Finding a Needle in a Haystack: Scenario Coverage Verification for Single-UAV Missions Using SMT

Qurrat Ul Ain[1], Abhoy Kole[2], Muhammad Hassan[2,3], Rolf Drechsler[2,3]
[1]Department of Software Engineering, NUCES, Islamabad, Pakistan
[2]Cyber-Physical Systems, DFKI, Bremen, Germany
[3]Institute of Computer Science, University of Bremen, Bremen, Germany
quratul.ain.v@isb.nu.edu.pk abhoy.kole@dfki.de {hassan, drechsler}@uni-bremen.de

***Abstract*—In this paper, we propose an SMT-based framework for scenario coverage verification of single-UAV missions under bounded environmental assumptions. We present an abstract mission-level model of a single UAV in a bounded 3D environment. The model captures mission states, stepwise 3D motion, battery depletion, obstacles, NFZ, and altitude constraints. Mission failure is represented by an emergency state corresponding to loss of safe recoverability. The SMT query searches for an admissible environment configuration and a corresponding execution that violates the mission requirements. A satisfiable result yields a concrete counterexample scenario, while an unsatisfiable result establishes bounded correctness over the modeled scenario family. The key contribution is lifting bounded verification from individual system instances to families of systems defined over a symbolic environment parameter space, enabling quantified reasoning over all admissible environment configurations within a single SMT query. This enables exhaustive scenario coverage within bounded assumptions, which is not provided by finite sampling-based validation within the same bounded setting.**

***Keywords*—Uncrewed aerial vehicles, formal verification, satisfiability modulo theories, scenario coverage, bounded model checking, mission safety, no-fly zones.**

I. Introduction

Uncrewed Aerial Vehicles (UAVs) are increasingly deployed in safety-critical missions that require autonomous navigation under operational constraints such as altitude limits, no-fly zones (NFZs), and obstacle avoidance [1]. In practice, mission validation is still largely carried out through simulation and scenario-based testing, where a finite set of environment instances is examined [2]. While such testing is useful, it does not guarantee that a given mission policy remains safe across *all* relevant situations within a bounded operating envelope [3]. This limitation is particularly significant for UAV systems, where mission safety often depends not only on the vehicle policy, but also on variations in environmental factors such as obstacle placement, NFZ geometry, and altitude constraints.

Formal verification offers a complementary perspective by reasoning over symbolic models rather than a small set of sampled executions. Over the past decades, such techniques have matured and are widely applied in domains such as hardware verification [4], [5], security protocols [6], and cyber-physical systems [7]. Recent work has also applied these techniques, including Satisfiability Modulo Theories-based (SMT) methods [8], to UAV planning, mission reasoning, and controller assurance. However, existing approaches rem'ain fragmented across abstraction layers and do not directly address the following question: given a fixed mission policy for a *single* UAV, do the mission requirements hold for *all* environment configurations within explicitly bounded assumptions? This differs fundamentally from path synthesis, optimization, or controller design.

In this paper, we propose an SMT-based framework for scenario coverage verification for single UAV missions. More concretely, the framework enables modeling of UAV, environment (including obstacles and NFZs), and total number of allowed obstacles and NFZs as abstract models. Mission execution is represented as a finite state-transition system consisting of an initial state, intermediate navigation states, and a landing state. Additionally, a failure state is defined which represents a mission failure. The system evolves over discrete steps subject to state-transition rules and bounded environment assumptions. The framework encodes the environment, mission, UAV, and the UAV transition model as an SMT problem. The verification query asks whether there exists an environment configuration and execution that violates the mission requirements. A satisfiable (SAT) result yields a concrete counterexample consisting of both an environment instance and a UAV trajectory. An unsatisfiable (UNSAT) result establishes that no violating scenario exists within the specified bounds, thereby providing bounded correctness guarantees and mission success. Consequently, the proposed framework provides scenario coverage verification over a symbolic space of admissible environment configurations. Specifically, we provide:

- an abstract mission-level state-transition model for a single UAV in a bounded 3D environment,
- an SMT encoding of mission requirements together with bounded environmental parameters, including obstacles, NFZs, and altitude limits,
- a verification interpretation in which SAT results correspond to concrete violating scenarios and UNSAT results establish bounded correctness, and
- an empirical evaluation demonstrating symbolic scenario coverage and counterexample extraction under bounded

This work was supported in part by the German Ministry for Research, Technology and Space (BMFTR) with project *ExaVerse* (grant number 01IW25003).

environment profiles.

The remainder of the paper presents the formal model, SMT framework, evaluation, and conclusion.

II. Preliminaries

Formal verification has increasingly been applied to UAV mission planning to provide stronger guarantees than simulation-based validation. For instance, the framework in [8] applies SMT-based reasoning to verify safety constraints in UAV flight planning, but focuses primarily on planner-level correctness rather than robustness across varying environment configurations.

Verification under explicit environment assumptions has been explored to highlight the dependence of correctness on operating conditions. In [1], bounded verification is performed under predefined assumptions, while [9] introduces assume-guarantee contracts to model obstacle-related constraints. However, these approaches do not support symbolic reasoning over a parametric family of environments.

Mission-level reasoning has been widely studied using temporal logic-based specifications. Reactive mission planning frameworks [10] and UAV-specific specification models [11], [12] enable formal representation of sequencing and safety constraints, while extensions such as [13] incorporate uncertainty through probabilistic guarantees. These methods focus on specification and synthesis rather than exhaustive verification under bounded environmental assumptions.

Beyond planning, formal verification of UAV systems has been explored at multiple levels. Airspace safety and detect-and-avoid logic are analyzed in [14], while control-level safety guarantees using SMT-based techniques are presented in [15]. At the software level, formal verification has been applied to autopilot systems and flight-plan generation [16], [17]. Although these approaches demonstrate the effectiveness of formal methods, they address different abstraction layers and do not provide mission-level scenario coverage under parametric environment models.

Despite these advances, there is no unified framework that verifies a fixed UAV mission policy across a bounded family of environment configurations using SMT. Existing approaches do not simultaneously provide (i) explicit symbolic modeling of environment parameter spaces, (ii) interpretation of SAT results as concrete counterexamples, and (iii) systematic reasoning over both safe and unsafe regions of the environment.

