Online Learning and Planning in Time-Varying Environments: An Aircraft Case Study

Gokul Puthumanaillam* Yuvraj Mamik† and and Melkior Ornik‡
University of Illinois Urbana-Champaign, Urbana, USA

Aerospace vehicles routinely encounter uncertain, time-varying, and partially observable environments, presenting considerable challenges for autonomous operation and planning. Traditional learning methods, which excel in static contexts, often falter in such highly dynamic settings. Building on recently established Time-Varying Partially Observable Markov Decision Processes (TV-POMDP) and Memory Prioritized State Estimation (MPSE) methodologies, this work demonstrates their application in the advanced GUAM simulation environment, which models NASA's Generic UAM concept. The contribution of this paper lies in refining these approaches to suit the complexity and unpredictability of aerospace contexts, where conventional learning strategies are insufficient. By applying MPSE, we enhance the estimation of environmental states with a weighted approach that respects the temporality and informational value of observations. The subsequent policy optimization process is informed by the estimations of these time-varying transition functions, leading to better long-term strategies that are aware of the rapid environmental shifts characteristic of aerospace scenarios. The validation of these methods through the GUAM simulator confirms their effectiveness, marking a positive step towards their practical implementation in autonomous aerospace vehicles that encounter continual, stochastic changes.

I. Nomenclature

S	=	finite set of states
A	=	finite set of actions
Z	=	finite set of observations
T	=	state transition probability function
O	=	observation function
R	=	reward function
$b(s)$	=	belief state
V	=	value function
T_t	=	time-varying transition probability function
A_s	=	autocorrelation score
R_s	=	recency score
D_s	=	deviation score
p	=	position of the UAM in Earth-fixed coordinates
Θ	=	orientation of the UAM
ω	=	angular velocity components about the body axes of the UAM
u	=	thrust force generated by the propulsion system
δ	=	deflection angles for the ailerons, elevator, and rudder
Ω	=	rotor velocity
p_{gps}	=	GPS coordinates
z_{alt}	=	altitude measurement from altimeter
ω_{imu}	=	gyroscope readings

*Department of Aerospace Engineering and Coordinated Science Laboratory, University of Illinois Urbana-Champaign, Urbana, USA. Email: gokulp2@illinois.edu

†Department of Aerospace Engineering, University of Illinois Urbana-Champaign, Urbana, USA. Email: ymamik2@illinois.edu

‡Department of Aerospace Engineering and Coordinated Science Laboratory, University of Illinois Urbana-Champaign, Urbana, USA. Email: mornik@illinois.edu

II. Introduction

Almost by definition, aerospace vehicles regularly operate in exceptionally challenging conditions. They are required to complete complex missions with little margin for error, and might do so in remote, uncertain, or hostile environments where safety is critical and there is little opportunity for mid-mission repair or a mission repeat. While autonomy offers an opportunity to reduce human risk and workload in these conditions, autonomous planning in such environments faces significant technical obstacles. In contrast to humans, who can often rely on their experience to intuitively respond to novel events, mission specifications, and environments, autonomous strategies necessarily depend on some combination of *robustness* –implementation of often-conservative strategies designed to work across uncertain system characteristics – and *learning* i.e., identification and adaptation to novel characteristics. Yet, robustness is only possible if the uncertainty is not excessive: if the environment changes sufficiently, the vehicle might have to abandon its prior strategy to complete its mission. On the other hand, learning is difficult in scenarios where an environment continually changes and thus can only provide a limited amount of relevant data.

The proposed paper aims to tackle planning for aerospace systems in exactly such — possibly stochastic, time-varying, a priori unknown — environments. To capture relevant features of complex environments, the technical content of this paper will consider optimal planning for unknown *time-varying*, possibly *partially observable*, *Markov decision processes (TV-POMDP)*. Learning and subsequent planning for unknown *time-invariant Markov Decision Process* is a classical problem in autonomy [1–3]. Relevant work [3–5] naturally relies on one of two mechanisms: (i) offline learning, in which data about the system is collected through repeated experimentation and subsequently analyzed to provide an optimal policy for future mission execution, and (ii) online learning, in which data is collected during the mission and the control policy is adapted on the fly based on the arriving data. Importantly, in order to collect enough data to learn, both of these frameworks necessarily assume that the environment remains constant during experimentation and mission execution. If operated in a time-varying environment, standard learning strategies will implicitly interpret change over time as "noise" and instead learn an optimal policy with respect to some imaginary, "average" time-invariant environment [6]. Instead of employing methods unaware of the environmental change over time, the theoretical foundation of this paper draws from emerging work on *Memory Prioritized State Estimation* (MPSE).

MPSE hinges on the understanding that in an environment characterized by incremental change, historical data still holds considerable value to learn the transition function — there exists a substantial correlation between past observations, recent observations and the current system state. To make use of this data, the methodology formulates an optimization problem which seeks to find current and past environment parameters which are maximally likely given all the collected data, constrained by the known *maximal possible change* in these parameters. The MPSE method has been previously used to develop learning and planning mechanisms for standard POMDPs and TV-POMDPs [7]. While there have been limited prior numerical experiments in aerospace domains — including a simple wind flow estimation model and air density estimation for hypersonic vehicles — the MPSE-based work has now matured enough to yield results in a high-fidelity, autonomy-driven aerospace environment. This paper seeks to introduce the MPSE-based methods to the aerospace decision-making community through an application to NASA's Generic UAM Simulation (GUAM) aircraft model, emulating a NASA Lift+Cruise concept vehicle configuration [8]. The paper showcases a significant step forward towards adoption of the MPSE approach in aerial autonomy: successful on-the-fly learning and control for a simulated autonomous aerial vehicle operating in a complex, time-varying and previously unknown environment with partial state observability.

