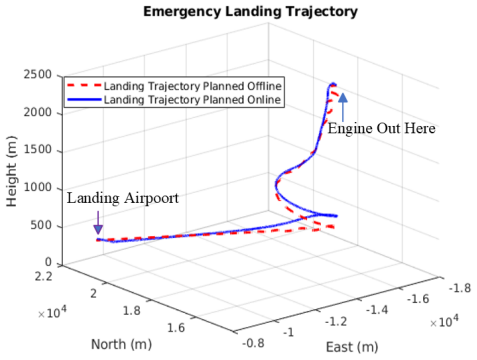

Fig. 8 Online And Offline Trajectory Comparison

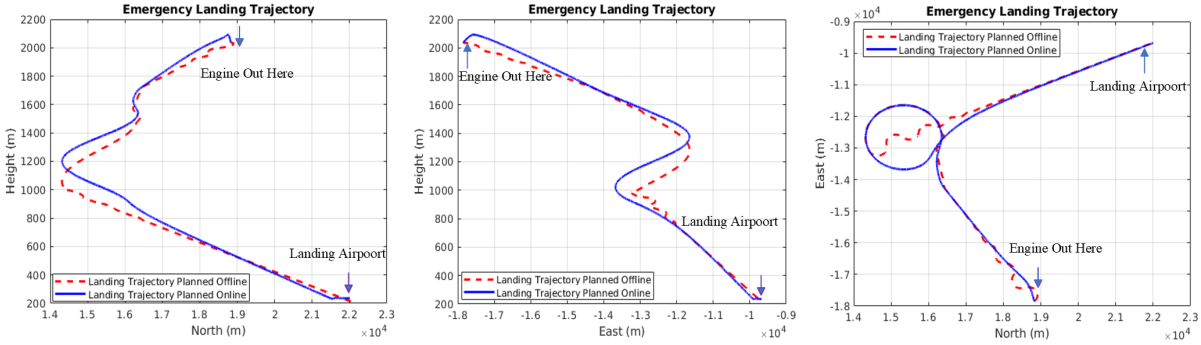

Fig. 9 Trajectory Comparison from different view

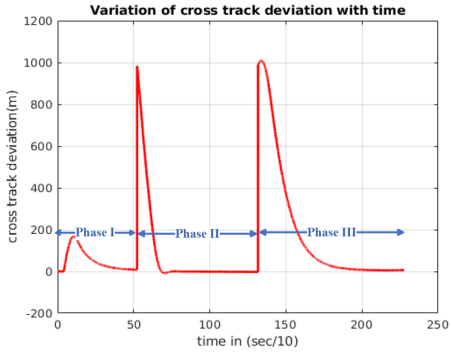

Fig. 10 Cross Tracking Error for Phase I II III

B. Comparison between the trajectory planned online and offline

We make a comparison between the planned path online and offline. Both simulations are configured with unified landing airport coordinates labeled 1 in software in the loop simulation and offline path planning. The trajectory planned offline coincides with planned online displayed in Figures 8. The Figures 9 shows those trajectory comparisons of the main view, left view, and the top view from left to the right. Due to less turns needed for an airplane in real-time simulation to match the heading direction of the reference straight line than offline path planning, the path planned offline for phase 1 can not entirely match the trajectory planned online at the start. However, the flight path will finally converge to the reference path to gradually approach to loiter center.

There is some difference between the position where simplified particle in simulation with low fidelity autopilot in use and airplane in simulation with high fidelity autopilot in use start to move along the loiter path in phase 2. The

transferring position where the status change from two to three is not totally matched in online planned path and offline planned path. However, Those unmatched positions where status transfer in path planning online and offline will not negatively impact the effectiveness of the carrot chasing method for navigating the airplane to move along the reference path in real-time. Autopilot of MASC framework will finally guide the assumed particle in low fidelity simulation and airplane in simulator converge to the reference straight line in phase 3. The loiter trajectories planned online and offline of phase 2 are matched well if it is observed from the top view shown in Figures 9.

The landing time in real-time path planning and the following is $10min$. The estimated landing time is 6 minutes by running offline path planning in acceleration mode in Matlab/Simulink. The simplified dynamics model-based landing time estimation is the same as the time of the real-time simulation in need, which can be used to predict the time of guiding an engine out airplane to conduct the emergency landing task.

The cross tracking error in three different phases shown in Figure 10 can approach zero respectively, which means the high fidelity autopilot can navigate the engine out airplane to move along the reference path in each phase. The trajectory tracking accuracy can meet the qualification.

