FAR-STL: Filter-Aware Robust Signal Temporal Logic for Online Monitoring of Cyber-Physical Systems

Supratim Gupta, Sobhan Chatterjee, Naijun Zhan, and Partha Roop

***Abstract*—Monitors of Signal Temporal Logic (STL) assume that the sensor-generated signals are of high-fidelity and continuous-time in nature. In practice, however, signals are sampled, digitally filtered, and corrupted by hardware noise—artefacts that the monitor ignores. Designing the filter independently of the monitor produces unreliable verdicts: under-filtering admits noise-induced false negatives, while over-filtering introduces a group delay that causes the monitor to evaluate stale data.**

We present Filter-Aware Robust STL (FAR-STL), a framework in which filter parameters are considered as *first-class artifacts*. FAR-STL restricts monitoring to a *valid aligned trace*—the steady-state window of the filtered signal—and scales robustness by a trust factor τ derived from the Signal to Noise Ratio (SNR) gain and sampling density. Crucially, the spatial safety guard (ϵ) and trust factor (τ) are *functions of the filter design parameters* enabling a joint synthesis loop that maximises robustness. We establish a (Δ^*, ϵ)-bisimilarity between the filtered discrete trace and the continuous ground truth, and prove that a positive FAR-STL verdict implies satisfaction of the original continuous-time STL formula. Evaluation on three published case studies—Adaptive Cruise Control (ACC) car following, cardiac electrogram monitoring, and drone flight control—demonstrates that joint synthesis is essential: a decoupled design either fails the SNR requirement or introduces unacceptable monitoring latency, both of which are highly undesirable in Cyber Physical Systems (CPS).

***Index Terms*—Signal Temporal Logic, Digital Filter Synthesis, Formal Verification, Hardware-Software Co-Design, Cyber-Physical Systems, Embedded Software.**

I. Introduction

Embedded Cyber-Physical Systems (CPS) in safety-critical domains—autonomous vehicles, industrial robots, medical devices—must satisfy formal safety specifications to meet standards such as IEC 61508 [1] and ISO 26262 [2]. Signal Temporal Logic (STL) [3] is the dominant formalism for this purpose: it allows engineers to express timing and magnitude requirements over real-valued signals, and its robustness degree ρ provides a quantitative safety margin that is amenable to both offline verification and online monitoring [4].

S. Gupta is with the University of Auckland, Auckland, New Zealand, and the National Institute of Technology Rourkela, Rourkela, India (on leave). E-mail: supratim.gupta@auckland.ac.nz

S. Chatterjee is with the University of Auckland, Auckland, New Zealand. E-mail: sobhan.chatterjee@auckland.ac.nz.

N. Zhan is with Peking University, Beijing, China. E-mail: znj@ios.ac.cn

P. Roop is with the University of Auckland, Auckland, New Zealand. E-mail: p.roop@auckland.ac.nz

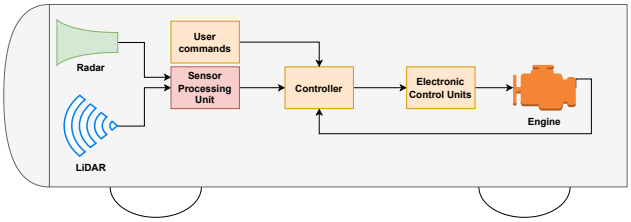

Fig. 1: Motivating ACC scenario: a sensor processing unit (SPU) condition the raw Radar/LiDAR data before the ECU evaluates the STL property $\phi = \Box_{[0,5]}(d > 2 \wedge d < 4)$

A. The Problem: Three Gaps in Existing STL Monitoring

Despite the maturity of STL monitoring, existing frameworks share three assumptions that fail in embedded deployments (see Section II for detailed comparison):

- **Full signal a priori.** Existing monitors assume the complete trace is available before evaluation. Online CPS monitoring requires verdicts as samples arrive; the valid evaluation window must account for both the formula horizon and the hardware-induced group delay k, a coupling that no prior framework formalises.
- **Sampling step fixed by the user.** The sampling period Δ controls the discretisation error, group delay, and Nyquist constraint simultaneously, yet the sampling frequency $F_s = 1/\Delta$ is treated as a user-supplied constant. No principled method jointly selects F_s with the filter to keep all three effects within formally bounded limits.
- **Filtering artefacts unaccounted for.** Real sensors operate well below the SNR required for formal monitoring (Table I). Digital filtering is the standard remedy, but it introduces passband ripple δ_p, group delay k, and residual noise σ—none of which are tracked by current STL frameworks. Existing uncertainty-aware approaches treat noise as a *passive, fixed* quantity and do not co-design the filter.

The interplay of these gaps is illustrated in Figure 2: under-filtering (σ too large) causes spurious violations; over-filtering (k too large) causes the monitor to evaluate stale data.

B. Our Approach: FAR-STL

We propose **Filter-Aware Robust STL (FAR-STL)**, a framework that addresses all three gaps by treating the digital filter as a *synthesis variable* rather than a fixed preprocessing black-box.