This gap motivates the proposed framework. Table I summarizes the key differences and positions the present work with respect to existing approaches. Unlike traditional bounded model checking, which verifies correctness for a fixed system instance, the proposed framework lifts verification to the level of environment families, enabling reasoning over a structured parameter space rather than a single model instance.

Existing approaches either fix the environment and verify behavior, or sample environment configurations. They do not treat the environment itself as a symbolic search space subject to formal reasoning. A key limitation across prior work is that the environment is either fixed or explored through sampling.

TABLE I
GAP-ORIENTED SUMMARY OF RELATED WORK. *(✗= No, ✓= Yes, ∼ = Partial.)*

Ref.	Focus	Bounded env	Scenario	Scalability	Main limitation w.r.t. this work
[8]	UAV flight planning with NFZ safety	∼	✗	✗	Planner-level only; no policy verification; no bounded env modeling
[1], [9]	Verification under environment assumptions	∼	Limited	Limited	Explicit assumptions; no symbolic env modeling; no universal SMT coverage
[10]–[13]	Mission-level temporal reasoning and specification	✗	✗	✗	Specification + synthesis only; no bounded-env verification
[14]–[17]	Adjacent single-UAV formal verification	✗	✗	✗	Layer-specific focus; no mission-level reasoning; no scenario coverage
This work	Single-UAV mission policy verification using SMT	✓	✓	✓	Unified framework; bounded env modeling; scenario-level counterexamples; scalable analysis

As a result, verification guarantees are restricted to individual instances or probabilistic coverage. In contrast, this work treats the environment itself as a symbolic parameter space and performs quantified reasoning over all admissible configurations within bounded assumptions. This enables bounded symbolic scenario coverage within the modeled environment family., which is not supported by existing approaches.

III. Formal Model and Problem Statement

We consider a single UAV operating in a bounded 3D grid environment. The objective is not to synthesize an optimal path, but to verify whether a fixed mission policy satisfies the mission requirements for all admissible environment configurations within a bounded horizon.

The model is based on the following deliberate design choices: 1) the UAV is modeled at the mission level rather than at the vehicle-dynamics level, making the analysis independent of platform-specific hardware, 2) the workspace is discretized as a bounded 3D grid, consistent with standard bounded model checking practice, 3) obstacles and NFZs are static during the bounded verification horizon, isolating the effect of environmental placement from temporal dynamics, and 4) continuous-time dynamics and speed are not modeled, as one transition represents one abstract mission-level step. These choices are intentional, the goal is to verify mission-policy correctness under environmental uncertainty before committing to a specific platform or controller design. The following subsections define the formal model and verification semantics in detail.

A. Scenario Definition

A verification scenario is defined as

$$\mathcal{S} = \langle M, R, \mathcal{E}, \Pi, X_0, \delta_\Pi, \Phi, K \rangle, \tag{1}$$

where M is the mission model, R is the UAV resource model, $\mathcal{E}$ is the bounded environment family, Π is the fixed mission policy, X_0 is the initial state, δ_Π is the transition relation w.r.t. fixed policy (see Section III-H), Φ is the mission requirement, and $K \in \mathbb{N}$ is the bounded verification horizon. The solver checks bounded execution traces of the form $X_0, X_1, \ldots, X_K$, where each X_k is a system state at step k.

Unlike standard bounded model checking, where the system model is fixed, the proposed formulation treats the environment as symbolic. Each admissible environment valuation $\theta \in \Theta$ instantiates a concrete transition system, so the verification problem ranges over both executions and environment configurations. This lifts verification from a single fixed environment configuration to a parameterized scenario space.

B. Workspace and Environment Model

The UAV operates in a bounded grid workspace

$$\Omega = \{0, \ldots, L_x\} \times \{0, \ldots, L_y\} \times \{0, \ldots, L_z\} \subseteq \mathbb{Z}^3. \quad (2)$$

The environment is parameterized by a symbolic valuation $\theta \in \Theta$, where Θ is the set of admissible environment valuations. Its formal definition is given later in Eq. (9), after endpoint feasibility has been introduced. For a given θ, the concrete environment is

$$E(\theta) = \langle \Omega, \mathcal{O}(\theta), \mathcal{N}(\theta), A^{\text{env}}_{\max}(\theta) \rangle . \quad (3)$$

Here, $\mathcal{O}(\theta)$ is the set of obstacles, $\mathcal{N}(\theta)$ is the set of NFZ, and $A^{\text{env}}_{\max}(\theta)$ is the environment altitude bound. We instantiate each verification problem with fixed symbolic obstacle and NFZ slots $\mathcal{O}(\theta) = \{O_1, \ldots, O_{n_O}\}$ and $\mathcal{N}(\theta) = \{N_1, \ldots, N_{n_N}\}$. The parameters of these slots are selected symbolically by θ. Varying n_O and n_N allows us to study how solver performance changes as the environment family becomes larger. The bounded environment family is $\mathcal{E} = \{E(\theta) \mid \theta \in \Theta\}$. Each obstacle O_i is modeled as a rectangular prism:

$$O_i = \big((x_i^{\min}, y_i^{\min}, 0), (x_i^{\max}, y_i^{\max}, h_i)\big).$$

The obstacle parameter bounds are captured by the predicate

$$\begin{aligned} \text{ObsBounds}(\theta) \iff \bigwedge_{i=1}^{n_O} \big(&0 \le x_i^{\min} \le x_i^{\max} \le L_x \\ &\wedge 0 \le y_i^{\min} \le y_i^{\max} \le L_y \\ &\wedge 0 \le h_i \le L_z\big). \end{aligned} \quad (4)$$

The obstacle occupancy predicate is

$$\begin{aligned} \text{Occ}_{O_i}(x, y, z) \iff &\big(x_i^{\min} \le x \le x_i^{\max}\big) \\ &\wedge \big(y_i^{\min} \le y \le y_i^{\max}\big) \\ &\wedge (0 \le z \le h_i)\,. \end{aligned} \quad (5)$$

Each NFZ N_j is modeled as a horizontal forbidden rectangle $N_j = \big((x_j^{\min}, y_j^{\min}), (x_j^{\max}, y_j^{\max})\big)$.