III. Technical Background

A. Partially Observable Markov Decision Processes

Partially Observable Markov Decision Processes (POMDPs) provide a mathematical framework for modeling decision-making in systems where the agent's perception of the environment is limited and uncertain. A POMDP is formally defined as a tuple $(S, A, Z, T, O, R, \gamma)$ where:

- S is a finite set of states representing the possible configurations of the environment.
- A is a finite set of actions available to the decision-making agent,
- Z is a finite set of observations that the agent can perceive,
- $T : S \times A \times S \to [0, 1]$ is the state transition probability function, with $T(s'|s, a)$ denoting the probability of the agent transitioning to state s' from state s after taking action a,
- $O : S \times A \times Z \to [0, 1]$ is the observation function, where $O(z|s', a)$ represents the probability of the agent receiving observation z when action a results in state s',

- $R : S \times A \rightarrow \mathbb{R}$ is the reward function, which assigns a numerical reward to each action taken in a given state,
- $\gamma \in [0, 1)$ is the discount factor, which represents the difference in importance between future rewards and immediate rewards.

The goal in a POMDP is to find a policy $\pi : H \rightarrow A$, mapping histories H of states, actions, and observations to actions, that maximizes the expected cumulative discounted reward. This objective is expressed as

$$\max_{\pi} \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t) \mid \pi\right], \tag{1}$$

where s_t and a_t are the state and action at time t, respectively.

Finding the optimal policy in equation 1 is challenging due to the exponential growth in computational resources required as the number of dimensions (state variables) in the problem increases. In stochastic and partially observable environments, an agent's direct observation of the true state is often obscured or incomplete. In stochastic and partially observable environments, an agent's direct observation of the true state is often obscured or incomplete. The agent must maintain a belief state $b(s)$, a probability distribution over S, updated after each action and observation according to Bayes' rule:

$$b'(s') = \frac{O(z|s', a) \sum_{s \in S} T(s'|s, a) b(s)}{\sum_{s'' \in S} O(z|s'', a) \sum_{s \in S} T(s''|s, a) b(s)} \tag{2}$$

The optimal policy in a TV-POMDP setting cannot be explicitly found due to the computational issues mentioned earlier. These challenges make it impractical to enumerate and evaluate all possible belief states and their corresponding action sequences. Instead, approximation methods such as point-based value iteration or Monte Carlo sampling are employed, which manage the complexity by focusing on a subset of belief points or simulations of the environment.

B. Time-Varying Partially Observable Markov Decision Processes

The assumption of stationary dynamics within the traditional POMDP framework limits its applicability in real-world scenarios, where environmental conditions are subject to constant change. As underscored by [9], the challenge in adapting POMDPs to non-stationary contexts lies in the temporal aspect of the environment. Time is unidirectional and inherently linked to the changing state of the world; incorporating it into the state space leads to two main bottlenecks: an increase in its size, and consequently, the computational complexity and secondly, not more than one observation sample can be collected in each time step. These challenges render traditional learning based approaches infeasible for time-varying applications, particularly when decision-making processes must be executed on-the-fly in dynamic environments.

Time-Varying POMDPs (TV-POMDPs) are proposed as an elegant way to encode timporal variation. By conceptualizing transition probabilities as time-varying functions, TV-POMDPs capture the essence of environmental dynamism without necessitating a direct increase in the dimensionality of the state space. The modified framework, represented as $(S, A, Z, T_t, O, R_t, \gamma, b_0)$, thus includes:

- A time-varying transition probability function $T_t : S \times A \times S \times \mathbb{N} \rightarrow [0, 1]$, which maps a state-action pair and a discrete time index to a probability distribution over subsequent states, reflecting the probabilistic nature of environmental changes over time.
- A time-varying reward function $R_t : S \times A \times \mathbb{N} \rightarrow \mathbb{R}$, which adapts the immediate reward associated with actions taken in specific states at different times, accommodating the shifting objectives or costs that can occur as the environment evolves.

To operationalize the objective function described in equation (1) in a dynamic, time-varying environment, we introduce the value function $V_t(b)$, which encapsulates the expected future rewards from a given belief state b at time t. The value function is recursively defined to account for both immediate rewards and the future benefits of actions taken in the current belief state. It is expressed as:

$$V_t(b) = \max_{a \in A} \left\{ \sum_{s \in S} b(s) \left[R_t(s, a, t) + \gamma \sum_{s' \in S} T_t(s, a, s', t) V_{t+1}(b') \right] \right\}, \tag{3}$$

where $V_{t+1}(b')$ represents the value of the subsequent belief state b', taking into consideration the time-varying transitions T_t and the instantaneous reward R_t.

At each time step t, the policy π is derived by choosing the action a that maximizes the immediate reward plus the discounted value of future rewards, as estimated by the value function.

C. GUAM Simulator

GUAM, as a simulation tool, provides an essential platform for validating our control model through extensive data generation and analysis. GUAM is a comprehensive simulation architecture [10] employed for spacecraft and aircraft models. It incorporates both medium and high fidelity aero-propulsive models, namely the Polynomial Model [11]. These models are instrumental in simulating the aerodynamic characteristics, providing a detailed understanding of the forces and moments acting upon them.

The simulation requires specific inputs to accurately model the aircraft's dynamics:

- Reference trajectory data: In order to generate a trajectory for the AAM, we feed in the reference trajectory that needs to be followed. This encompasses the heading frame velocity, inertial positions, and heading angles, which are critical for defining the desired flight path of the vehicle.
- Environment and turbulence models: Environment models controls the effect of the atmospheric conditions, including wind and turbulence in the simulation.
- Aircraft model parameters: These parameters covers aspects such as lift and cruise configurations, weight, center of gravity, and actuator types. Each of these parameters are meticulously calibrated to reflect the design and performance characteristics of the Advanced Air Mobility (AAM) vehicle under study.