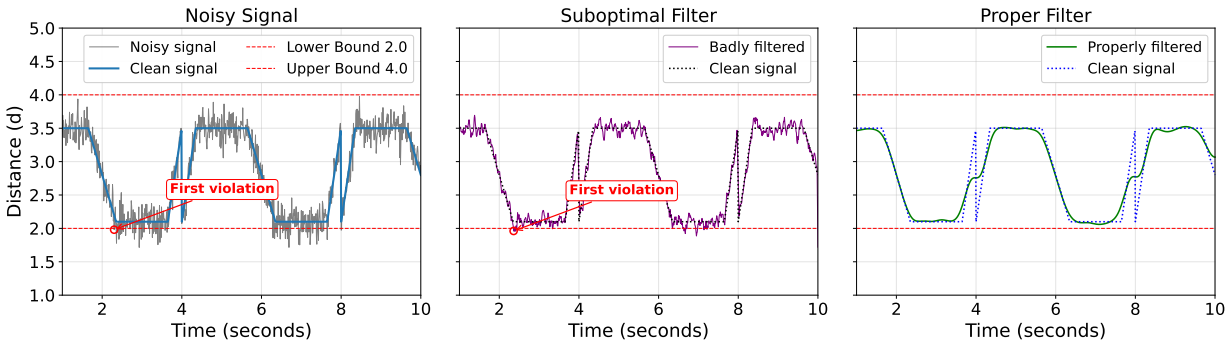

Fig. 2: Effect of noise and filter delay on STL satisfaction.

Fig. 3: Motivating ACC scenario: noise causes spurious violations (under-filtering) while the filter group delay causes the monitor to evaluate stale data (over-filtering).

TABLE I: Automotive sensor SNR specifications and verification targets [5], [6], [2].

Sensor	Std.	SNR_{in}	SNR_{tgt}	Req. gain
TDA4VM Radar	ETSI EN 301 091	75 dB	82 dB	≥ 7 dB
ARS548 Radar	FCC Part 95	82 dB	85 dB	≥ 3 dB
Velodyne LiDAR	ISO 26262	68 dB	80 dB	≥ 12 dB
Luminar LiDAR	ASIL-D	85 dB	88 dB	≥ 3 dB

SNR_{in}: Raw input quality; **SNR_{tgt}**: Target verification quality;
Req. gain: Filter noise suppression required to meet the target.

Formalisation. FAR-STL takes a standard STL formula ϕ together with hardware constraints (sensor SNR, ADC limits, reaction budget ν) and automatically derives a hardware-aware discrete-time variant φ. The spatial safety guard ϵ, a total uncertainty margin, and the trust factor $\tau(F_s, \text{SNR}_{\text{gain}})$ are *formal functions* of the design variables M (filter order), F_s, σ, δ_p. This makes FAR-STL not a new logic with higher expressiveness, but a *verified, hardware-aware instantiation of STL semantics*: for any single fixed hardware configuration, it instantiates a standard discrete-time STL formula (Lemma 1), but the framework automates and formally verifies that instantiation across the entire design space.

Synthesis pipeline. The pipeline (Figure 4) operates in two phases. *Phase 1* generates a library $\mathcal{H}$ of hardware-feasible (h, F_s) pairs that satisfy SNR and temporal alignment constraints. *Phase 2* evaluates each library entry on the valid aligned trace, selects the configuration maximising trust-weighted robustness $\tilde{\rho} = \rho \cdot \tau$, and returns the optimal design θ^*. A (Δ^*, ϵ)-bisimilarity theorem (Theorem 2) guarantees that a positive FAR-STL verdict on the discrete filtered trace implies satisfaction of the original continuous-time STL specification.

C. Contributions

1) **FAR-STL framework** (Sections III–IV): a formally defined discretisation of STL with filter parameters $(M, F_s, \delta_p, \sigma)$ as explicit variables, comprising a valid aligned trace, temporal horizon H_φ, valid evaluation domain $\mathcal{D}_\varphi$, trust-weighted robustness $\tilde{\rho}$, and a formal monitor with *currently safe/unsafe* verdicts for online use.
2) **STL–FAR-STL correspondence lemma** (Lemma 1, Section IV): under (Δ^*, ϵ)-bisimilarity, FAR-STL satisfaction is equivalent to STL satisfaction up to the ϵ spatial guard, formally establishing that FAR-STL is a sound and complete instantiation of STL over filtered discrete traces.
3) **Joint synthesis algorithm** (Section V): a two-phase pipeline that exhaustively searches the hardware-feasible design space and returns the filter–sampling configuration with maximum verified robustness.
4) **Formal soundness guarantee** (Theorem 2, Section V): a (Δ^*, ϵ)-bisimilarity theorem with a structured proof covering soundness (Point 1), algorithm termination (Point 2), and completeness (Point 3).

The remainder of the paper is organised as follows. Section II reviews related work. Section III provides STL and signal processing background. Section IV introduces FAR-STL formally and states the STL–FAR-STL correspondence. Section V presents the synthesis pipeline and soundness proof. Section VI evaluates the framework on three case studies: ACC car following, cardiac electrogram monitoring, and drone flight control. Section VII concludes.

II. Related Work

STL monitoring and robustness. Maler and Nickovic [7] introduced STL monitoring for continuous-time signals; Donzé and Maler [8] added the quantitative robustness semantics that underpins this work. The foundational toolsets **Breach** [9] and **S-TaLiRo** [10] established the paradigms for parameter synthesis and requirement falsification. Efficient online and offline monitoring algorithms have since been developed [11], [12], including compositional frameworks for algebraic quantitative monitoring [13] and satisfiability-checking methods based on tree-shaped tableaux [14]. Optimisation-based extensions, such as input-signal-space optimisation in S-TaLiRo [15] and time-staging enhancement [16], have significantly improved the efficiency of search-based testing. All of these works assume access to a clean, high-fidelity signal trace: signal quality and the hardware pipeline that produces it are outside the scope of the formal reasoning. The runtime verification survey in [4] treats signals as given inputs with no connection to the sensing hardware.

