From Cold Start to Critical: Formal Synthesis of Hybrid Controllers

PI: Dane A. Sabo
dane.sabo@pitt.edu

Advisor: Dr. Daniel G. Cole
dgcole@pitt.edu

Monday 22nd September, 2025

1 Goals and Outcomes

The goal of this research is to develop a unified framework combining temporal logic synthesis with continuous-time verification methods to create autonomous hybrid control systems with complete correctness guarantees. Hybrid control systems have great potential for autonomous control applications because they can switch between different control laws based on discrete triggers in the system's operating range. This approach allows autonomous controllers to use several tractable control laws optimized for different regions in the state space, rather than relying on a single controller across the entire operating range. But, the discrete transitions between control laws in hybrid controllers present significant challenges in proving stability and liveness properties for the complete system. While tools from control theory can establish properties for individual control modes, these guarantees do not generalize when mode switching is introduced. Conversely, significant advances in formal methods have enabled automatic synthesis of discrete controllers from temporal logic specifications—tools like Strix can generate provably correct switching logic for complex logical requirements. However, these synthesis approaches assume instantaneous mode transitions and operate purely in discrete state spaces. In hybrid systems, transitions occur along continuous trajectories governed by differential equations, creating a fundamental verification gap that neither purely discrete synthesis nor traditional control theory can address alone.

This research addresses a fundamental challenge in hybrid controller synthesis and verification by unifying discrete system synthesis with continuous system analysis. We will leverage formal methods to create controllers that are correct-by-construction, enabling guarantees about the complete system's behavior. To demonstrate this approach, we will develop an autonomous controller for nuclear power plant start-up procedures. Nuclear power represents an excellent test case because the continuous reactor dynamics are well-studied, while the discrete mode switching requirements are explicitly defined in regulatory procedures and operating guidelines. Current nuclear reactor control *is* already a hybrid system. For example, during reactor startup, operators must transition from initial cold conditions through controlled heating phases to predetermined power levels. Each phase employs different automated controllers: temperature ramp controllers during heatup, reactivity controllers approaching criticality, and load-following controllers during operation. The decision of when to switch between these controllers currently relies on human operators interpreting written procedures. Our approach would formalize such transition conditions and synthesize the switching logic automatically.

The capability to create high-assurance hybrid control systems has significant potential to reduce labor costs in operating critical systems by removing human operators from control loops. Nuclear power stands to benefit substantially from increased controller autonomy, as operations and maintenance represent the largest expense for current reactor designs. While emerging technologies such as microreactors and small modular reactors will reduce maintenance costs through factory-manufactured replacement components, they face increased per-megawatt operating costs if required to maintain traditional staffing levels. However, if increased autonomy can be safely introduced, these economic challenges can be addressed while maintaining safety standards.

If this research is successful, we will achieve the following outcomes:

1. **Formalize mode switching requirements as logical specifications that can be synthesized into discrete controller implementations.** The discrete transitions between continuous controller modes are often explicitly defined in operating procedures and regulatory requirements for critical systems. These natural language requirements will be translated

into temporal logic specifications, which will then be synthesized into provably correct discrete controllers for continuous mode switching.

2. **Develop and verify formal characterizations of hybrid mode dynamics and safety conditions.** We will establish mathematical frameworks distinguishing transitory modes with reachability requirements to target states from stabilizing modes with invariant maintenance properties. For linear dynamics, classical control theory will establish stability and performance within each mode. For nonlinear systems, reachability analysis will verify that transitory modes drive the system toward intended transitions while maintaining safety constraints, and that stabilizing modes preserve their designated operating regions. This unified approach will enable provable conditions for safe state space traversal and transition timing.
3. **Prove that hybrid system implementations achieve safety and performance specifications across operational mode sequences.** By synthesizing discrete controller transitions from logical specifications using correct-by-construction methods and verifying that continuous components perform appropriately between discrete transitions, we can establish mathematical guarantees that the hybrid system maintains safety constraints and meets performance requirements during autonomous operational sequences such as reactor startup procedures, where multiple control modes must be coordinated to achieve higher-level operational objectives.