IV. Proposed Methodology

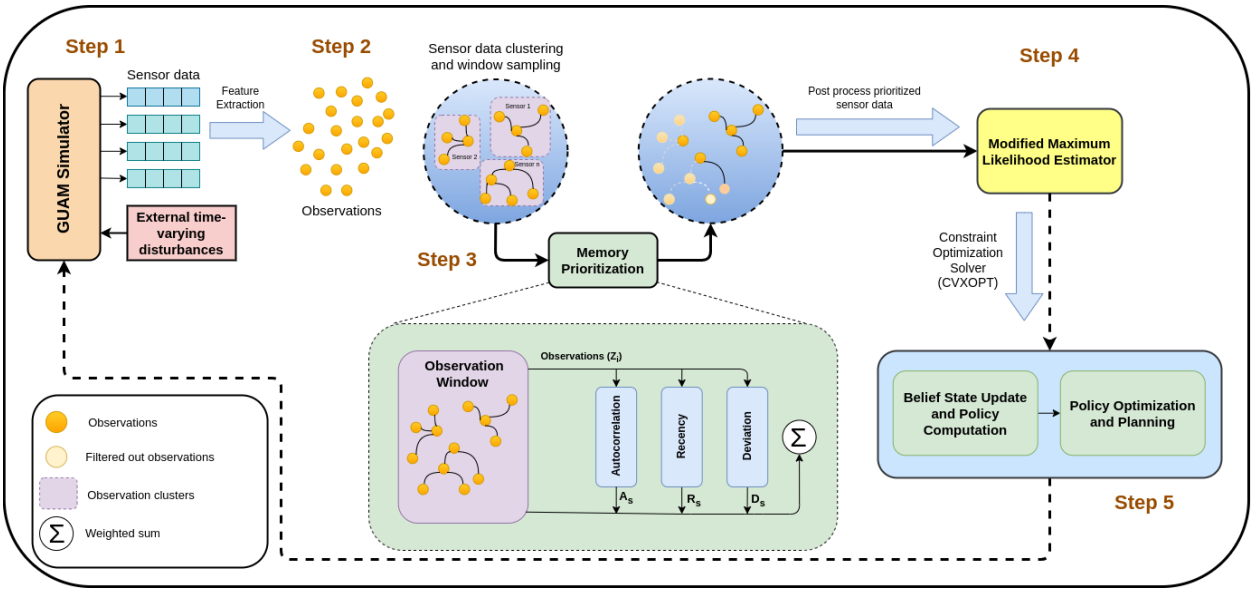

Fig. 1 Memory Prioritized State Estimation adapted for the GUAM framework

The proposed methodology, as illustrated in Figure 1, presents a comprehensive approach adapted to the GUAM environment, specifically tailored to address the challenges of autonomous navigation of AAMs in time-varying conditions. This methodology integrates several key components: data collection and feature extraction, clustering and window sampling of sensor data, memory prioritization, and a modified approach to Maximum Likelihood Estimation (MLE). Each of these components plays a critical role in the estimation of the time-varying transition probability function (T_t) in a TV-POMDP framework. The architecture diagram (Figure 1) encapsulates the workflow and interdependencies of these components, providing a visualization of how the system processes data, makes decisions, and adapts to environmental changes. The subsequent subsections delve into each component in detail, elucidating their individual contributions to the overall efficacy of the proposed approach.

A. Data Collection and Feature Extraction

The GUAM Simulator is used to record a multidimensional time series data from the sensors $\mathcal{X} \in \mathbb{R}^{m \times p}$, where m represents the number of time steps, and p is the number of sensor measurements at each time step. The raw sensor data matrix $\mathcal{X}$ is preprocessed to center the variables by subtracting the mean of each column from the corresponding sensor measurements.

Principal Component Analysis (PCA) [12] is then applied to reduce the dimensionality of the data while retaining most of the variance. The PCA transformation is achieved through the eigen-decomposition of the covariance matrix C of the centered data:

$$C = \frac{1}{m-1}\mathcal{X}^\top \mathcal{X}. \tag{4}$$

The eigenvalues λ_i and eigenvectors $\sqsubseteq_i$ of C are computed, and the eigenvectors are ordered by decreasing eigenvalues. The top k eigenvectors $\mathcal{V}_k = [\sqsubseteq_1, \sqsubseteq_2, \ldots, \sqsubseteq_k]$ corresponding to the largest k eigenvalues are selected to form the projection matrix.

The feature set $\mathcal{F}$ is obtained by projecting the centered data $\mathcal{X}$ onto the subspace spanned by $\mathcal{V}_k$:

$$\mathcal{F} = \mathcal{X}\mathcal{V}_k. \tag{5}$$

Each row in $\mathcal{F}$ represents the extracted features for a given time step in a reduced k-dimensional space, where $k \ll p$.

This preprocessing of raw observations helps us condense the high-dimensional sensor data into a more manageable form, reducing computational complexity and focusing on the most informative aspects of the data. Secondly, in the context of Memory Prioritized State Estimation (MPSE) and Partially Observable Markov Decision Processes (POMDPs), a more compact and essential representation of the state space is crucial for efficient state estimation and decision-making processes. By distilling the sensor data into principal components, we ensure that the most significant features influencing the agent's state are retained while discarding redundant or less informative data.

B. Sensor Data Clustering and Window Sampling

The sensor data is clustered to identify patterns and relationships within the observations. Doing so helps segment the data into meaningful groups, facilitating the identification of windows of observations, where a window refers to an interval of time containing multiple successive observations, that are likely to be more informative about the current system dynamics. With the feature-selected dataset $\mathcal{F}$, we apply k-means clustering algorithm [13] to segment the data into k clusters, where each cluster contains observations with similar features. The clustering can be mathematically represented as a function $\Psi : \mathbb{R}^k \to \{1, \ldots, k\}$, which assigns each feature vector to one of the k clusters.