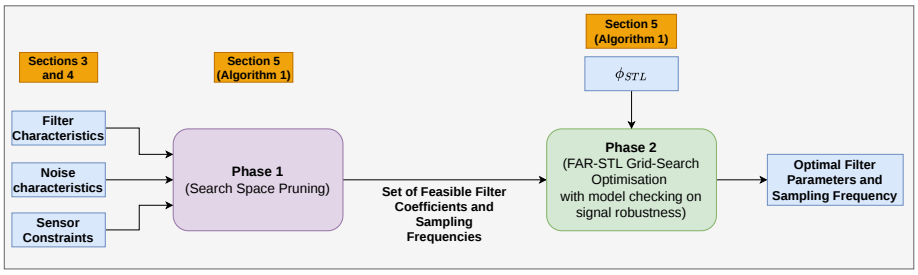

Fig. 4: FAR-STL two-phase synthesis pipeline. Phase 1 builds a hardware-feasible filter library $\mathcal{H}$. Phase 2 selects the optimal design θ^* by maximising trust-weighted robustness subject to the soundness guard.

Spatio-Temporal Foundations (STREL & SaSTL). Spatio-Temporal Reach and Escape Logic (STREL) [17] and SaSTL [18] establish standards for reasoning about networked CPS agents, introducing spatial operators over dynamic graphs [19], [20], [21]. While FAR-STL leverages these benchmarks for evaluation in scenarios such as automotive platooning, it goes beyond them by formally embedding FIR filter parameters into the semantics. This ensures that spatial verdicts—such as a "somewhere" or "everywhere" property—are not corrupted by digital artefacts like group delay or passband ripple, which are typically ignored in high-level spatio-temporal logic.

Passive vs. active uncertainty handling. Several works extend STL to handle trace uncertainty, but do so *passively*: they widen the semantic margin to accommodate a fixed, pre-determined noise level. While Interface-aware STL (IaSTL) [22] establishes a foundation for signal distance reasoning, its primary objective is assessing performance through input vacuity (ν) and output robustness (μ). Existing works [23], [21], [24] often utilise interval or three-valued logic to represent noise-induced uncertainty. FAR-STL is distinct because it handles uncertainty *actively*; instead of merely accepting a noise-induced "unknown" interval, it jointly synthesises the perception pipeline (coefficients h and rate F_s) to reduce noise and minimise the uncertainty interval before monitoring. By synthesising digital filters and applying a spatial safety guard (ϵ), FAR-STL formally bounds the cumulative effects of sampling discretisation, passband distortion, and residual noise. Consequently, while FAR-STL adapts the absolute and relative distance concepts from IaSTL, it uniquely utilises them to establish the (Δ^*, ϵ)-bisimilarity mapping necessary for robust digital filtering.

Discrete-time and sampled STL. Discrete-time variants of STL have been studied since the original work of Maler and Nickovic [7]. Synchronous monitoring of Lustre signals [25] provides a native discrete-time semantics with verified equivalence to continuous-time STL under a signal-invariance hypothesis. FAR-STL differs from these works in two respects: (i) it makes the *sampling frequency itself a synthesis variable*—the temporal horizon H_φ and valid evaluation domain $\mathcal{D}_\varphi$ are explicit functions of F_s and filter order M—and (ii) it explicitly models the group delay k to ensure that the evaluation window is aligned to the physical timeline. These features are essential for online monitoring in resource-constrained embedded systems, where F_s is not a free parameter. However, they must be selected jointly with the filter to meet reaction-time and SNR budgets.

Filter design for CPS. Digital filter design for embedded sensing is a mature field [26]. Observer-based verification for synchronous models [27], [28] integrates control-theoretic observers with formal properties, but does not address STL semantics over filtered traces. Probabilistic perception pipelines for autonomous driving [29] characterise sensor uncertainty but do not formalise the link to STL verdicts. To our knowledge, FAR-STL is the first framework that formally embeds FIR filter parameters into STL semantics and provides a verified joint synthesis algorithm with a soundness proof connecting discrete monitor verdicts to continuous-time ground truth.

Runtime Verification. Runtime Verification (RV) provides a formal, scalable method for analysing individual behaviours in Cyber-Physical Systems (CPS). Modern RV builds upon foundational quantitative semantics established by **Breach** [9] and **S-TaLiRo** [10], which provide the robustness "engines" necessary for real-time monitoring. Industrial workflows have successfully integrated these paradigms to detect complex feature interactions in intelligent vehicles [30]. To meet high-throughput demands, hardware-accelerated monitors have been implemented using FPGAs [31] and Automata Processors [32]. Recent work has extended these techniques to distributed environments using formula progression to manage clock drift [19]. More recently, the field has pivoted toward predictive runtime verification (PRV) [33] and conformal prediction (CP) to offer statistical guarantees under distribution shifts [34], [20] or out-of-distribution scenarios [35].

In contrast to these traditional and predictive frameworks, FAR-STL treats the perception pipeline as a synthesisable component rather than a fixed constraint. While Breach and S-TaLiRo focus on the logical evaluation of signal traces, FAR-STL's explicit modelling of group delay k, and discretisation error Δ^* ensures that the monitor is physically aligned with the continuous-time ground truth. This prevents cases where distorted digital traces might mask physical-layer violations—a risk that high-level monitoring tools generally do not address. Through its (Δ^*, ϵ)-bisimilarity mapping, FAR-STL provides a formal "Soundness Certificate" for implementation that current high-performance monitoring frameworks lack.

III. Background

A. Signal Temporal Logic

STL [7] specifies properties of real-valued continuous-time signals. xLet $\mathbf{x}(t) = (x_1(t), \ldots, x_p(t))$ denote a p-dimensional continuous-time signal with $t \in \mathbb{R}_{\geq 0}$.