2 State of the Art and Limits of Current Practice

This research aims to advance high-assurance autonomous hybrid control systems by bridging disparate approaches from control theory and computer science. While both fields tackle hybrid system verification, they approach the problem from fundamentally different perspectives. Control theory emphasizes performance and stability, while computer science focuses on correctness through formal verification. This gap represents both the primary challenge and the key opportunity for intellectual contribution.

2.1 Control Theory and Hybrid Systems

Hybrid systems combine continuous dynamics ('flows') with discrete transitions ('jumps'). Following the standard formulation, a hybrid system can be expressed as:

$$\dot{x}(t) = f(x(t), q(t), u(t)) \tag{1}$$

$$q(k+1) = \nu(x(k), q(k), u(k)) \tag{2}$$

Here, $f(\cdot)$ defines the continuous dynamics while $\nu(\cdot)$ governs discrete transitions. The continuous states x, discrete state q, and control input u interact to produce hybrid behavior. The discrete state q defines the current continuous dynamics mode. The only constraint on $\nu(\cdot)$ is avoiding Zeno behavior—infinite jumps in finite time. Our focus centers on continuous autonomous hybrid systems, where continuous states remain unchanged during jumps and no external control input is required. Physical systems naturally exhibit this property—a nuclear reactor switching from warm-up to load-following control cannot instantaneously change its temperature or rod position.

An intuitive approach to building hybrid controllers is to stitch together multiple simple controllers for different state space regions. This approach creates significant verification challenges. Even with linear time-invariant modes, global stability cannot be guaranteed using linear control theory alone. Instead, researchers have developed Lyapunov-based approaches, though finding appropriate Lyapunov functions remains challenging. Stability conditions for switched linear systems can provide necessary and sufficient conditions using multiple Lyapunov functions [1], but these methods require restrictive assumptions. Individual Lyapunov functions must be monotonically nonincreasing at every switching time, which proves impractical for many real systems. Common Lyapunov functions face even stricter existence conditions, often impossible for systems with fundamentally different dynamics across modes [2, 3].

LIMITATION: Lyapunov-based methods of ensuring stability are prohibitively challenging to apply to hybrid control systems. Finding Lyapunov functions to prove stability in the sense of Lyapunov is not practical for working with hybrid systems. Additional requirements on the Lyapunov functions makes finding appropriate functions extremely difficult.

Reachability analysis offers an alternative verification approach by computing system output ranges for given inputs. Unlike Lyapunov methods, reachability extends naturally to nonlinear systems. Hamilton-Jacobi frameworks established the mathematical foundation for computing reachable sets in continuous and hybrid systems, enabling formal verification of safety-critical applications [4].

LIMITATION: Reachability analysis is not scalable to large hybrid systems. Reachability analysis faces two critical limitations. First, computational complexity grows exponentially with state dimension—current methods remain limited to 6-8 dimensional systems despite recent algorithmic advances. Second, approximation errors compound over long time horizons, potentially

rendering analysis meaningless for extended trajectories. Zonotope-based methods suffer accuracy degradation when propagating across mode boundaries, while ellipsoidal methods produce conservative over-approximations that become increasingly pessimistic with each mode transition.

Recent work on using control barrier functions has improved hybrid system verification efforts. Neural network based control barrier function approximations achieve 10-100X speedup over classical Hamilton-Jacobi methods while maintaining safety guarantees for 7-dimensional autonomous racing systems [5]. This breakthrough demonstrates that deep neural networks can approximate Hamilton-Jacobi partial differential equations and enables application to previously intractable high-dimensional systems, but future work must address the explainability problem for neural networks to ensure control barrier functions are valid.