For each cluster C_i, we identify observation windows, W_i, based on temporal proximity and feature similarity. Mathematically, a window W_i is defined as a sequence of feature vectors $\mathcal{F}_i \subset \mathcal{F}$ that fall within a specific time range and cluster:

$$W_i = \{\mathcal{F}_i \mid \mathcal{F}_i = \mathcal{F}(t_s : t_e), \Psi(\mathcal{F}(t)) = C_i, \forall t \in [t_s, t_e] \subseteq \{1, \ldots, m\}\}. \tag{6}$$

These windows are then used to construct the temporally contextualized inputs for the memory prioritization step, ensuring that the dynamics of the environment are captured over different intervals for subsequent analysis.

Within each observation window, not every observation holds equal value for the estimation process. To efficiently allocate computational resources, we introduce a memory prioritization mechanism. It assigns weights to observations based on their autocorrelation, recency, and deviation – key indicators of informativeness in time-varying contexts.

C. Memory Prioritization

Memory prioritization effectively filters out less informative observations, reducing the computational load and improving the efficiency and accuracy of the estimation process. Given the feature set $\mathcal{F}$, we evaluate the temporal relevance and informativeness of each feature vector f_i within the window W_i. The assessment involves three key metrics:

1) *Autocorrelation*: This metric assesses the temporal dependency between observations, capturing the extent to which past states influence the current state. To measure the dependence of features, the autocorrelation for

a window W_i of features at lag k (where lag k refers to the time separation between the observations being correlated) is calculated as follows:

$$\rho_{W_i}(k) = \frac{\sum_{j=k+1}^{|W_i|}(f_j - \bar{f}_{W_i})(f_{j-k} - \bar{f}_{W_i})}{\sqrt{\sum_{j=1}^{|W_i|}(f_j - \bar{f}_{W_i})^2 \sum_{j=1}^{|W_i|}(f_{j-k} - \bar{f}_{W_i})^2}}, \tag{7}$$

where $\bar{f}_{W_i}$ is the mean feature vector in window W_i.

2) *Recency*: Observations lose relevance with time, which is quantified by the recency score R_s. R_{s_i} for a feature vector f_i is inversely proportional to the time elapsed since its observation:

$$R_{s_i} = \frac{1}{t_{\text{current}} - t_i + \varepsilon}, \tag{8}$$

where t_{current} is the current time step, t_i is the time step when f_i was observed, and ε is a small constant to avoid division by zero.

3) *Deviation*: The deviation score D_{s_i} quantifies how much the feature vector f_i deviates from the mean of its window:

$$D_{s_i} = \|f_i - \bar{f}_{W_i}\|, \tag{9}$$

where $\|\cdot\|$ denotes the Euclidean norm, and $\bar{f}_{W_i}$ is the mean feature vector for window W_i.

The weight ω_i assigned to each feature vector f_i combines these factors and is defined as

$$\omega_i = w_a A_{s_i} + w_r R_{s_i} + w_d D_{s_i}, \tag{10}$$

where $A_{s_i} = \rho_{W_i}(t_{\text{current}} - t_i)$ is the autocorrelation score for f_i, and w_a, w_r, and w_d are the weights corresponding to autocorrelation, recency, and deviation, respectively. These values are chosen based on the nature of the system. The aggregate weight ω_i serves as a heuristic for the feature vector's expected contribution to the accurate estimation of the state transition probabilities in the TV-POMDP framework.

D. Modified Maximum Likelihood Estimation

The Modified Maximum Likelihood Estimation (MLE) incorporates the weights from the memory prioritization step. The likelihood function is defined as $L(T_t|\mathcal{F}, \mathcal{A}, \Omega)$ over the feature set $\mathcal{F}$, action set $\mathcal{A}$, and weight set Ω, reflecting the probability of observing a particular sequence of features given a sequence of actions and the model parameters T_t:

$$L(T_t|\mathcal{F}_{0:T}, \mathcal{A}_{0:T}, \Omega_{0:T}) = \prod_{t=0}^{T} \left[\text{Prob}_{T_t}(f_t|a_t, \Omega_t)\right]^{\omega_t}, \tag{11}$$

where $\text{Prob}_{T_t}(f_t|a_t, \Omega_t)$ is the probability of observing feature vector f_t given action a_t and the model parameters at time t, weighted by ω_t from the set Ω. This modified MLE effectively integrates the sequential, weighted nature of the observations and actions within a dynamic environment, providing a computationally efficient approach to estimating the time-varying transition probabilities in the TV-POMDP framework.

To estimate T_t, the time-varying transition probability function, a constrained optimization problem is defined that maximizes the log-likelihood function subject to constraints on the maximal rate of change between successive time steps, denoted by $\Delta_{\max}$. The optimization problem is formulated as follows:

$$\begin{aligned} \hat{T}_t &= \arg\max_{T_t} \sum_{t=0}^{T} \omega_t \log \text{Prob}_{T_t}(f_t|a_t, \Omega_t) \\ &\text{subject to} \quad \|T_t - T_{t-1}\| \leq \Delta_{\max}, \quad \forall t \in \{1, \ldots, T\}, \end{aligned} \tag{12}$$

The reference to the work in [6] proves that this optimization problem is convex, which implies that it can be solved efficiently using optimization libraries. We used CVXOPT [14] as our solver. The result is an estimation of T_t that is not only informed by the most relevant and recent observations but also utilizes only the prioritized data thereby making it computationally efficient.

E. Policy Optimization and Planning

The optimal policy π^* seeks to maximize the expected cumulative discounted reward within the framework of the *estimated transition probabilities* $\hat{T}_t$. Doing so involves updating the belief state $b_t(s)$ at each time step t, which is a probability distribution over the state space S, based on the latest observations and actions. The updated belief state $b_{t+1}(s')$ is computed with the estimated $\hat{T}_t$ from equation (12) as follows:

$$b_{t+1}(s') = \eta \Omega(z_t|s', a_t) \sum_{s \in S} \hat{T}_t(s, a_t, s') b_t(s), \tag{13}$$

where $\Omega(z_t|s', a_t)$ denotes the observation likelihood of receiving z_t given the action a_t and the subsequent state s', and η is a normalizing constant ensuring that $b_{t+1}(s')$ forms a valid probability distribution.