Definition 1 (STL Syntax)**.** *An STL formula ϕ is defined by the grammar*

$$\phi ::= \top \mid \mu \mid \neg\phi \mid \phi_1 \wedge \phi_2 \mid \phi_1 \,\mathcal{U}_{[t_1,t_2]}\, \phi_2 \tag{1}$$

where μ is an atomic predicate $f(\mathbf{x}(t)) > 0$ for a function f satisfying the local Lipschitz condition, and $[t_1, t_2]$ with $0 \leq t_1 < t_2 < \infty$ is the temporal interval. Derived operators: $\Diamond_{[t_1,t_2]}\phi \equiv \top\, \mathcal{U}_{[t_1,t_2]}\, \phi$ and $\Box_{[t_1,t_2]}\phi \equiv \neg\Diamond_{[t_1,t_2]}\neg\phi$.

Definition 2 (STL Boolean Semantics)**.** *The satisfaction relation $(\mathbf{x}, t) \models \phi$ is defined recursively:*

$$(\mathbf{x}, t) \models \mu \iff f(\mathbf{x}(t)) > 0 \tag{2}$$

$$(\mathbf{x}, t) \models \neg\phi \iff (\mathbf{x}, t) \not\models \phi \tag{3}$$

$$(\mathbf{x}, t) \models \phi_1 \wedge \phi_2 \iff (\mathbf{x}, t) \models \phi_1 \text{ and } (\mathbf{x}, t) \models \phi_2 \tag{4}$$

$$(\mathbf{x}, t) \models \phi_1 \,\mathcal{U}_{[t_1,t_2]}\, \phi_2 \iff \exists\, t' \in [t{+}t_1, t{+}t_2] \text{ s.t. } (\mathbf{x}, t') \models \phi_2 \text{ and } \forall\, t'' \in [t, t'),\ (\mathbf{x}, t'') \models \phi_1 \tag{5}$$

Definition 3 (STL Quantitative Semantics [8])**.** *The robustness degree $\rho(\phi, \mathbf{x}, t) \in \mathbb{R}$ satisfies $\rho > 0 \Leftrightarrow (\mathbf{x}, t) \models \phi$ and is defined recursively:*

$$\rho(\mu, \mathbf{x}, t) = f(\mathbf{x}(t)) \tag{6}$$

$$\rho(\neg\phi, \mathbf{x}, t) = -\rho(\phi, \mathbf{x}, t) \tag{7}$$

$$\rho(\phi_1 \wedge \phi_2, \mathbf{x}, t) = \min(\rho(\phi_1, \mathbf{x}, t),\ \rho(\phi_2, \mathbf{x}, t)) \tag{8}$$

$$\rho(\Box_{[t_1,t_2]}\phi, \mathbf{x}, t) = \inf_{s \in t+[t_1,t_2]} \rho(\phi, \mathbf{x}, s) \tag{9}$$

$$\rho(\Diamond_{[t_1,t_2]}\phi, \mathbf{x}, t) = \sup_{s \in t+[t_1,t_2]} \rho(\phi, \mathbf{x}, s) \tag{10}$$

$$\rho(\phi_1 \,\mathcal{U}_{[t_1,t_2]}\, \phi_2, \mathbf{x}, t) = \sup_{s \in t+[t_1,t_2]} \min\Big(\rho(\phi_2, \mathbf{x}, s), \tag{11}$$

$$\inf_{r \in [t,s]} \rho(\phi_1, \mathbf{x}, r)\Big) \tag{12}$$

The robustness function is 1-Lipschitz continuous with respect to the sup*-norm on signal space [8].*

Example 1 (Longitudinal Safety Contracts for ACC)**.** *Consider the following example of a cruise control system with two vehicles: p_1, v_1, a_1 are the position, velocity, and acceleration of the ego vehicle. p_2, v_2, a_2 are the position, velocity, and acceleration of the lead vehicle. The forward-motion requirement (1) follows the gap-recovery requirement (inspired by [36]) (2) is the timed STL policy instantiated in the Phase 2 gap monitor:*

1) $\Box(v_1 \geq 0 \wedge v_2 \geq 0)$
   Both vehicles move forward (never reverse).
2) *Let $g := p_1 - p_2$ denote the inter-vehicle gap (m) and let $d_{min} > 0$ be a fixed minimum-gap threshold (m). The* gap-recovery *contract is*

$$\phi_{gap} = \Box\Big((g < d_{min}) \rightarrow \Diamond_{[0.5,\,1.5]}(g \geq d_{min})\Big).$$

Within each $100\,ms$ window, if the gap is at most d_{min}, then within 0.5–$1.5\,s$ the gap must be at least d_{min} again. The time limit 0.5–$1.5\,s$ is chosen based on human reaction time reported in literature.

We will use this example throughout the paper to illustrate the concepts covered.

B. Signal Processing Background

We introduce the signal processing concepts required for the FAR-STL framework.

Sampling. A continuous-time signal $\mathbf{x}(t)$ is sampled at frequency F_s Hz to produce the discrete sequence $\mathbf{x}[n] := \mathbf{x}(n\Delta)$, where $\Delta = 1/F_s$ is the Nyquist sampling period. The *signal frequency* $F_{\text{signal}} \leq F_s/2$ is the highest frequency or bandwidth of the signal at the sensor that can be faithfully represented without aliasing; sampling at $F_s \geq 2F_{\text{signal}}$ is required by the Shannon–Nyquist theorem.

FIR filtering. A Finite Impulse Response (FIR) filter of *order* M (even, Type I) has $M{+}1$ symmetric coefficients and transforms a discrete input sequence $\tilde{\mathbf{x}}[n]$ to an output $\hat{\mathbf{x}}[n]$ by

$$\hat{\mathbf{x}}[n] = \sum_{m=0}^{M} h[m]\, \tilde{\mathbf{x}}[n-m] \tag{13}$$

where $h = (h[0], \ldots, h[M])$ are the filter coefficients (e.g. $M = 34$ yields 35 taps). A Type I FIR filter has a *linear phase response*, ensuring that all frequency components experience the same constant delay.