2.2 Formal Methods and Reactive Synthesis

Correctness requirements are specified using temporal logic, which captures system behaviors through temporal relations. Linear Temporal Logic (LTL) provides four fundamental operators: next (X), eventually (F), globally (G), and until (U). Consider a nuclear reactor SCRAM requirement:

Natural language: "If a high temperature alarm triggers, control rods must immediately insert and remain inserted until operator reset."

This plain language requirement can be translated into a rigorous logical specification.

LTL specification:

$$\mathsf{G}(\text{HighTemp} \rightarrow \mathsf{X}(\text{RodsInserted} \wedge (\neg\text{RodsWithdrawn}\ \mathsf{U}\ \text{OperatorReset}))) \tag{3}$$

Once requirements are translated into these logical specifications, they can be checked using computational tools. Cyber-physical systems naturally exhibit discrete behavior amenable to formal analysis. Systems with finite states can be modeled as finite state machines, where all possible states and transitions are explicitly enumerable as logical specifications. This enables exhaustive verification through model checking—a technique extensively employed in high-assurance digital systems and safety-critical software. This mathematical framework has been extended to hybrid automata [6], which established the standard model combining discrete control graphs with continuous dynamics. Hybrid automata bridge program-analysis techniques to hybrid systems with infinite state spaces through symbolic model-checking based on reachability analysis [7] but are not scalable for the same limitations mentioned earlier with other reachability techniques.

NASA's Formal Requirements Elicitation Tool (FRET) bridges natural language and mathematical specifications through FRETish—a structured English-like language automatically translatable to temporal logic [8]. FRET enables hierarchical requirement organization, realizability checking for specification conflicts, integration with verification tools like CoCoSim, and runtime monitoring through their Copilot tool.

From realizable specifications, reactive synthesis tools automatically generate controllers. The Reactive Synthesis Competition (SYNTCOMP) has driven algorithmic improvements for over a decade, with tools like Strix dominating recent competitions through efficient parity game solving [9,10]. Strix is able to translate linear temporal logic specifications into deterministic automata automatically while maximizing generated automata quality.

LIMITATION: Unexplored application of temporal logic requirements in nuclear con-

trol. Despite extensive nuclear power documentation and mature reactive synthesis tools, little work has combined these for nuclear control applications. Nuclear procedures are written in structured natural language that maps well to temporal logic, yet the synthesis community has not engaged with this domain. This represents a significant unexplored opportunity where formal methods could provide immediate practical impact.

Hybrid automata extend finite automata by representing discrete states as control modes with continuous dynamics. Transitions between nodes indicate discrete state changes through $v(\cdot)$ execution. This provides intuitive graphical representation of mode switching logic. Differential dynamic logic (dL) expands this idea and offers the most complete logical foundation for hybrid verification. dL introduced two key modalities [11, 12]:

- Box modality $[\alpha]\phi$: for all executions of hybrid system α, property ϕ holds
- Diamond modality $\langle\alpha\rangle\phi$: there exists an execution of α where ϕ holds

These modalities enable direct reasoning about hybrid systems including continuous dynamics. The KeYmaera X theorem prover implements dL through axiomatic tactical proving, successfully verifying collision avoidance in train and aircraft control [13].

LIMITATION: There is a high expertise barrier for dL verification, and scalability remains an issue. While dL is expressive enough for any hybrid behavior, verification requires expertise in differential equations, logical specifications, and sequent calculus. The proof effort remains challenging even with automated assistance. Users must understand both the mathematical intricacies of their system and the logical framework, then guide the prover through complex proof steps. This expertise barrier prevents wider adoption despite the framework's theoretical completeness.

The state of the art reveals a field in transition. Traditional boundaries between control theory and formal methods are dissolving through learning-based approaches and compositional techniques. While fundamental challenges remain—particularly scalability and the theory-practice gap—the convergence of approaches promises to enable verification and synthesis for systems of unprecedented complexity. The next section describes how this research will contribute to bridging these gaps through unified synthesis frameworks applied to nuclear reactor control.