The policy $\pi(b)$ is determined by maximizing the expected utility, which is computed through the value function $V_t(b)$ for the belief state b. This is formulated as:

$$\pi(b) = \arg\max_{a \in A} \left\{ \sum_{s' \in S} b(s) R_t(s, a) + \gamma \sum_{s' \in S} \hat{T}_t(s, a, s') V_{t+1}(b') \right\}, \tag{14}$$

where $R_t(s, a)$ is the immediate reward received after taking action a in state s at time t, and γ is the discount factor representing the trade-off between immediate and future rewards. This policy is recomputed at every time instant, adapting to the updates in the transition probability estimates. This optimization strategy also ensures that the policy is not only optimal with respect to the long-term rewards but also adaptive to the dynamic changes captured by the time-varying transition probabilities $\hat{T}_t$.

Having established the methodology, we now transition to an in-depth analysis of the system's performance.

V. Analysis

This section presents the empirical evaluation of our proposed methodology within a time-varying scenario. We consider a scenario where the GUAM agent is provided with 200 waypoints which track a reference trajectory. The goal of the agent is to track the trajectory as close as possible by following the waypoints. Adapted from [15], the experiment utilizes GUAM simulator's turbulence models to reproduce the effects of time-varying winds and aerodynamic disturbances.

A. Simulation Setup

The simulation setup employed the GUAM environment with the Lift+Cruise aircraft model within GUAM, designed to be representative of a NASA Lift+Cruise vehicle configuration [8].

1. State Space

The state space $\mathcal{S}$ is a detailed representation of the vehicle's kinematic and dynamic attributes and includes additional parameters pertinent to the operational health of the UAM. It is formally defined as:

$$\mathcal{S} = \{(p, \dot{p}, \Theta, \omega, \text{sys}) \mid p \in \mathbb{R}^3, \dot{p} \in \mathbb{R}^3, \Theta \in SO(3), \omega \in \mathbb{R}^3, \text{sys} \in \{0, 1\}^n\},$$

where:

- $p = (x, y, z) \in [-1500, 1500]^3$ denotes the position of the UAM in Earth-fixed coordinates.
- $\dot{p} = (\dot{x}, \dot{y}, \dot{z} \in [-50, 50]^3)$ represents the velocity vector in the respective axes.
- $\Theta = (\phi, \theta, \psi) \in [-\pi, \pi]^3$ denotes the orientation of the vehicle given by roll, pitch, and yaw angles.
- $\omega = (p, q, r) \in [-1, 1]^3$ represents the angular velocity components about the body axes.
- sys is a binary vector representing the status of the flight. A value of 0 represents the simulation ended due to the aircraft crashing.

The status of the flight $sys = 0$ leads to an immediate cessation of the simulation run. It is caused by one of the following reasons:

- Structural limits are exceeded, where the forces or torques acting on the vehicle surpass its physical tolerance, risking structural integrity.
- Boundary violations occur when the vehicle's position p exits the designated safe operational envelope $[-1500, 1500]^3$.

- Loss of control is detected, characterized by orientation angles Θ or angular velocities ω falling outside the operational range $[-\pi, \pi]^3$ and $[-1, 1]^3$, respectively.
- Collision with obstacles or terrain within the simulation environment is registered.
- System failures transpire, indicated by any element of the binary vector sys being set to 0, signifying critical malfunctions.
- Unsafe landing dynamics are encountered if the landing velocity $\dot{p}$ exceeds the thresholds for a controlled descent and touchdown.

2. Action Space

The action space $\mathcal{A}$ encompasses the set of controls the UAM can execute, directly influencing its state transition. This space is characterized by:

$$\mathcal{A} = \{(u, \delta, \Omega) \mid u, \Omega \in \mathbb{R}^+, \delta \in \mathbb{R}^3\},$$

where:

- $u \in [0, 200]$ represents the thrust force generated by the propulsion system.
- $\delta = (\delta_a, \delta_e, \delta_r) \in [-30°, 30°]^3$ comprises the deflection angles for ailerons, elevator, and rudder, respectively.
- Ω represents the rotor velocity.

3. Observation Model

The observation model $\mathcal{Z}$ captures the sensory feedback from the environment and the vehicle's state, which is partially observable due to sensor noise and limitations. It is expressed as:

$$\mathcal{Z} = \{(p_{gps}, z_{alt}, \omega_{imu} \mid p_{gps} \in \mathbb{R}^2, z_{alt} \in \mathbb{R}^+, \omega_{imu} \in \mathbb{R}^3\},$$

where:

- $p_{gps} = (x_{gps}, y_{gps}) \in [-1500, 1500]^2$ denotes the GPS coordinates.
- $z_{alt} \in [0, 500]$ represents the altitude measurement from the altimeter.
- $\omega_{imu} = (\omega_x, \omega_y, \omega_z) \in [-1, 1]^3$ captures the gyroscope readings which measure the angular velocity of the vehicle in three dimensions: roll, pitch, and yaw.

4. Reward Function

The reward function $\mathcal{R}$, for state-action pair (s_t, a_t) is structured to encourage optimal path following and task completion [16], defined as:

$$\mathcal{R}(s_t, a_t) = \begin{cases} +10, & \text{for each waypoint reached in the correct sequence within a 3m radius,} \\ +50, & \text{upon successful arrival at the final goal,} \\ -1, & \text{for every unit distance traveled, when not reaching a waypoint, to encourage efficient path following,} \\ -1000, & \text{if the UAM experiences system failure } (sys = 0), \\ 0, & \text{when not moving from its current position i.e., hovering.} \end{cases}$$

The reward values are quantified such that reaching waypoints and the final goal yields positive rewards, while inefficient trajectories or system failures incur penalties, promoting a balance between precision and expedience.