Group delay. The *group delay* of a linear-phase FIR filter of order M is $k_{\text{fir}} = \lfloor M/2 \rfloor$ samples. Including computational overhead T_{calc} (ADC conversion, FIR critical path, logic evaluation), the total shift is

$$k = \left\lfloor \frac{M}{2} \right\rfloor + \lceil T_{\text{calc}} \cdot F_s \rceil\,. \tag{14}$$

A shift of k samples means the filter output at index n corresponds to the physical time $t = (n-k)\Delta$.

Passband distortion. The *passband ripple* δ_p is the maximum amplitude deviation within the filter's passband: $\delta_p = \max_{\omega \in \Omega_{\text{pass}}} |1 - |H(e^{j\omega})||$. It bounds the per-sample distortion between the true signal and the filter output.

SNR and SNR gain. The *Signal-to-Noise Ratio* (SNR) in decibels measures the ratio of signal power to noise power. A filter that attenuates out-of-band noise provides a theoretical *SNR gain*

$$\text{SNR}_{\text{gain}} = 10 \log_{10}\left(\frac{1}{\sum_{m=0}^{M} |h[m]|^2}\right) \text{dB}, \tag{15}$$

under the assumption that out-of-band noise is fully suppressed. The output SNR is $\text{SNR}_{\text{out}} = \text{SNR}_{\text{in}} + \text{SNR}_{\text{gain}}$.

Signal model. The noisy sensor output is modelled as $\tilde{\mathbf{x}}(t) = \mathbf{x}(t) + \eta(t)$, where $\mathbf{x}(t)$ is the true signal and $\eta(t) \sim \mathcal{N}(0, \sigma^2)$ is additive Gaussian noise (justified by the Central Limit Theorem for combined noise sources [37]). Although Gaussian noise is unbounded in theory, it is bounded in practice by the finite dynamic range of the ADC; synthesised FIR filters reduce stationary noise variance, keeping ϵ

a reliable per-configuration deterministic bound. At low SNR beyond hardware capacity, FAR-STL withholds a definitive `SAFE` verdict rather than issuing a false-safe certificate. We assume $\mathbf{x}(t)$ is Lipschitz continuous with constant $L = 1$.

IV. Filter-Aware Robust STL (FAR-STL)

FAR-STL is a *hardware-aware instantiation framework* for STL. It does not change the expressiveness of STL; rather, it makes the filter design variables $(M, F_s, \sigma, \delta_p)$ as explicit formal objects so that the spatial safety guard ϵ and trust factor τ can be computed automatically for any candidate hardware configuration, and a verified soundness bridge to the continuous ground truth is provided.

Definition 4 (Valid Aligned Trace)**.** *Let $\tilde{\mathbf{x}}[n]$ be the noisy discrete signal sampled at frequency F_s. Let $\hat{\mathbf{x}}[n] = \sum_{m=0}^{M} h[m]\,\tilde{\mathbf{x}}[n-m]$ be the output of the FIR filter with coefficients h and group delay k as in* (14)*. The* valid aligned trace *is the subsequence*

$$\hat{\mathbf{y}}[n] = \begin{cases} \hat{\mathbf{x}}[n] & \text{if } n \geq k, \\ 0 & \text{if } n < k, \end{cases} \tag{16}$$

with the time mapping $t = (n-k)\Delta$*, i.e. $n = \lfloor t/\Delta \rfloor + k$. As stated, indices $n < k$ lie in the filter transient and are excluded.*

Restricting to $n \geq k$ ensures that $\hat{\mathbf{y}}[n]$ reflects the steady-state filter response aligned to the physical time $(n-k)\Delta$.

Definition 5 (FAR-STL Syntax)**.** *A FAR-STL formula φ is defined over the valid aligned trace $\hat{\mathbf{y}}$ by the grammar*

$$\varphi ::= \top \mid \mu \mid \neg\varphi \mid \varphi_1 \wedge \varphi_2 \mid \varphi_1 \,\mathcal{U}_{[n_1,n_2]}\, \varphi_2 \tag{17}$$

where μ is an atomic predicate $f(\hat{\mathbf{y}}[n]) > \epsilon$,

$$\epsilon = L\Delta^* + \delta_p + \sigma \tag{18}$$

is the spatial safety guard*. While Δ^* nominally represents the optimal sampling period ($\Delta = 1/F_s$), if controller jitter ($\delta_j \geq 0$) is present, it is integrated into the temporal uncertainty term by setting $\Delta^* = \Delta + \delta_j$. The $n_1, n_2 \in \mathbb{Z}_{\geq 0}$ are discrete interval bounds. Derived operators $\Box_{[n_1,n_2]}$ and $\Diamond_{[n_1,n_2]}$ are defined as in STL.*

Remark 1 (Spatial Safety Guard)**.** *The spatial safety guard $\epsilon = L\Delta^* + \delta_p + \sigma$ has three components: $L\Delta^*$ bounds the combined temporal discretisation and jitter error (Lipschitz constant L times the total temporal uncertainty), δ_p bounds the filter passband distortion, and σ bounds the residual noise standard deviation. For a fixed hardware configuration, all three are constants; over the synthesis library, they vary with (M, F_s).*