3 Research Approach

This research will overcome the limitations of current practice to build high-assurance hybrid control systems for critical infrastructure. To achieve this goal, we must accomplish three main thrusts:

1. Translate operating procedures and requirements into temporal logic formulae
2. Create the discrete half of a hybrid controller using reactive synthesis
3. Develop continuous controllers to operate between modes, and verify their correctness using reachability analysis, barrier certificates, and assume-guarantee contracts

The following sections discuss how these thrusts will be accomplished.

3.1 Requirements: $(Procedures \wedge FRET) \rightarrow TemporalSpecifications$

The motivation behind this work stems from the fact that commercial nuclear power operations remain manually controlled by human operators, despite significant advances in control systems sophistication. The key insight is that procedures performed by human operators are highly prescriptive and well-documented. This suggests that human operators in nuclear power plants may not be entirely necessary given today's available technology.

Written procedures and requirements in nuclear power are sufficiently detailed that we may be able to translate them into logical formulae with minimal effort. If successful, this approach would enable automation of existing procedures without requiring system reengineering—a significant advantage for deployment in existing facilities. The most efficient path to accomplish this translation is through automated tools such as NASA's Formal Requirement Elicitation Tool (FRET).

FRET employs a specialized requirements language similar to natural language called FRETish. FRETish restricts requirements to easily understood components while eliminating ambiguity. FRET enforces this structure by requiring all requirements to contain six components:

1. Scope: *What modes does this requirement apply to?*
2. Condition: *Scope plus additional specificity*
3. Component: *What system element does this requirement affect?*
4. Shall
5. Timing: *When does the response occur?*
6. Response: *What action should be taken?*

FRET provides functionality to check the *realizability* of a system. Realizability analysis determines whether written requirements are complete by examining the six structural components. Complete requirements are those that neither conflict with one another nor leave any behavior undefined. Systems that are not realizable from their procedure definitions and design requirements present problems beyond autonomous control implementation. Such systems contain behavioral inconsistencies that represent the physical equivalent of software bugs. Using FRET during autonomous controller development allows us to identify and resolve these errors systematically.

The second category of realizability issues involves undefined behaviors that are typically left to human judgment during control operations. This ambiguity is undesirable for high-assurance systems, since even well-trained humans remain prone to errors. By addressing these specification gaps in FRET during autonomous controller development, we can deliver controllers free from these vulnerabilities.

FRET also provides the capability to export requirements in temporal logic format. This export functionality enables progression to the next step of our approach: synthesizing discrete mode switching behavior from the formalized requirements.

3.2 Discrete Synthesis: $(TemporalLogic \wedge ReactiveSynthesis) \rightarrow DiscreteAutomata$

This section describes how the discrete component of the hybrid system will be constructed. The formal specifications created in FRET will be processed by reactive synthesis tools to generate mode switching components. These components effectively automate the human decision-making elements of reactor operation by implementing the logic typically specified in written procedures. By eliminating the human component from control decisions, we remove the possibility of human error and advance hybrid system autonomy.

Reactive synthesis is an active research field in computer science focused on generating discrete controllers from temporal logic specifications. The term "reactive" indicates that the system responds to environmental inputs to produce control outputs. These synthesized systems are finite in size, where each node represents a unique discrete state. The connections between nodes, called *state transitions*, specify the conditions under which the discrete controller moves from state to state. This complete mapping of possible states and transitions constitutes a *discrete automaton*. From the automaton graph, it becomes possible to fully describe the dynamics of the discrete system and develop intuitive understanding of system behavior. Once constructed, the automaton can be straightforwardly implemented using standard programming control flow constructs.

We will use discrete automata to represent the switching behavior of our hybrid system. This approach yields an important theoretical guarantee: because the discrete automaton is synthesized entirely through automated tools from design requirements and operating procedures, we can prove that the automaton—and therefore our hybrid switching behavior—is correct by construction.