5. Time-varying disturbances:

The time-varying transition probability function, denoted by $T_t(s, a, s', t)$, quantifies the likelihood of transitioning from state s to state s' given action a at time t. In our simulation, we consider the environmental dynamics induced by aerodynamic disturbances, which are represented by a time-varying component of the transition probabilities. This component is modeled over the simulation time window $[0, 10]$ as:

$$T_t(s, a, s', t) = \frac{1}{1 + e^{-(t-5)}} \quad \text{for } 0 \leq t \leq 10, \tag{15}$$

where t is the discrete time step within the simulation timeframe. The subscript t in T_t emphasizes the dependency of the transition probabilities on the time step, reflecting the dynamic nature of the environment. Furthermore, the rate of change in the transition probabilities is constrained by $\Delta_{\max} = 0.2$:

$$|T_t(s, a, s', t) - T_{t-1}(s, a, s', t-1)| \leq \Delta_{\max}, \tag{16}$$

When an aerodynamic disturbance occurs, it causes the vehicle to deviate from its intended trajectory. The disturbance's impact is modeled as a deviation angle θ, representing the yaw response of the UAM to gusts of wind or turbulent airflow. While θ originates from a normal distribution due to the stochastic nature of wind disturbances, practical considerations require that this angle be bounded within the UAM's operational limits. Consequently, θ follows a truncated normal distribution $\theta \sim \mathcal{N}_{\text{trunc}}(0, \sigma^2, \theta_{\min}, \theta_{\max})$, where σ^2 denotes the variance of the wind disturbance effect, and $\theta_{\min} = -10^\circ$, $\theta_{\max} = 10^\circ$ are the lower and upper truncation points, respectively.

B. Baseline Comparisons

To validate the performance of our TV-POMDP framework, we compare it against three established baselines: model reference adaptive control (MRAC), robust control and the standard POMDP approach that considers the environment to be time-invariant and uses classical estimation techniques. All the baselines serve to benchmark the efficacy of our proposed method in handling complex, time-varying scenarios within the GUAM simulation environment.

1. MRAC Adaptive Controller for UAM

In the context of our UAM simulation, the MRAC adaptive controller is tailored to maintain the vehicle's trajectory within close proximity to a reference path, despite the unpredictable aerodynamic disturbances encountered. The reference model for the UAM is designed to represent an ideal flight path in an undisturbed environment.

The reference model dynamics are represented as:

$$\dot{x}_{\text{ref}}(t) = A_{\text{ref}} x_{\text{ref}}(t) + B_{\text{ref}} u_{\text{ref}}(t), \tag{17}$$

where $x_{\text{ref}}(t)$ is the state vector describing the desired position and orientation in the absence of disturbances, and $u_{\text{ref}}(t)$ is the control input vector that would maintain this desired state.

Given the aerodynamic nature of the disturbances, the MRAC is configured to adjust the control inputs to the UAM's propulsion and control surfaces to counteract the effects of wind gusts and turbulence. The control law, incorporating adaptive gains, is defined as:

$$u(t) = K_{\text{ad}}(t) x(t) + K_{\text{ref}} r(t), \tag{18}$$

where $K_{\text{ad}}(t)$ dynamically adjusts based on the observed state deviations due to disturbances, $x(t)$ is the current state of the UAM, and $r(t)$ is a reference signal aimed at guiding the UAM along the reference path.

The adaptive gains are updated by the rule:

$$\dot{K}_{\text{ad}}(t) = -\Gamma \left(e(t) x(t)^\top + \beta K_{\text{ad}}(t) \right), \tag{19}$$

where $e(t) = x(t) - x_{\text{ref}}(t)$ is the tracking error, Γ is a diagonal matrix of positive adaptation rates, and β is a forgetting factor that mitigates the influence of older errors on the adaptive gains.

2. Standard POMDP Approach for UAM

The standard Partially Observable Markov Decision Process (POMDP) approach assumes a static operational environment. The objective function is formulated as equation (1) and utilizes a deterministic policy based on a static belief model, which is updated using conventional Bayesian estimation techniques [17].

3. Robust Controller for UAM

The robust controller is engineered to withstand uncertainties and disturbances without the need for real-time parameter adaptation.

The design of the robust controller is premised on worst-case scenario modeling. It aims to maintain the vehicle's stability and trajectory tracking despite the worst possible disturbance effects within a predefined set. The controller

employs a fixed-gain feedback law, which is determined through a rigorous design process that accounts for the uncertainties and disturbances characterized in the simulation environment.

The control law is formalized as:

$$u(t) = K_{\text{robust}} x(t), \tag{20}$$

where K_{robust} is a constant gain matrix computed during the design phase. The matrix K_{robust} is derived through μ-synthesis tailored to ensure performance and stability in the presence of model uncertainties and external disturbances.

C. Simulation Execution and Evaluation

The performance of the optimized policy was evaluated through a series of in-flight maneuvers within a single comprehensive simulation case, including:

- A transition from hover to forward flight.
- A climbing right-hand turn during cruise flight.
- Flight under sinusoidal input conditions.

These segments were components of a unified simulation trajectory designed to test the autonomous vehicle's ability to adapt its control policy to changes in operating conditions. The integrated test case systematically examined the vehicle's performance through hovering, cruise, and aggressive maneuvering within a single simulation run.

We monitored the entire simulation execution using Simulink's built-in scopes, which provided real-time feedback. Resultant trajectories and control inputs were further analyzed using GUAM's plotting utility to visualize the vehicle's performance throughout the full sequence, indicating successful policy adaptation in response to varying flight demands.