Definition 6 (Temporal Horizon of a FAR-STL Formula)**.** *The* temporal horizon $H_\varphi \in \mathbb{Z}_{\geq 0} \cup \{\infty\}$ *is the maximum discrete look-ahead required to evaluate φ at any base index. For discrete interval bounds $n_1, n_2 \in \mathbb{Z}_{\geq 0}$:*

$$H_\varphi = \begin{cases} 0 & \text{if } \varphi = \mu \\ H_\psi & \text{if } \varphi = \neg\psi \\ \max(H_{\psi_1}, H_{\psi_2}) & \text{if } \varphi = \psi_1 \wedge \psi_2 \\ n_2 + H_\psi & \text{if } \varphi = \Box_{[n_1,n_2]}\psi \\ n_2 + H_\psi & \text{if } \varphi = \Diamond_{[n_1,n_2]}\psi \\ n_2 + \max(H_{\psi_1}, H_{\psi_2}) & \text{if } \varphi = \psi_1 \,\mathcal{U}_{[n_1,n_2]}\, \psi_2 \end{cases} \tag{19}$$

If any sub-formula has $H = \infty$, then $H_\varphi = \infty$. The continuous-time interval $[t_1, t_2]$ of an STL formula is discretised as $n_1 = \lfloor t_1/\Delta \rfloor$, $n_2 = \lfloor t_2/\Delta \rfloor$.

Definition 7 (Valid Evaluation Domain)**.** *The* valid evaluation domain *for a formula φ over a trace of length N is*

$$\mathcal{D}_\varphi = \begin{cases} \{n \in \mathbb{Z} \mid k \leq n \leq N - H_\varphi\} & \text{if } H_\varphi < \infty \\ \{n \in \mathbb{Z} \mid n \geq k\} & \text{if } H_\varphi = \infty \end{cases} \tag{20}$$

We require $N > k + H_\varphi$ for $\mathcal{D}_\varphi$ to be non-empty. The lower bound k excludes transient samples; the upper bound $N - H_\varphi$ ensures that future samples required by φ exist.

Definition 8 (FAR-STL Boolean Semantics)**.** *The satisfaction relation $(\hat{\mathbf{y}}, n) \models \varphi$ for $n \in \mathcal{D}_\varphi$ is defined recursively. Let m denote the* bound dummy variable *swept by temporal operators (distinct from the base index n):*

$$(\hat{\mathbf{y}}, n) \models \mu \iff f(\hat{\mathbf{y}}[n]) > \epsilon \tag{21}$$

$$(\hat{\mathbf{y}}, n) \models \neg\varphi \iff (\hat{\mathbf{y}}, n) \not\models \varphi \tag{22}$$

$$(\hat{\mathbf{y}}, n) \models \varphi_1 \wedge \varphi_2 \iff (\hat{\mathbf{y}}, n) \models \varphi_1 \text{ and } (\hat{\mathbf{y}}, n) \models \varphi_2 \tag{23}$$

$$\begin{aligned} (\hat{\mathbf{y}}, n) \models \varphi_1 \,\mathcal{U}_{[n_1,n_2]}\, \varphi_2 \iff \\ \exists\, m \in [n+n_1,\, n+n_2] \cap \mathcal{D}_\varphi \text{ s.t. } (\hat{\mathbf{y}}, m) \models \varphi_2 \\ \wedge\ \forall\, m' \in [n, m),\ (\hat{\mathbf{y}}, m') \models \varphi_1 \end{aligned} \tag{24}$$

For $n \notin \mathcal{D}_\varphi$, $(\hat{\mathbf{y}}, n)$ does not satisfy any formula.

Note that the temporal operator binds m (the look-ahead index), while n remains fixed as the evaluation base. This is the standard scoping convention for temporal logics and was a source of confusion in an earlier version of this work.

Definition 9 (FAR-STL Quantitative Semantics)**.** *The robustness $\rho(\varphi, \hat{\mathbf{y}}, n)$ follows the same recursive structure as STL (Definition 3), with the continuous signal $\mathbf{x}(t)$ replaced by the valid aligned trace $\hat{\mathbf{y}}[n]$ and the formula ϕ replaced by the FAR-STL formula φ. For $n \notin \mathcal{D}_\varphi$, $\rho = -\infty$ (conservative default).*

Definition 10 (Improved FAR-STL Robustness)**.** *Given hardware configuration (h, F_s, M) with computed SNR_{gain} and required signal bandwidth F_{signal}, the* trust factor *is*

$$\tau = \exp\left(-\lambda_1 \frac{SNR_{target} - SNR_{in}}{SNR_{gain}} - \lambda_2 \frac{F_{signal}}{F_s}\right) \in (0, 1], \tag{25}$$

where $\lambda_1, \lambda_2 > 0$ are weighting coefficients. The FAR-robustness *is*

$$\tilde{\rho}(\varphi, \hat{\mathbf{y}}, n) = \rho(\varphi, \hat{\mathbf{y}}, n) \cdot \tau. \tag{26}$$

τ is penalised when the SNR margin is small (low SNR_{gain} relative to the required gain) or when the sampling rate is close to the Nyquist limit.

To illustrate how the trust factor τ and the resulting FAR-robustness $\tilde{\rho}$ prevent the standard monitor from issuing dangerously optimistic verdicts, we present a comparative analysis in the context of an emergency braking scenario.

The Essential Role of the Trust Factor (τ)

Before detailing the formal instantiation, we highlight why the Trust Factor τ is the cornerstone of FAR-STL's reliability. As illustrated in Figure 5, relying solely on spatial guards (ϵ) is insufficient when the hardware pipeline introduces significant latency.

- **Mitigating the Perception Gap:** In Panel (b), the FIR filter's group delay k creates a window where the ego vehicle is unaware that the lead vehicle has already breached the safety distance.
- **Correcting False Positives:** Standard STL robustness ρ (Panel c) remains optimistic ($\rho > 0$) during this gap because it is "blind" to the filter's delay. It issues a dangerous false-safe verdict.
- **Active Soundness:** By scaling robustness by the trust factor ($\tilde{\rho} = \rho \cdot \tau$), FAR-STL actively penalizes configurations with high latency. In Panel (c), $\tilde{\rho}$ drops below the ϵ threshold, correctly flagging the violation and ensuring physical soundness.