Correctness of the switching controller is paramount to this work. Mode switching represents the primary responsibility of human operators in control rooms today. Human operators possess the advantage of real-time judgment—when mistakes occur, they can correct them dynamically with capabilities that extend beyond written procedures. Autonomous control lacks this adaptive advantage. Instead, we must ensure that autonomous controllers replacing human operators will not make switching errors between continuous modes. By synthesizing controllers from logical specifications with guaranteed correctness, we eliminate the possibility of switching errors.

3.3 $(DiscreteAutomata \wedge ControlTheory \wedge Reachability) \rightarrow ContinuousModes$

While discrete system components will be synthesized with correctness guarantees, they represent only half of the complete system. Autonomous controllers like those we are developing exhibit continuous dynamics within discrete states. This section describes how we will develop continuous control modes, verify their correctness, and address the unique verification challenges of hybrid systems.

First, we must address the limitation left unresolved in the previous section. The approach described for producing discrete automata yields physics-agnostic specifications that represent only half of a complete hybrid autonomous controller. These automata alone cannot define the full behavior of the control systems we aim to construct. The continuous modes will be developed after discrete automaton construction, leveraging the automaton structure and transitions to design multiple smaller, specialized continuous controllers.

The discrete automaton transitions are key to the supervisory behavior of the autonomous con-

troller. These transitions mark decision points for switching between continuous control modes and define their strategic objectives. We will classify three types of high-level continuous controller objectives based on discrete mode transitions:

1. **Stabilizing:** A stabilizing control mode has one primary objective: maintaining the hybrid system within its current discrete mode. This corresponds to steady-state normal operating modes, such as a full-power load-following controller in a nuclear power plant. Stabilizing modes can be identified from discrete automata as nodes with only incoming transitions.
2. **Transitory:** A transitory control mode has the primary goal of transitioning the hybrid system from one discrete state to another. In nuclear applications, this might represent a controlled warm-up procedure. Transitory modes ultimately drive the system toward a stabilizing steady-state mode. These modes may have secondary objectives within a discrete state, such as maintaining specific temperature ramp rates before reaching full-power operation.
3. **Expulsory:** An expulsory mode is a specialized transitory mode with additional safety constraints. Expulsory modes ensure the system is directed to a safe stabilizing mode during failure conditions. For example, if a transitory mode fails to achieve its intended transition, the expulsory mode activates to immediately and irreversibly guide the system toward a globally safe state. A reactor SCRAM exemplifies an expulsory continuous mode: when initiated, it must reliably terminate the nuclear reaction and direct the reactor toward stabilizing decay heat removal.

Building continuous modes after constructing discrete automata enables local controller design focused on satisfying discrete transitions. The primary challenge in hybrid system verification is ensuring global stability across transitions. Current techniques struggle with this problem because dynamic discontinuities complicate verification. This work alleviates these problems by designing continuous controllers specifically with transitions in mind. By decomposing continuous modes according to their required behavior at transition points, we avoid solving trajectories through the entire hybrid system. Instead, we can use local behavior information at transition boundaries.

To ensure continuous modes satisfy their requirements, we will employ three main techniques: reachability analysis, assume-guarantee contracts, and barrier certificates. Reachability analysis computes the reachable set of states for a given input set. While trivial for linear continuous systems, recent advances have extended reachability to complex nonlinear systems. We will use reachability to define continuous state ranges at discrete transition boundaries and verify that requirements are satisfied within continuous modes. Assume-guarantee contracts will be employed when continuous state boundaries are not explicitly defined. For any given mode, the input range for reachability analysis is defined by the output ranges of discrete modes that transition to it. This ensures each continuous controller is prepared for its possible input range, enabling subsequent reachability analysis. Finally, we will use barrier certificates to prove that mode transitions are satisfied. Barrier certificates ensure that continuous modes on either side of a transition behave appropriately. For example, a barrier certificate can guarantee that a transitory mode transferring control to a stabilizing mode will always move away from the transition boundary, rather than destabilizing the target mode. Combining these three techniques will enable us to prove that continuous components of our hybrid controller satisfy discrete requirements.