The results indicated that the optimized policy π^* provided by our TV-POMDP framework could successfully navigate the vehicle through the simulated time-varying disturbances. The policy demonstrated adaptive control behavior that was robust to environmental changes, validating the effectiveness of the proposed methodology.

	Time-Varying Transition $T_t = 1/(1+e^{-(t-5)}); 0 \le t \le 10$	
	Total distance traveled before crashing (ft)	**MAE**
MPSE	1834	0.10321
MRAC	1318	0.29876
POMDP (Classical estimation)	1210	0.30219
Robust Controller	730	0.40862

Table 1 Performance comparison of the GUAM agent controlled by MPSE, POMDP with classical estimation, MRAC, and a robust controller over a time-varying transition function. The total distance before crashing and the Mean Absolute Error (MAE) are tabulated to evaluate control efficacy.

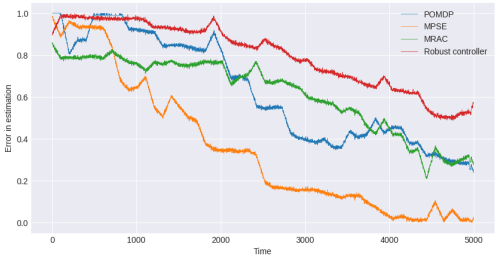

Fig. 2 Error evolution over time for the GUAM agent controlled by MPSE, POMDP with classical estimation, robust controller and MRAC.

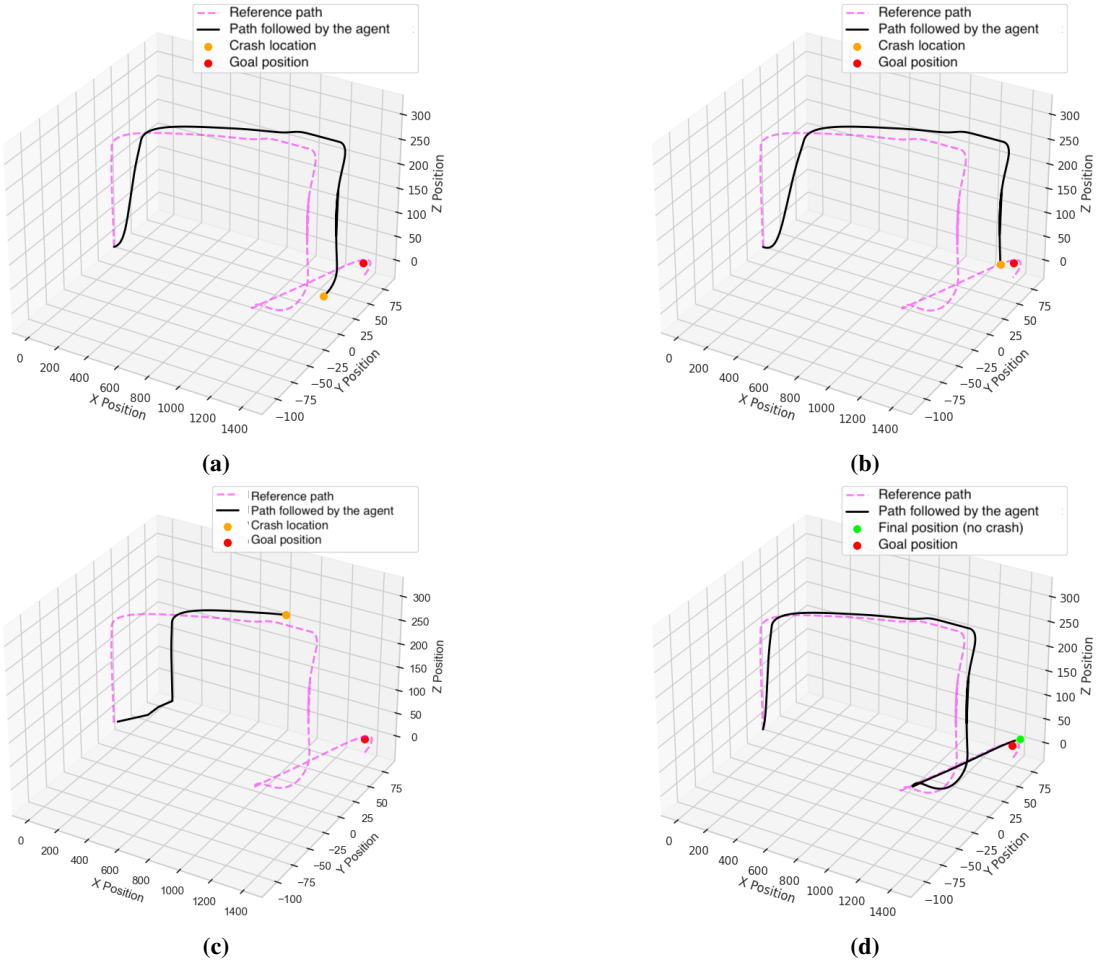

Fig. 3 Visualisation of the trajectories followed by the GUAM agent operating to achieve a complex maneuver(a) illustrates the path followed by the MRAC controller, (b) illustrates the path followed by the system modeled as a POMDP coupled with classical estimation, (c) illustrates the path followed by the robust controller, and (d) illustrates the path followed by modeling the system as a TV-POMDP coupled with a MPSE.