Example 2 (Applying FAR-STL to Example 1)**.** *We illustrate the step-by-step instantiation of the longitudinal safety policies from Example 1 into hardware-aware FAR-STL form.*

STL Policy 1:

$$\phi_1 = \square(v_1 \geq 0 \wedge v_2 \geq 0)$$

This is a universal safety requirement at all times.

STL Policy 2 (gap recovery):

$$\phi_{gap} = \square\Big((g < d_{min}) \rightarrow \Diamond_{[0.5,\,1.5]}(g \geq d_{min})\Big)$$

We now instantiate these policies into FAR-STL for a concrete hardware/software stack, making all uncertain parameters explicit.

Assume:

- *Sampling frequency: F_s (Hz), so $\Delta = 1/F_s$*
- *FIR filter order: M (even), coefficients $h[0], ..., h[M]$, group delay k (known from filter)*
- *Processing delay T_{calc} (possibly known)*
- *Measurement noise std. dev. σ (unknown/assumed)*
- *Positioning error δ_p (unknown/assumed)*
- *Control jitter δ_j (unknown/assumed, default 0 if not present)*
- *Lipschitz constant L (to be specified)*
- *The trace length is N*

Step 1: Discretise time intervals.

- *For untimed $\square$ in ϕ_1, the temporal horizon is the available data:* $[0, N]$.
- *For* $[0.5,\, 1.5]$ *in* $\Diamond_{[0.5,\,1.5]}$ *within* ϕ_{gap}*, discretise:*

$$n_{start} = \lfloor 0.5 \cdot F_s \rfloor, \quad n_{end} = \lfloor 1.5 \cdot F_s \rfloor.$$

Step 2: Compute group delay and valid aligned domain.

- *Group delay: k (given by FIR filter, e.g., $k = \lfloor M/2 \rfloor$ for a Type I symmetric filter)*
- *Temporal horizon for each property:*

$$H_{\varphi_1} = \infty$$
$$H_{\varphi_{gap}} = n_{win} + n_{end}$$

- *Valid aligned domain for each: $\mathcal{D}_{\varphi_1}$ and $\mathcal{D}_{\varphi_{gap}}$ are as obtained in Definition 7.*

Step 3: Compute spatial guard ϵ (see* (29)*).

$$\epsilon = L\Delta^* + \delta_p + \sigma$$

where $\Delta^* = \Delta + \delta_j$.

Step 4: Form the FAR-STL formulas. *Let $\hat{v}_i[n]$ denote filtered velocity traces and $\hat{g}[n]$ the filtered, valid aligned* gap *trace (ego–lead spacing), produced from noisy ranging or odometry after the same FIR chain as in Definition 4. Both φ_1 and φ_{gap} retain the original STL thresholds; the ϵ margin is enforced by the FAR-STL semantics rather than hard-coded per policy.*

1. *Velocity guard (FAR-STL):*

$$\varphi_1 = \square\big([\hat{v}_1[m] \geq 0] \wedge [\hat{v}_2[m] \geq 0]\big)$$

evaluated at $n \in \mathcal{D}_{\varphi_1}$, where $m \in [n, N]$.

2. *Gap recovery on the scalar trace $\hat{g}$ (same structure as ϕ_{gap}, Step 1):*

$$\varphi_{gap} = \square_{[0,\,n_{win}]}\Big([\hat{g}[m] < d_{min}] \rightarrow \Diamond_{[n_{start},\,n_{end}]}\,[\hat{g}[m'] \geq d_{min}]\Big) \tag{27}$$

evaluated at $n \in \mathcal{D}_{\varphi_{gap}}$, with d_{min} the fixed threshold from Example 1. Each predicate $\hat{g}[m] < d_{min}$ and $\hat{g}[m'] \geq d_{min}$ is the canonical FAR-STL guard, so the same ϵ margin that appears for the velocity predicates is enforced here too—there is no policy-specific asymmetry in how ϵ is applied. Here m is bound by the outer $\square_{[0,n_{win}]}$ and m' by the inner $\Diamond$ relative to each such m, as in the quantitative STL semantics (Definition 9).

The values of ϵ, k, n_{win}, n_{start}, n_{end}, and domains can be computed concretely once all hardware and noise parameters are committed; until then, the formulas above are the hardware-aware monitors used in the implementation.

Definition 11 (FAR-STL Monitor)**.** *A FAR-STL monitor for policy φ and configuration $\theta = (h, F_s)$ is a function $\mathcal{M}^{\theta}_{\varphi} : \hat{\mathbf{y}} \times \mathbb{Z}_{\geq k} \rightarrow \{$safe, unsafe, currently safe, currently unsafe$\}$ defined as follows for $n \geq k$. Let prefix$(n) \equiv (H_{\varphi} = \infty) \vee (H_{\varphi} < \infty \wedge n + H_{\varphi} > N)$.*

$$\mathcal{M}^{\theta}_{\varphi}(\hat{\mathbf{y}}, n) = \begin{cases} \textit{safe} & \neg\textit{prefix}(n) \wedge \tilde{\rho}(\varphi, \hat{\mathbf{y}}, n) > \epsilon \\ \textit{unsafe} & \neg\textit{prefix}(n) \wedge \tilde{\rho}(\varphi, \hat{\mathbf{y}}, n) \leq \epsilon \\ \textit{currently safe} & \textit{prefix}(n) \wedge \tilde{\rho}(\varphi, \hat{\mathbf{y}}, n) > \epsilon \\ \textit{currently unsafe} & \textit{prefix}(n) \wedge \tilde{\rho}(\varphi, \hat{\mathbf{y}}, n) \leq \epsilon \end{cases} \tag{28}$$