This unified approach addresses a fundamental gap in hybrid system design by bridging formal methods and control theory through a systematic, tool-supported methodology. By translating

existing nuclear procedures into temporal logic, synthesizing provably correct discrete switching logic, and developing verified continuous controllers, we create a complete framework for autonomous hybrid control with mathematical guarantees. The result is an autonomous controller that not only replicates human operator decision-making but does so with formal assurance that switching logic is correct by construction and continuous behavior satisfies safety requirements. This methodology transforms nuclear reactor control from a manually intensive operation requiring constant human oversight into a fully autonomous system with higher reliability than human-operated alternatives. More broadly, this approach establishes a replicable framework for developing high-assurance autonomous controllers in any domain where operating procedures are well-documented and safety is paramount.

References

[1] José C Geromel and Patrizio Colaneri. Stability and stabilization of continuous-time switched linear systems. *SIAM Journal on Control and Optimization*, 45(5):1915–1930, 2006.

[2] Michael S Branicky. Multiple lyapunov functions and other analysis tools for switched and hybrid systems. *IEEE Transactions on Automatic Control*, 43(4):475–482, 1998.

[3] Daniel Liberzon. *Switching in systems and control*. Birkhäuser Boston, 2003.

[4] Ian M Mitchell, Alexandre M Bayen, and Claire J Tomlin. A time-dependent hamilton-jacobi formulation of reachable sets for continuous dynamic games. *IEEE Transactions on Automatic Control*, 50(7):947–957, 2005.

[5] Shuo Yang, Yiwei Chen, Xiang Yin, and Rahul Mangharam. Learning local control barrier functions for hybrid systems. *arXiv preprint arXiv:2401.14907*, 2024.

[6] Rajeev Alur, Costas Courcoubetis, Thomas A Henzinger, and Pei-Hsin Ho. Hybrid automata: An algorithmic approach to the specification and verification of hybrid systems. In *Hybrid Systems*, pages 209–229. Springer, 1993.

[7] Rajeev Alur, Costas Courcoubetis, Nicolas Halbwachs, Thomas A Henzinger, Pei-Hsin Ho, Xavier Nicollin, Alfredo Olivero, Joseph Sifakis, and Sergio Yovine. The algorithmic analysis of hybrid systems. *Theoretical Computer Science*, 138(1):3–34, 1995.

[8] Dimitra Giannakopoulou, Anastasia Mavridou, Julian Rhein, Thomas Pressburger, Johann Schumann, and Nija Shi. Capturing and analyzing requirements with fret. Technical Report NASA/TM-20220007610, NASA Ames Research Center, 2022.

[9] Philipp J Meyer and Michael Luttenberger. Strix: Explicit reactive synthesis strikes back! In *International Conference on Computer Aided Verification*, pages 578–586. Springer, 2018.

[10] Swen Jacobs, Roderick Bloem, Romain Brenguier, et al. The 4th reactive synthesis competition (syntcomp 2017): Benchmarks, participants & results. In *6th Workshop on Synthesis*, volume 260 of *EPTCS*, 2017.

[11] André Platzer. Differential dynamic logic for hybrid systems. *Journal of Automated Reasoning*, 41(2):143–189, 2008.

[12] André Platzer. A complete uniform substitution calculus for differential dynamic logic. *Journal of Automated Reasoning*, 59(2):219–265, 2017.

[13] Nathan Fulton, Stefan Mitsch, Jan-David Quesel, Marcus Völp, and André Platzer. Keymaera x: An axiomatic tactical theorem prover for hybrid systems. In *International Conference on Automated Deduction*, pages 527–538. Springer, 2015.