The results, as depicted in Figure 3 and quantified in Table 1, elucidate the trajectories and performance metrics of the GUAM agent executing a complex maneuver sequence from hover to forward flight, followed by cruise and modulation through sinusoidal inputs. The trajectories illustrate the paths rendered by the GUAM agent under the control of the MRAC, POMDP with classical estimation, robust controller and the TV-POMDP coupled with MPSE. Notably, while the MRAC and POMDP controllers can navigate the agent toward the goal, we observe significant deviations from the reference trajectory, leading to premature system failures ($sys = 0$) before reaching the final waypoint. Robust controller on the other hand, fails much before the other baselines – in particular due to loss of control where the angular velocities falls outside the operational range ($[-1, 1]^3$). These deviations are quantitatively corroborated by the higher Mean Absolute Error (MAE) values reported for MRAC POMDP and robust controller as opposed to the lower MAE for the MPSE approach. Furthermore, the total distance traveled before a system crash reinforces the superior robustness of the MPSE method, a considerable increase over the distance traversed under MRAC and POMDP controls. Figure 2 shows the error in the trajectory tracked over time. It is evident that the error associated with the MPSE approach not only diminishes at a notably accelerated rate but also stabilizes at a substantially lower magnitude relative to the comparative methods. These results collectively underscore the augmented accuracy and reliability offered by the TV-POMDP framework with MPSE in maintaining adherence to the desired trajectory and achieving operational longevity, even in the face of intricate dynamic disturbances.

VI. Conclusion

This work has demonstrated the application of Time-Varying Partially Observable Markov Decision Processes (TV-POMDP) and Memory Prioritized State Estimation (MPSE) within the advanced GUAM simulation environment. The results underscore the necessity of online adaptive planning methodologies for aerospace vehicles faced with uncertain, time-varying, and partially observable environments.

Empirical evaluations within the GUAM simulator have substantiated the effectiveness of our refined approach. The results highlight the heightened accuracy in trajectory following when compared to traditional methods and increased resilience to system perturbations. In conclusion, this work not only provides an application of a robust framework for the advancement of autonomous aerospace systems but also lays the groundwork for future implementations that will benefit real-world aeronautical applications. Future work will focus on adapting these simulation-verified methods to enhance computational efficiency thus bringing us closer to realizing fully autonomous aerospace systems capable of reliable operation in complex, unpredictable, and time-varying environments.

Acknowledgments

This work was supported by NASA under grant 80NSSC22M0070 and by the Office of Naval Research under grant N00014-23-1-2505. We would like to thank our collaborators for their contributions and insights.

References

[1] Rodriguez, A., Parr, R., and Koller, D., "Reinforcement learning using approximate belief states," *12th International Conference on Neural Information Processing Systems*, 1999, pp. 1036–1042.

[2] Strehl, A. L., Li, L., and Littman, M. L., "Reinforcement learning in finite MDPs: PAC analysis," *Journal of Machine Learning Research*, Vol. 10, 2009, pp. 2413–2444.

[3] Sutton, R. S., and Barto, A. G., *Reinforcement Learning: An Introduction*, 2nd ed., The MIT Press, 2018.

[4] Watkins, C. J. C. H., and Dayan, P., "Q-learning," *Machine Learning*, Vol. 8, No. 3, 1992, pp. 279–292.

[5] Sutton, R. S., McAllester, D., Singh, S., and Mansour, Y., "Policy gradient methods for reinforcement learning with function approximation," *Advances in Neural Information Processing Systems*, Vol. 12, edited by S. Solla, T. Leen, and K. Müller, MIT Press, 1999.

[6] Ornik, M., and Topcu, U., "Learning and planning for time-varying MDPs using maximum likelihood estimation," *Journal of Machine Learning Research*, Vol. 22, No. 1, 2021.

[7] Puthumanaillam, G., Liu, X., Mehr, N., and Ornik, M., "Weathering ongoing uncertainty: learning and planning in a time-varying partially observable environment," *Preprint*, 2023.

[8] Acheson, M. J., Gregory, I. M., and Cook, J., "Examination of unified control incorporating generalized control allocation," *AIAA Scitech 2021 Forum*, 2021.

[9] Liu, L., and Sukhatme, G. S., "A solution to time-varying markov decision processes," *IEEE Robotics and Automation Letters*, Vol. 3, No. 3, 2018, pp. 1631–1638.

[10] Gregory, I. M., Campbell, N. H., Neogi, N. A., Holbrook, J. B., Grauer, J. A., Bacon, B. J., Murphy, P. C., Moerder, D. D., Simmons, B. M., Acheson, M. J., Britton, T. C., and Cook, J. W., "Intelligent contingency management for urban air mobility," *Dynamic Data Driven Applications Systems*, 2020, pp. 22–26.

[11] Cunis, T., Burlion, L., and Condomines, J.-P., "Piecewise polynomial modeling for control and analysis of aircraft dynamics beyond stall," *Journal of Guidance, Control, and Dynamics*, Vol. 42, No. 4, 2019, pp. 949–957.

[12] Maćkiewicz, A., and Ratajczak, W., "Principal components analysis (PCA)," *Computers Geosciences*, Vol. 19, No. 3, 1993, pp. 303–342.

[13] Sinaga, K. P., and Yang, M.-S., "Unsupervised k-means clustering algorithm," *IEEE Access*, Vol. 8, 2020, pp. 80716–80727.

[14] Andersen, M. S., Dahl, J., and Vandenberghe, L., "CVXOPT: A python package for convex optimization, version 1.1.5," , 2012. Available at `abel.ee.ucla.edu/cvxopt`.

[15] Chu, T., Starek, M. J., Berryhill, J., Quiroga, C., and Pashaei, M., "Simulation and characterization of wind impacts on sUAS flight performance for crash scene reconstruction," *Drones*, Vol. 5, No. 3, 2021.

[16] Dayal, A., Cenkeramaddi, L. R., and Jha, A., "Reward criteria impact on the performance of reinforcement learning agent for autonomous navigation," *Applied Soft Computing*, Vol. 126, 2022, p. 109241.

[17] Ross, S., Pineau, J., Chaib-draa, B., and Kreitmann, P., "A Bayesian approach for learning and planning in partially observable markov decision processes." *Journal of Machine Learning Research*, Vol. 12, No. 5, 2011.