The NFZ parameter bounds are captured by the predicate

$$\begin{aligned} \text{NFZBounds}(\theta) \iff \bigwedge_{j=1}^{n_N} \big(&0 \le x_j^{\min} \le x_j^{\max} \le L_x \\ &\wedge 0 \le y_j^{\min} \le y_j^{\max} \le L_y\big). \end{aligned} \quad (6)$$

Since NFZ membership is independent of altitude in this abstraction, we define

$$\begin{aligned} \text{NFZ}_{N_j}(x, y) \iff &\big(x_j^{\min} \le x \le x_j^{\max}\big) \\ &\wedge \big(y_j^{\min} \le y \le y_j^{\max}\big)\,. \end{aligned} \quad (7)$$

We define the environment bound predicate as the conjunction of the obstacle bounds (Eq. (4)), the NFZ bounds (Eq. (6)), and environment altitude bound:

$$\begin{aligned} \text{EnvBounds}(\theta) \iff &\text{ObsBounds}(\theta) \wedge \text{NFZBounds}(\theta) \\ &\wedge 0 \le A^{\text{env}}_{\max}(\theta) \le L_z. \end{aligned} \quad (8)$$

For a position $p = (x, y, z)$, we write $\text{Occ}_{O_i}(p)$ as shorthand for $\text{Occ}_{O_i}(x, y, z)$, and similarly write $\text{NFZ}_{N_j}(p)$ as shorthand for $\text{NFZ}_{N_j}(x, y, z)$.

C. Mission and Resource Model

The mission model is $M = \langle p_{\text{src}}, p_{\text{dst}}, A^{\text{mis}}_{\max} \rangle$, where $p_{\text{src}} \in \Omega$ is the source position, $p_{\text{dst}} \in \Omega$ is the destination landing position, and $A^{\text{mis}}_{\max}$ is the mission altitude limit. We assume that both source and destination lie on the ground plane, i.e., $p_{\text{src}} = (x_{\text{src}}, y_{\text{src}}, 0)$ and $p_{\text{dst}} = (x_{\text{dst}}, y_{\text{dst}}, 0)$.

The UAV resource model is $R = \langle B_0, A^{\text{HW}}_{\max} \rangle$, where $B_0 \in \mathbb{Z}_{\ge 0}$ is the initial battery level and $A^{\text{HW}}_{\max}$ is the hardware altitude limit. The active altitude bound is

$$A_{\max}(\theta) = \min\{A^{\text{HW}}_{\max}, A^{\text{mis}}_{\max}, A^{\text{env}}_{\max}(\theta)\}.$$

Using the workspace Ω from Eq. (2) and the obstacle and NFZ predicates from Eqs. (5) and (7), a position $p \in \Omega$ is clear of environmental constraints when

$$\text{ClearEnv}(p, \theta) \iff \bigwedge_{i=1}^{n_O} \neg \text{Occ}_{O_i}(p) \wedge \bigwedge_{j=1}^{n_N} \neg \text{NFZ}_{N_j}(p).$$

Then the endpoint-validity predicate is

$$\begin{aligned} \text{ValidEndpoints}(\theta) \iff &\text{ClearEnv}(p_{\text{src}}, \theta) \\ &\wedge \text{ClearEnv}(p_{\text{dst}}, \theta). \end{aligned}$$

The admissible environment-valuation space is defined as

$$\Theta = \{\theta \mid \text{EnvBounds}(\theta) \wedge \text{ValidEndpoints}(\theta)\}. \quad (9)$$

D. Mission States and System State

Mission execution is modeled using the finite set of mission modes

$$S = \{S_1, S_2, S_3, S_4, S_5, S_6^{\text{obs}}, S_6^{\text{nfz}}, S_7\},$$

where S_1 denotes the initial ground state, S_2 takeoff, S_3 nominal flight, S_4 destination reached, S_5 landed and terminated, S_6^{obs} obstacle avoidance, S_6^{nfz} NFZ avoidance, and S_7 emergency or failure. The full state space is

$$\mathcal{X} = S \times \Omega \times \mathbb{Z}. \quad (10)$$

Thus, each system state consists of a mission mode, a grid position, and a battery value. At each step $k \in \{0, \ldots, K\}$, the system state is $X_k = \langle q_k, p_k, b_k \rangle \in \mathcal{X}$, where $q_k \in S$ is the mission mode, $p_k \in \Omega$ is the UAV position, and b_k

is the battery value. The initial state is $X_0 = \langle S_1, p_{\text{src}}, B_0 \rangle$. The landed state S_5 and the failure state S_7 are terminal modes, their absorbing behavior is defined in the full transition relation.

E. Fixed Mission Policy

The mission policy is fixed before verification and is not synthesized by the solver. We represent it as $\Pi = \langle \pi_{\text{nom}}, \pi_{\text{obs}}, \pi_{\text{nfz}} \rangle$, where π_{nom} gives the nominal motion command, π_{obs} gives the obstacle-avoidance command, and π_{nfz} gives the NFZ-avoidance command.

The set of admissible nonzero motion commands is

$$\mathcal{D} = \{-1, 0, 1\}^3 \setminus \{(0, 0, 0)\}. \tag{11}$$

Each command $\Delta = (\Delta x, \Delta y, \Delta z) \in \mathcal{D}$ moves the UAV by at most one grid cell along each coordinate axis. The command $(0, 0, 0)$ is excluded for active flight states, since it represents no physical movement.

For avoidance behavior, let $\mathcal{D}_{\text{lat}}$ denote the set of lateral motion commands. We define

$$\mathcal{D}_{\text{lat}} = \{(1, 0, 0), (-1, 0, 0), (0, 1, 0), (0, -1, 0)\},$$
$$\mathcal{D}_{\text{obs}} = \mathcal{D}_{\text{lat}} \cup \{(0, 0, 1)\}, \qquad \mathcal{D}_{\text{nfz}} = \mathcal{D}_{\text{lat}}.$$

Thus, $\mathcal{D}_{\text{obs}}$ allows obstacle avoidance by lateral motion or by climbing one grid cell, whereas $\mathcal{D}_{\text{nfz}}$ restricts NFZ avoidance to lateral motion because NFZ membership is altitude-independent in this abstraction.