C. Landing sites selection case study

In this section, we demonstrate the capability of the MASC framework for selecting the optimal feasible airport with the reference of assumed landing time. This is offline airport selection module which belongs to MASC Framework. This feasible landing selection scheme includes the offline path plan branch, landing time estimation branch and optimal landing site selection branch. Planning a reference landing trajectory for each backup airport happens before implementing the real-time emergency landing task. To pick the most feasible airport for an engine-out airplane to land, The MASC framework plans several landing trajectories offline for each backup airport and estimate the expected landing time. This airport selection module of the MASC framework needs the input of the several runway directions and airport coordinates listed in Table 3.

We build the mathematical model for getting the discretized reference trajectory point. Offline trajectory architecture subscribes these path point to predict the landing path and expected landing time approaching to each landing site. To avoid the unnecessary calculation when selecting feasible and optimal airport, we do not adopt the high density of reference trajectory point. This airport selection module is fast calculation and light weighted since build duration of entire path planning offline procedure needs $0h$ $0m$ $3.083s$ in average to finish. The model needed compile time differs varied computer configuration. We run this offline path planning procedure in acceleration mode in Matlab/Simulink. So Landing selection module is not computationally expensive.

Landing Coordinates	North(m)	East(m)	Height(m)	Runway Direction(Deg)	Landing Time(s)
1	21822	-9751.8	235	24.17	650
2	11822	-6751.8	235	130	700
3	46000	-39751.8	235	40	N/A
4	36000	-19751.8	235	40	800

Table 3 Backup Landing Coordinates

D. Result And Discussion

In this simulation, We configure four different airports to conduct selection. From the Figures 11, airports labeled 1,2,4 in table 3 are approachable. For the Airport labeled 3, the airplane crashed before transferring status 2 due to the long distance of the loitering center and the position where the airplane is engine-out. So the farthest airport 3 is not approachable. To land at the airport labeled two in table 1 needs 600 s, which is the shortest time. The whole emergency landing task needs 10 min in software in the loop simulation. So, this low fidelity model can be used to predict the actual time in need of an emergency landing process. Under the assumption of the average flying velocity, The landing trajectory for airport 1 is also the shortest. Figures 11 shows the axonometric drawing of the planned trajectories for matching airports. Figures 12 display the main view, left view, and top view of those paths. The position labeled by the purple arrow is the landing airport. The blue one means engine out position.