The policy functions are deterministic:

$$\pi_{\text{nom}} : \mathcal{X} \times \Theta \to \mathcal{D}, \tag{12}$$
$$\pi_{\text{obs}} : \mathcal{X} \times \Theta \to \mathcal{D}_{\text{obs}}, \tag{13}$$
$$\pi_{\text{nfz}} : \mathcal{X} \times \Theta \to \mathcal{D}_{\text{nfz}}. \tag{14}$$

Here, $\mathcal{X}$ denotes the system state space defined in Eq. (10). The nominal policy is assumed to be destination-directed, i.e., during nominal flight, it selects commands that do not move the UAV farther from the destination, while avoidance commands may temporarily deviate from this behavior to avoid obstacles or NFZ. Concretely, π_{nom} selects the axis-aligned step minimising Manhattan distance to p_{dst}, π_{obs} climbs one grid cell, and π_{nfz} steps laterally away from the NFZ boundary.

F. Motion and Battery Dynamics

For active physical movement, the UAV applies a nonzero motion command $\Delta_k = (\Delta x_k, \Delta y_k, \Delta z_k) \in \mathcal{D}$, where $\mathcal{D}$ is the admissible motion-command set defined in Eq. (11). The position and battery updates are

$$p_{k+1} = p_k + \Delta_k, \tag{15}$$
$$b_{k+1} = b_k - \text{Cost}(\Delta_k). \tag{16}$$

Logical state changes, such as entering an avoidance state or marking the mission as landed, do not involve physical movement. For such transitions, we use the zero command $\Delta^0 = (0, 0, 0)$. The uniform motion cost model is

$$\text{Cost}(\Delta) = \begin{cases} c_{\text{move}}, & \Delta \in \mathcal{D}, \\ 0, & \Delta = \Delta^0, \end{cases} \tag{17}$$

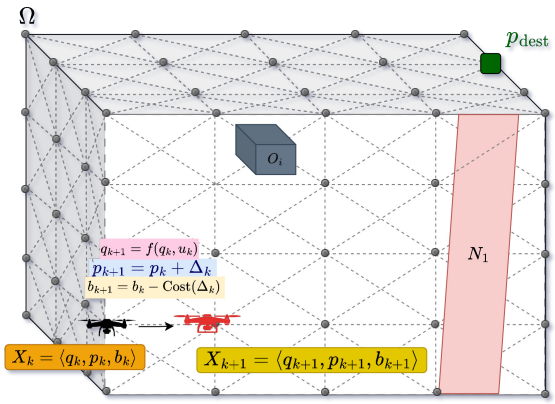

Fig. 1. One-step UAV transition in the discretized three-dimensional workspace Ω. The system state $X_k = \langle q_k, p_k, b_k \rangle$ evolves to $X_{k+1} = \langle q_{k+1}, p_{k+1}, b_{k+1} \rangle$ using the grid update $p_{k+1} = p_k + \Delta_k$. Obstacles O_i and NFZ N_j restrict which transitions are feasible under the environment valuation θ.

where $c_{\text{move}} > 0$ is the battery cost of one mission-level movement step.

Thus, the cost does not depend on whether the UAV moves horizontally, vertically, or diagonally. All nonzero grid movements consume the same amount of battery, whereas zero-motion logical transitions consume no battery.

For compactness, we define the execution predicate

$$\text{Exec}(X_k, \Delta_k, X_{k+1}) \iff (p_{k+1} = p_k + \Delta_k) \wedge (b_{k+1} = b_k - \text{Cost}(\Delta_k)). \tag{18}$$

Fig. 1 illustrates the geometric interpretation of one mission-level transition. The UAV operates on a bounded three-dimensional grid workspace Ω, and each state $X_k = \langle q_k, p_k, b_k \rangle$ contains the current mission mode, grid position, and battery level. At each step, the policy selects a motion command Δ_k, and the position is updated according to Eq. (15). The transition is feasible only if the resulting grid position satisfies the workspace, altitude, obstacle, NFZ, and battery constraints. Thus, Fig. 1 provides an intuitive view of how the motion model, environment valuation θ, and transition relation δ_Π interact during one bounded execution step.

G. Safety Predicates

For $p = (x, y, z)$ and $\theta \in \Theta$, the pointwise physical safety predicate is

$$\text{PosSafe}(p, \theta) \iff p \in \Omega \wedge 0 \le z \le A_{\max}(\theta) \wedge \text{ClearEnv}(p, \theta).$$

The predicate $\text{PosSafe}(p, \theta)$ specifies whether a grid position is physically safe under environment valuation θ. A position is safe if it lies inside the bounded workspace, satisfies the active altitude bound, does not intersect any obstacle, and does not lie inside any NFZ. The full state-safety predicate is

$$\text{Safe}(X_k, \theta) \iff \text{PosSafe}(p_k, \theta) \wedge b_k \ge 0.$$

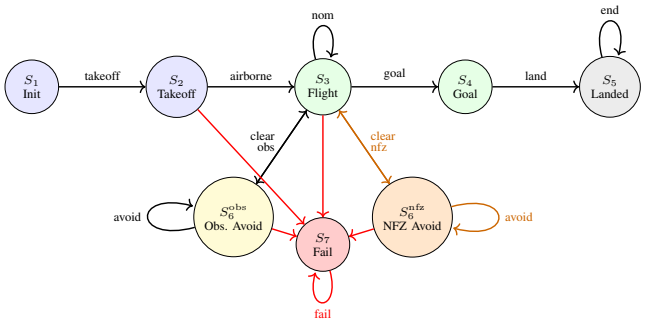

Fig. 2. Mission-level state diagram of the UAV showing nominal execution, obstacle avoidance, NFZ avoidance, landing, and emergency transitions.

Since the model is a mission-level discrete abstraction, safety is checked only at the grid positions visited by the UAV. The continuous swept volume between two consecutive grid positions is not modeled. We use the following auxiliary predicates for step safety and environment blocking:

$$\begin{aligned}\text{StepSafe}(p,\Delta,\theta) &\iff \text{PosSafe}(p+\Delta,\theta),\\ \text{ObsBlock}(p,\theta) &\iff \bigvee_{i=1}^{n_O} \text{Occ}_{O_i}(p),\\ \text{NFZBlock}(p,\theta) &\iff \bigvee_{j=1}^{n_N} \text{NFZ}_{N_j}(p).\end{aligned}$$

A commanded move is feasible if the resulting grid position is safe and, using the battery update in Eq. (16), the successor battery value satisfies $b_{k+1} \geq 0$.

$$\text{Feas}(X_k,\Delta,\theta) \iff \text{StepSafe}(p_k,\Delta,\theta) \wedge b_k - \text{Cost}(\Delta) \geq 0. \tag{19}$$

H. Full Transition Relation

The full transition relation δ_Π defines how the UAV state evolves under the fixed mission policy. It combines mission-mode transitions, policy-selected motion commands, position updates, battery updates, and failure handling. Fig. 2 gives the corresponding mission-mode view, the UAV progresses from initialization and takeoff to nominal flight, may enter obstacle or NFZ avoidance when the corresponding guard is triggered, reaches landing through the goal state, or transitions to failure when an emergency guard holds. Let $\Delta^0 = (0,0,0)$ denote the zero-motion command and $\Delta^{\text{to}} = (0,0,1)$ denote the takeoff command. For $\alpha \in \{\text{nom}, \text{obs}, \text{nfz}\}$, the policy-selected motion command and its candidate successor position are

$$\begin{aligned}\Delta_k^\alpha &= \pi_\alpha(X_k,\theta),\\ \hat{p}_{k+1}^\alpha &= p_k + \Delta_k^\alpha.\end{aligned}$$

Here, π_α denotes the corresponding nominal, obstacle-avoidance, or NFZ-avoidance policy component from Eqs. (12)–(14), and the candidate successor position follows the update rule in Eq. (15). The main guards used in the transition relation are

$$\begin{aligned}\gamma_{\text{obs}}(X_k,\theta) &\iff \text{ObsBlock}(\hat{p}_{k+1}^{\text{nom}},\theta),\\ \gamma_{\text{nfz}}(X_k,\theta) &\iff \neg\text{ObsBlock}(\hat{p}_{k+1}^{\text{nom}},\theta)\\ &\quad \wedge \text{NFZBlock}(\hat{p}_{k+1}^{\text{nom}},\theta),\\ \text{NomClear}(X_k,\theta) &\iff \text{Feas}(X_k,\pi_{\text{nom}}(X_k,\theta),\theta),\\ \gamma_{\text{emg}}(X_k,\Delta,\theta) &\iff \neg\text{Feas}(X_k,\Delta,\theta).\end{aligned}$$

The definition of γ_{nfz} gives obstacle avoidance priority when the nominal successor intersects both an obstacle and an NFZ. At the mission level, the allowed mode changes are represented by the guarded mode-transition relation

$$\delta_S \subseteq S \times \Gamma \times S, \tag{20}$$

where Γ is the set of transition guards. The guarded transitions are summarized in Table II where each row represents a transition (S_a,γ,S_b) that is enabled when the guard condition holds. They are interpreted at the mission level. We distinguish two kinds of one-step transitions. A transition is either mode-only or movement-based. Mode-only transitions change only the mission mode and satisfy $p_{k+1} = p_k$ and $b_{k+1} = b_k$, examples include entering avoidance mode, marking the destination as reached, and completing landing. Movement transitions apply a nonzero motion command selected by the fixed policy and satisfy Eq. (18). Let $\mathcal{T}$ denote the guarded transitions in Table II. The full transition relation is

$$\delta_\Pi(X_k,X_{k+1},\theta) \iff \bigvee_{r\in\mathcal{T}} \text{Enabled}_r(X_k,X_{k+1},\theta). \tag{21}$$

For each row r, Enabled_r encodes the corresponding source mode, target mode, guard, and update rule. Infeasible movement commands, according to Eq. (19), transition to S_7. The terminal modes S_5 and S_7 are absorbing, so $X_{k+1} = X_k$.

TABLE II
GUARDED MISSION-STATE TRANSITIONS.

From	To	Guard meaning
S_1	S_2	mission begins and the UAV enters takeoff mode
S_2	S_3	takeoff command is feasible and the UAV becomes airborne
S_3	S_3	nominal flight continues safely
S_3	S_4	destination position is reached
S_4	S_5	landing is completed at the destination
S_3	S_6^{obs}	nominal successor intersects an obstacle
S_6^{obs}	S_6^{obs}	obstacle avoidance is still required
S_6^{obs}	S_3	obstacle is cleared and nominal flight may resume
S_3	S_6^{nfz}	nominal successor intersects a NFZ
S_6^{nfz}	S_6^{nfz}	NFZ avoidance is still required
S_6^{nfz}	S_3	NFZ is avoided and nominal flight may resume
S_i	S_7	emergency or infeasible command, where $S_i \in \{S_2, S_3, S_6^{\text{obs}}, S_6^{\text{nfz}}\}$

I. Mission Requirements

The safety requirement states that every state in the bounded execution must satisfy the physical safety constraints:

$$\Phi_{\text{safe}}(\mathbf{X}_{0:K}, \theta) \iff \bigwedge_{k=0}^{K} \text{Safe}(X_k, \theta). \tag{22}$$

The mission-completion requirement states that the UAV must reach the landed state at the destination within the bounded horizon, and must never enter the failure state:

$$\Phi_{\text{goal}}(\mathbf{X}_{0:K}) = \left(\bigvee_{k=0}^{K} (q_k = S_5 \wedge p_k = p_{\text{dst}}) \right) \wedge \left(\bigwedge_{k=0}^{K} q_k \neq S_7 \right) \tag{23}$$

The full mission requirement is

$$\Phi = \Phi_{\text{safe}} \wedge \Phi_{\text{goal}}. \tag{24}$$

Thus, a successful execution is one that reaches the landed state at the destination within the bounded horizon, never enters the failure state, and satisfies all safety constraints throughout the run.