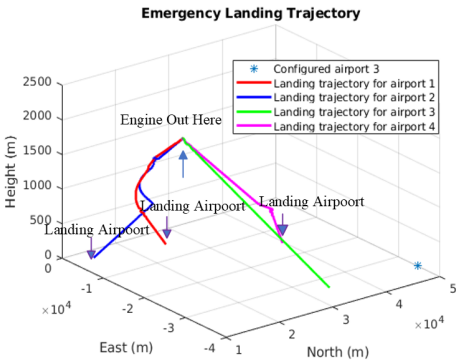

Fig. 11 Trajectories Planned offline

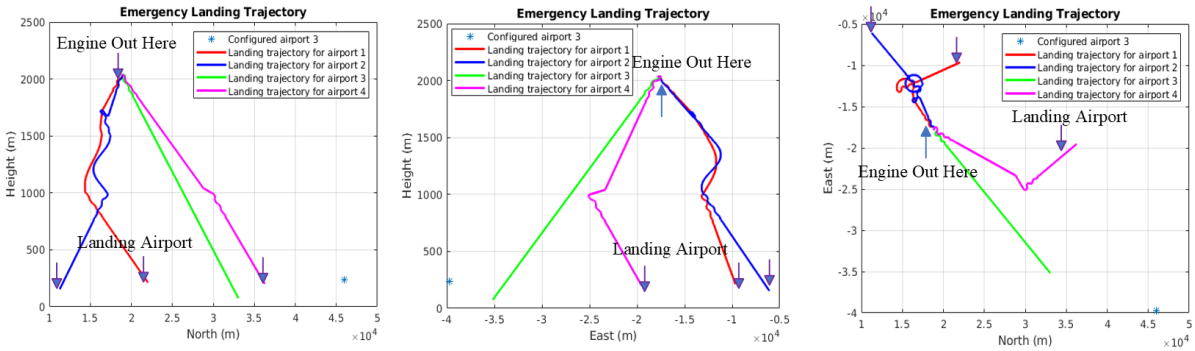

Fig. 12 Trajectories Planned offline from different view

V. Conclusion

In this paper, we proposed a novel and lightweight location based guidance architecture, Virtual Sully, which enables UAVs to perform automatically safety navigation for landing under large uncertainties. Our framework can also be applied into development and test of any path planning algorithm for ground mobile robot. Within this framework, a Multi-Level Adaptation approach in mission planning, tracking, and stabilizing control was presented in the context of engine-out emergency landing under severe weather conditions. Using a high-fidelity simulation environment, the effectiveness of the approach is verified by successful emergency landings of a full-scale fixed-wing aircraft under engine failure for a wide range of initial aircraft states and weather uncertainties. Especially after diagnosing broken engine, the subsystem for estimating remaining power integrated in this MASC is capable to evaluate the feasibility and select an optimal landing site in several back up landing areas.

In the future, we plan to address other challenges within the Virtual Sully Autopilot Framework such as vision based landing guidance and navigation[31]. We will also extend the framework to handle obstacle avoidance due to flying over a complex urban area and even malicious attacks. Also, we plan to involve more advance control theory to the autopilot controller design to improve the system robustness and adaptability. In this paper, rudder control has not been used. In the future, we plan to incorporate the rudder control to track a desired angle of sideslip and provide the fixed-wing aircraft with more agility and maneuvering performance.

References

[1] "Sullenberger Made the Right Move, Landing on the Hudson," , [posted 05-May-2010]. URL `https://www.wired.com/2010/05/ntsb-makes-recommendations-after-miracle-on-the-hudson-investigation/`.

[2] Sha, L., "Dependable system upgrade," *Real-Time Systems Symposium, 1998. Proceedings. The 19th IEEE*, IEEE, 1998, pp. 440–448.

[3] Sha, L., "Using Simplicity to Control Complexity," *IEEE Software*, Vol. 18, No. 4, 2001, pp. 20–28. https://doi.org/10.1109/MS.2001.936213.

[4] Crenshaw, T. L., Gunter, E., Robinson, C. L., Sha, L., and Kumar, P., "The simplex reference model: Limiting fault-propagation due to unreliable components in cyber-physical system architectures," *The 28th IEEE International Real-Time Systems Symposium*, 2007, pp. 400–412.

[5] Wang, X., Hovakimyan, N., and Sha, L., "L1Simplex: fault-tolerant control of cyber-physical systems," *Proceedings of the ACM/IEEE 4th International Conference on Cyber-Physical Systems*, ACM, 2013, pp. 41–50.

[6] Yoon, M.-K., Liu, B., Hovakimyan, N., and Sha, L., "VirtualDrone: Virtual Sensing, Actuation, and Communication for Attack-Resilient Unmanned Aerial Systems," *Proceedings of the ACM/IEEE International Conference on Cyber-Physical Systems*, 2017.

[7] D'Andrea, R., "Guest Editorial Can Drone Deliver?" *IEEE Transactions on Automation Science and Engineering*, Vol. 11, No. 3, 2014, pp. 647–648.

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

[9] Lin, C. A., Shah, K., Mauntel, L. C. C., and Shah, S. A., "Drone Delivery of Medications: Review of the Landscape and Legal Considerations," *American Journal of Health-System Pharmacy*, Vol. 75, No. 3, 2018, pp. 153–158.

[10] Jourdan, D., Piedmonte, M., Gavrilets, V., Vos, D., and McCormick, J., "Enhancing UAV survivability Through Damage Tolerant Control," *AIAA Guidance, Navigation, and Control Conference*, 2010.

[11] Ayhan, B., Kwan, C., Budavari, B., Larkin, J., and Gribben, D., "Path planning for UAVs with engine failure in the presence of winds," *IECON 2018-44th Annual Conference of the IEEE Industrial Electronics Society*, IEEE, 2018, pp. 3788–3794.

[12] Atkins, E. M., Portillo, I. A., and Strube, M. J., "Emergency Flight Planning Applied to Total Loss of Thrust," *Journal of Aircraft*, Vol. 43, No. 4, 2006, pp. 1205–1216. https://doi.org/10.2514/1.18816, URL https://doi.org/10.2514/1.18816.

[13] Atkins, E., "Emergency Landing Automation Aids: An Evaluation Inspired by US Airways Flight 1549," *AIAA Infotech@ Aerospace 2010*, 2010, p. 3381.

[14] Asadi, D., Sabzehparvar, M., Atkins, E. M., and Talebi, H. A., "Damaged airplane trajectory planning based on flight envelope and motion primitives," *Journal of Aircraft*, Vol. 51, No. 6, 2014, pp. 1740–1757.