J. Bounded Verification Problem

The bounded scenario coverage verification problem asks whether the fixed mission policy satisfies the mission requirement for every admissible environment configuration $\theta \in \Theta$ within the chosen horizon K. Let $\mathbf{X}_{0:K} = (X_0, \ldots, X_K)$ denote a bounded execution trace. For a given θ, the trace is valid if it starts from the initial state and follows the transition relation at every step:

$$\text{Run}_K(\mathbf{X}_{0:K}, \theta) \iff X_0 = \langle S_1, p_{\text{src}}, B_0 \rangle \wedge \bigwedge_{k=0}^{K-1} \delta_\Pi(X_k, X_{k+1}, \theta).$$

The verification property is that every valid bounded execution satisfies the mission requirement:

$$\forall \theta \in \Theta,\ \forall \mathbf{X}_{0:K} \in \mathcal{X}^{K+1} : \quad \text{Run}_K(\mathbf{X}_{0:K}, \theta) \Rightarrow \Phi(\mathbf{X}_{0:K}, \theta).$$

Equivalently, the SMT solver checks the counterexample query

$$\exists \theta \in \Theta,\ \exists \mathbf{X}_{0:K} : \quad \text{Run}_K(\mathbf{X}_{0:K}, \theta) \wedge \neg \Phi(\mathbf{X}_{0:K}, \theta). \tag{25}$$

If Eq. (25) is SAT, the satisfying assignment gives an admissible environment configuration and bounded execution that violate the mission requirement, i.e., a concrete counterexample scenario. If it is UNSAT, then no violating scenario exists for the fixed policy within the specified environment bounds and horizon.

IV. Proposed SMT-based Framework

The overall verification workflow is illustrated in Fig. 3. It summarizes the end-to-end pipeline, mission specifications and bounded environment assumptions are abstracted into a mission-level UAV model and symbolic environment parameterization, which are then encoded as a bounded SMT verification problem. The solver outcome is interpreted as either a SAT counterexample scenario, consisting of a concrete environment configuration and UAV trajectory, or an UNSAT bounded-correctness result. The framework also supports scalability analysis by varying obstacle density, NFZ size, and obstacle height. The proposed SMT-based framework performs scenario coverage verification for a fixed single-UAV mission policy under bounded environmental assumptions. Rather than evaluating fixed environment configurations one at a time, the mission model and environment parameters are encoded symbolically, enabling reasoning over all admissible configurations in the parameter space Θ.

The framework integrates three key components: (i) the mission-level UAV model, which captures mission modes, motion, resource usage, and safety constraints; (ii) the symbolic environment parameterization, which defines bounded obstacle, NFZ, and altitude configurations; and (iii) the SMT-based counterexample query, which combines the mission model and environment parameters into a unified verification problem. The transition relation δ_Π in Eq. (21) and the mission requirement Φ in Eq. (24) are encoded symbolically together with the bounded environment assumptions.

The transition system is unrolled over a finite horizon K, producing a symbolic execution trace $X_0, X_1, \ldots, X_K$. The initial state X_0 is fixed by the mission definition, while each successor state is constrained by the transition relation. Environment parameters, including obstacle corners, NFZ bounds, and altitude limits, are declared as symbolic integer variables constrained by $\text{EnvBounds}(\theta)$. The resulting SMT query is the counterexample formulation in Eq. (25), discharged using Z3. A SAT result yields a concrete violating scenario consisting of an admissible environment configuration and a corresponding UAV trajectory. An UNSAT result establishes that no violating scenario exists for the fixed mission policy within the specified environment bounds and horizon. Thus, the framework provides exhaustive scenario coverage over the modeled bounded environment family rather than validation over sampled configurations.

V. Experimental Evaluation

This section evaluates the proposed SMT-based scenario coverage verification framework under varying environmental bounds and grid sizes. We describe the experimental setup, present scalability results across five workspace configurations, and analyse the verification outcomes in detail, including two concrete case interpretations.

A. Experimental Setup

The evaluation is conducted over a bounded environment parameter space Θ (Eq. 9), which defines the admissible

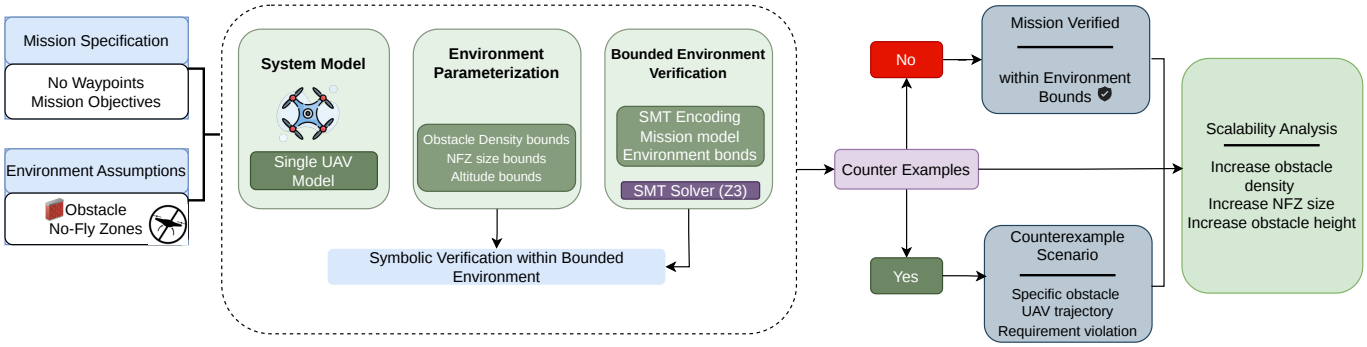

Fig. 3. SMT-based scenario coverage verification framework. SAT yields a violating scenario, while UNSAT establishes bounded correctness.

SAFE	**RISKY**
Obstacle O_i	**Obstacle** O_i
$x_i^{\min}$ in [40, 70]	$x_i^{\min}$ in [10, 45]
$y_i^{\min}$ in [25, 55]	$y_i^{\min}$ in [0, 10]
$x_i^{\max} = x_i^{\min} + \Delta x_i$	$x_i^{\max} = x_i^{\min} + \Delta x_i$
Δx_i in [2, 4]	Δx_i in [8, 12]
$y_i^{\max} = y_i^{\min} + \Delta y_i$	$y_i^{\max} = y_i^{\min} + \Delta y_i$
Δy_i in [2, 4]	Δy_i in [8, 12]
h_i in [2, 4]	h_i in [10, 14]
No-fly zone N_j	**No-fly zone** N_j
$x_j^{\min}$ in [45, 75]	$x_j^{\min}$ in [12, 45]
$y_j^{\min}$ in [25, 55]	$y_j^{\min}$ in [0, 8]
$x_j^{\max} = x_j^{\min} + \Delta x_j$	$x_j^{\max} = x_j^{\min} + \Delta x_j$
Δx_j in [2, 4]	Δx_j in [8, 12]
$y_j^{\max} = y_j^{\min} + \Delta y_j$	$y_j^{\max} = y_j^{\min} + \Delta y_j$
Δy_j in [2, 4]	Δy_j in [6, 8]

Fig. 4. Environment parameter ranges for obstacle slots O_i and no-fly-zone slots N_j in SAFE and RISKY regions. Obstacle slots are denoted by O_i, and no-fly-zone slots are denoted by N_j. The variables x and y define horizontal placement, while h_i denotes obstacle height along the z-axis. The quantities Δx and Δy represent the horizontal extents (widths) of obstacles and no-fly zones, used to derive $x^{\max}$ and $y^{\max}$ from the corresponding lower bounds. NFZ constraints are independent of altitude.

placement and size ranges of obstacle slots O_i and NFZ slots N_j. The number of obstacle and NFZ slots is varied over $n_O, n_N \in 1, 2, 3, 4$, resulting in 16 configurations per grid and region. Each configuration corresponds to a single SMT query in which all obstacle and NFZ parameters are treated as bounded integer variables.

The parameter space is further divided into SAFE and RISKY regions, as illustrated in Fig. 4. These regions are defined relative to the nominal mission corridor followed by the fixed policy in the absence of environmental conflicts. SAFE bounds place obstacles and NFZs away from this corridor, whereas RISKY bounds place them near or along the corridor, making mission violations more likely when the fixed policy cannot safely recover.

We evaluate five workspace grids: $25\times25\times25$, $50\times50\times50$, $75\times75\times75$, $100\times100\times100$, and $125\times125\times125$. All queries are solved using Z3 (v4.12) via Python, with a per-query timeout of 10 minutes and a total evaluation budget of 4 hours per grid. For each SAFE or RISKY region, the SMT query ranges over all admissible valuations $\theta \in \Theta$ satisfying the corresponding bounds. Thus, each query reasons symbolically over the bounded environment family rather than evaluating a single sampled instance.

B. Experimental Results

Table III summarizes the solver outcomes across grid sizes, environment regions, and obstacle/NFZ slot counts. Within the four-hour budget, the number of completed queries decreases as the grid size increases, reflecting the growing complexity of the SMT instances. SAFE-region queries predominantly return UNSAT, indicating that no violating scenario exists within the specified bounded symbolic region, whereas RISKY-region queries return SAT when the solver finds an admissible violating environment valuation. The longest runtimes occur for dense configurations with $(n_O, n_N) = (4, 4)$, while sparse configurations with $(n_O, n_N) = (1, 1)$ terminate fastest. Dashes in Table III indicate non-comparable aggregates for early-terminating RISKY runs rather than missing data.

We illustrate the verification outcomes with two representative cases from experiments with grid size $100\times100\times100$. An UNSAT instance from the SAFE region and a SAT instance from the RISKY region. Each case specifies the bounded environment profile, the solver outcome, and its geometric interpretation.

a) UNSAT case (SAFE region, $100\times100\times100$): A bound profile with $(n_O, n_N) = (2, 2)$ and horizon $K = 30$ was evaluated. The obstacle and NFZ parameters were constrained to SAFE ranges that lie above the UAV trajectory corridor. Z3 returned UNSAT, indicating that no admissible placement of two obstacles and two NFZs within these bounds can violate the mission constraints. Geometrically, all constraints satisfy $y^{\min} \geq 38$, placing them entirely above the horizontal arm $(y = 0)$ and beyond the turning region of the trajectory. The UAV trajectory traverses $y \in \{0, 4, 16, 28\}$ along the vertical segment, all of which lie strictly below the minimum y-bound of any constraint. Therefore, no intersection between the trajectory and environment constraints is possible. This

TABLE III
VERIFICATION OUTCOMES ACROSS GRID SIZES AND ENVIRONMENT REGIONS.

Grid	Region	#Total	#SAT	#UNSAT	TO	Avg RT (s)	**Longest (s)**	(n_O, n_N)	**Shortest (s)**	(n_O, n_N)
$25\times25\times25$	SAFE	39,051	0	39,050	1	0.200–0.64	**1.84**	(4,4)	**0.035**	(1,1)
$25\times25\times25$	RISKY	—	0	—	—	0.060–0.18	**0.44**	(4,4)	**0.035**	(1,1)
$50\times50\times50$	SAFE	18,089	0	18,088	1	0.410–1.97	**4.62**	(4,4)	**0.049**	(1,1)
$50\times50\times50$	RISKY	—	0	—	—	0.090–0.22	**0.50**	(4,4)	**0.049**	(1,1)
$75\times75\times75$	SAFE	7,908	0	7,908	0	0.140–2.37	**7.43**	(4,4)	**0.140**	(1,1)
$75\times75\times75$	RISKY	4,861	4,861	0	0	0.067–1.07	**2.20**	(4,4)	**0.067**	(1,1)
$100\times100\times100$	SAFE	3,643	0	3,643	—	1.060–5.43	**9.59**	(4,4)	**0.320**	(1,1)
$100\times100\times100$	RISKY	3,526	3,526	0	—	0.180–1.24	**2.30**	(4,4)	**0.077**	(1,1)
$125\times125\times125$	SAFE	2,285	0	2,285	1	0.680–9.42	**16.43**	(4,4)	**0.680**	(1,1)
$125\times125\times125$	RISKY	2,379	2,379	0	0	0.106–1.61	**3.04**	(4,4)	**0.106**	(1,1)