[15] Bayen, A. M., Mitchell, I. M., Osihi, M. K., and Tomlin, C. J., "Aircraft Autolander Safety Analysis Through Optimal Control-based Reach Set Computation," *Journal of Guidance, Control, and Dynamics*, Vol. 30, No. 1, 2007, pp. 68–77.

[16] Choudhury, S., Scherer, S., and Singh, S., "RRT*-AR: Sampling-based alternate routes planning with applications to autonomous emergency landing of a helicopter," *2013 IEEE International Conference on Robotics and Automation*, IEEE, 2013.

[17] Sláma, J., "Emergency Landing Guidance for an Aerial Vehicle with a Motor Malfunction," 2018.

[18] Chen, Y.-B., Luo, G.-C., Mei, Y.-S., Yu, J.-Q., and Su, X.-L., "UAV path planning using artificial potential field method updated by optimal control theory," *Int. J. Syst. Sci.*, Vol. 47, No. 6, 2016, pp. 1407–1420.

[19] Dai, J., Wang, Y., Wang, C., Ying, J., and Zhai, J., "Research on hierarchical potential field method of path planning for UAVs," *2018 2nd IEEE Advanced Information Management,Communicates,Electronic and Automation Control Conference (IMCEC)*, IEEE, 2018.

[20] Budiyanto, A., Cahyadi, A., Adji, T. B., and Wahyunggoro, O., "UAV obstacle avoidance using potential field under dynamic environment," *2015 International Conference on Control, Electronics, Renewable Energy and Communications (ICCEREC)*, IEEE, 2015.

[21] Hwang, I., Kim, S., Kim, Y., and Seah, C. E., "A survey of fault detection, isolation, and reconfiguration methods," *IEEE transactions on control systems technology*, Vol. 18, No. 3, 2009, pp. 636–653.

[22] Gao, Z., Cecati, C., and Ding, S. X., "A survey of fault diagnosis and fault-tolerant techniques—part I: Fault diagnosis with model-based and signal-based approaches," *IEEE Trans. Ind. Electron.*, Vol. 62, No. 6, 2015, pp. 3757–3767.

[23] Jafarnejadsani, H., Lee, H., and Hovakimyan, N., "L1 adaptive sampled-data control for uncertain multi-input multi-output systems," *Automatica*, Vol. 103, 2019, pp. 346–353.

[24] Cao, C., and Hovakimyan, N., "Design and Analysis of a Novel $\mathcal{L}_1$ Adaptive Control Architecture with Guaranteed Transient Performance," *IEEE Transactions on Automatic Control*, Vol. 53, No. 2, 2008, pp. 586–591.

[25] Hovakimyan, N., and Cao, C., $\mathcal{L}_1$ *Adaptive Control Theory:Guaranteed Robustness with Fast Adaptation*, Society for Industrial and Applied Mathematics, Philadelphia, PA, 2010.

[26] Cao, C., and Hovakimyan, N., "$\mathcal{L}_1$ adaptive output-feedback controller for non-strictly-positive-real reference systems: Missile longitudinal autopilot design," *Journal of guidance, control, and dynamics*, Vol. 32, No. 3, 2009, pp. 717–726.

[27] Sujit, P., Saripalli, S., and Sousa, J. B., "Unmanned aerial vehicle path following: A survey and analysis of algorithms for fixed-wing unmanned aerial vehicless," *IEEE Control Systems Magazine*, Vol. 34, No. 1, 2014, pp. 42–59.

[28] Sujit, P. B., Saripalli, S., and Sousa, J. B., "An evaluation of UAV path following algorithms," *2013 European Control Conference (ECC)*, IEEE, 2013.

[29] Park, S., Deyst, J., and How, J., "A New Nonlinear Guidance Logic for Trajectory Tracking," *AIAA Guidance, Navigation, and Control Conference and Exhibit*, American Institute of Aeronautics and Astronautics, 2004. https://doi.org/10.2514/6.2004-4900, URL https://doi.org/10.2514/6.2004-4900.

[30] Bemporad, A., and Morari, M., "Robust model predictive control: A survey," *Robustness in identification and control*, Springer, 1999, pp. 207–226.

[31] Cesetti, A., Frontoni, E., Mancini, A., Zingaretti, P., and Longhi, S., "A Vision-Based Guidance System for UAV Navigation and Safe Landing using Natural Landmarks," *Journal of Intelligent and Robotic Systems*, Vol. 57, No. 1-4, 2009, pp. 233–257. https://doi.org/10.1007/s10846-009-9373-3, URL https://doi.org/10.1007/s10846-009-9373-3.