UNSAT result provides a bounded correctness guarantee over the entire admissible region of Θ, covering all configurations consistent with the specified bounds.

b) SAT case (RISKY region, $100\times100\times100$).: A bound profile with $(n_O, n_N) = (2, 1)$ and horizon $K = 14$ was evaluated under the RISKY parameter ranges. Z3 returned SAT in 0.41 s and produced a concrete counterexample. In the satisfying assignment, the NFZ is placed within the RISKY bounds so that its horizontal footprint intersects the initial nominal mission corridor. For example, the admissible choice $x_j^{\min} = 12$, $\Delta x_j = 8$, $y_j^{\min} = 0$, and $\Delta y_j = 6$ yields the NFZ footprint $x \in [12, 20]$ and $y \in [0, 6]$. Under the fixed policy, the bounded execution reaches a corridor point whose horizontal coordinates lie inside this NFZ footprint. Since NFZ membership is altitude-independent in the model, this state violates the safety requirement. Thus, the SAT assignment provides a concrete violation scenario. The rapid SAT runtime reflects the nature of satisfiable queries, where the solver terminates after finding one violating configuration.

VI. CONCLUSION AND FUTURE WORK

This paper presented an SMT-based framework for symbolic scenario coverage verification of single-UAV mission policies under bounded environmental assumptions. Unlike simulation-based validation, which evaluates a finite set of sampled scenarios, the proposed approach treats the environment as a symbolic parameter space, enabling exhaustive reasoning over all admissible configurations within specified bounds. The results demonstrate that safety is not uniformly guaranteed across environment configurations, but depends critically on the interaction between UAV motion, resource constraints, and environment geometry. The framework systematically identifies violating configurations through SAT counterexamples and provides bounded correctness guarantees through UNSAT results. Scalability analysis shows that solver performance is influenced by grid size and configuration density, with predictable increases in runtime for larger and denser environments. Nevertheless, the framework remains tractable for practical mission-level verification, with most queries completing within seconds and only limited timeouts at the largest grid scale. Future work will extend the framework along several directions. These include multi-UAV verification using compositional or assume–guarantee reasoning, incorporation of dynamic and uncertain environments, and refinement toward continuous or hybrid models.

REFERENCES

[1] H. T. Dinh and T. Holvoet, "Verifying autonomous decision making against environment assumptions: An experience report," in *Proc. IEEE Int. Conf. Robotic Computing (IRC)*, 2020, pp. 327–335.

[2] S. Zudaire, F. Gorostiaga, C. Sanchez, G. Schneider, and S. Uchitel, "Assumption monitoring using runtime verification for uav temporal task plan executions," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2021, pp. 6824–6830.

[3] M. Luckcuck, M. Farrell, L. Dennis, C. Dixon, and M. Fisher, "Formal specification and verification of autonomous robotic systems: A survey," *ACM Computing Surveys*, vol. 52, pp. 1–41, 2020.

[4] A. Mahzoon, D. Große, and R. Drechsler, "Revsca-2.0: Sca-based formal verification of nontrivial multipliers using reverse engineering and local vanishing removal," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 41, pp. 1573–1586, 2022.

[5] R. Drechsler, *Advanced formal verification*. Springer, 2004.

[6] A. M. Zbrzezny, O. Siedlecka-Lamch, S. Szymoniak, A. Zbrzezny, and M. Kurkowski, "Timed interpreted systems as a new agent-based formalism for verification of timed security protocols," *Applied Sciences*, vol. 14, no. 22, p. 10333, 2024.

[7] A. Rashid, U. Siddique, and S. Tahar, "Formal verification of cyber-physical systems using theorem proving," in *Formal Techniques for Safety-Critical Systems*. Springer, 2020, pp. 3–18.

[8] R. Bouchekir, M. Guzman, A. Cook, J. Haindl, and R. Woolnough, "Formal verification for safe ai-based flight planning for uavs," in *Proc. IEEE/IFIP Int. Conf. Dependable Systems and Networks Workshops (DSN-W)*, 2023, pp. 275–282.

[9] K. Nallan, S. Mishra, and A. A. Julius, "An assume-guarantee framework for multiple-obstacle collision avoidance," in *Proc. Vertical Flight Society Forum & Technology Display*, 2021.

[10] H. Kress-Gazit, G. E. Fainekos, and G. J. Pappas, "Temporal-logic-based reactive mission and motion planning," *IEEE Transactions on Robotics*, vol. 25, no. 6, pp. 1370–1381, 2009.

[11] L. R. Humphrey and M. J. Patzek, "Model checking human-automation uav mission plans," in *AIAA Guidance, Navigation, and Control Conference*, 2013.

[12] B. U. Kim and L. R. Humphrey, "Satisfiability checking of ltl specifications for verifiable uav mission planning," in *AIAA Aerospace Sciences Meeting*, 2014.

[13] L. R. Humphrey, E. M. Wolff, and U. Topcu, "Formal specification and synthesis of mission plans for unmanned aerial vehicles," in *AAAI Spring Symposium Series*, 2014.

[14] C. M. noz and A. Narkawicz, "Formal analysis of extended well-clear boundaries for unmanned aircraft," in *NASA Formal Methods*, ser. Lecture Notes in Computer Science, vol. 9690, 2016, pp. 221–226.

[15] M. Lahijanian, S. B. Andersson, and C. Belta, "Temporal logic motion planning and control with probabilistic satisfaction guarantees," *IEEE Transactions on Robotics*, vol. 28, no. 2, pp. 396–409, 2012.

[16] B. Pollien, C. Garion, G. Hattenberger, P. Roux, and X. Thirioux, "Verifying the mathematical library of a uav autopilot with frama-c," in *Formal Methods for Industrial Critical Systems (FMICS)*, 2021.

[17] ——, "A verified uav flight plan generator," in *IEEE/ACM Int. Conf. Formal Methods in Software Engineering (FormaliSE)*, 2023.