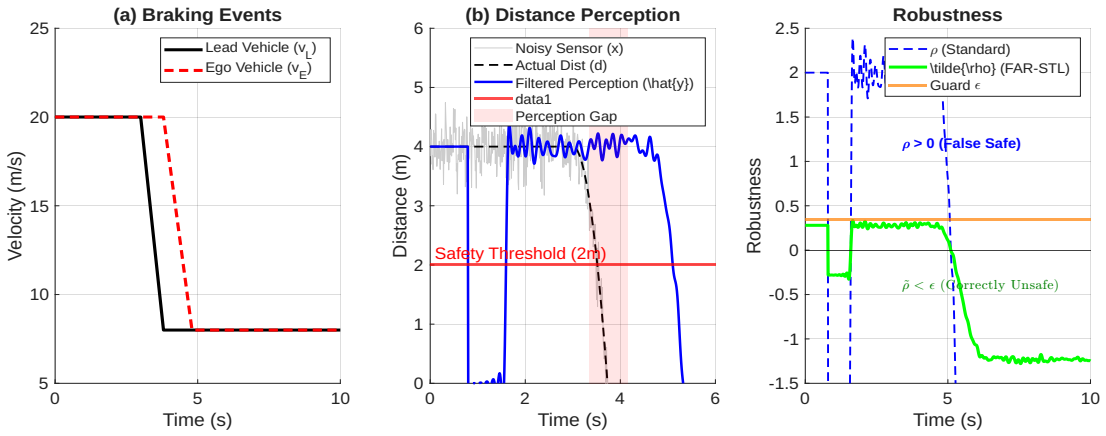

Fig. 5: Comparing standard STL robustness (ρ) with FAR-STL robustness ($\tilde{\rho}$) for an emergency braking scenario. Panel (a): Velocity profiles showing the deceleration lag. Panel (b): The *perception gap* where true distance is unsafe but filtered data suggests safety. Panel (c): Standard robustness $\rho > 0$ falsely indicates safety, while $\tilde{\rho} < \epsilon$ correctly detects the unsafe condition by incorporating the trust factor τ.

The currently safe/unsafe *verdicts apply when the available prefix is too short to resolve the full formula horizon—either because $H_\varphi = \infty$ or because H_φ is finite but large relative to the trace length ($n + H_\varphi > N$). Definitive* safe/unsafe *verdicts require a complete horizon window and hold only when $\neg$prefix(n).*

A. Bisimilarity and STL–FAR-STL Correspondence

Theorem 1 ((Δ^*, ϵ)-Bisimilarity [38], [22])**.** *Let $\Delta^* = \Delta + \delta_j$ be the total temporal uncertainty (δ_j is controller jitter), and let $I_n = [(n-k)\Delta^*, n\Delta^*]$. A valid aligned trace $\hat{\mathbf{y}}$ is* (Δ^*, ϵ)-bisimilar *to $\mathbf{x}$ if, for every $n \in \mathcal{D}_\varphi$,*

$$\sup_{t \in I_n} \|\mathbf{x}(t) - \hat{\mathbf{y}}[n]\|_\infty \leq \epsilon, \qquad \epsilon = L\Delta^* + \delta_p + \sigma. \tag{29}$$

The following lemma establishes the formal relationship between a continuous-time STL formula ϕ and its FAR-STL instantiation φ over a filtered, discrete trace. It shows that FAR-STL is a *sound and complete hardware-aware instantiation* of STL semantics over the valid aligned trace.

Lemma 1 (STL–FAR-STL Correspondence)**.** *Let ϕ be an STL formula and φ the corresponding FAR-STL formula derived by discretising the temporal bounds with $\Delta = 1/F_s$ and incorporating the spatial guard $\epsilon = L\Delta^* + \delta_p + \sigma$. Let $\hat{\mathbf{y}}$ be a valid aligned trace that is (Δ^*, ϵ)-bisimilar to the continuous signal $\mathbf{x}$ (Definition 1). Then:*

1) ***(FAR-STL $\Rightarrow$ STL)*** *$\rho(\varphi, \hat{\mathbf{y}}, n) > \epsilon \implies (\mathbf{x}, t) \models \phi$ for all $t \in I_n$.*
2) ***(STL $\Rightarrow$ FAR-STL)*** *$\rho(\phi, \mathbf{x}, t) > 2\epsilon$ for all $t \in I_n \implies \rho(\varphi, \hat{\mathbf{y}}, n) > \epsilon$.*

Proof. Both directions follow from the 1-Lipschitz property of the STL robustness function [8] and the bisimilarity bound (29).

(1, forward direction.) Assume $\rho(\varphi, \hat{\mathbf{y}}, n) > \epsilon$. For any $t \in I_n$:

$$|\rho(\phi, \mathbf{x}, t) - \rho(\varphi, \hat{\mathbf{y}}, n)| \leq \sup_{t \in I_n} \|\mathbf{x}(t) - \hat{\mathbf{y}}[n]\|_\infty \leq \epsilon.$$

Therefore $\rho(\phi, \mathbf{x}, t) \geq \rho(\varphi, \hat{\mathbf{y}}, n) - \epsilon > 0$, which by Definition 2 implies $(\mathbf{x}, t) \models \phi$.

(2, backward direction.) Assume $\rho(\phi, \mathbf{x}, t) > 2\epsilon$ for all $t \in I_n$. By the same 1-Lipschitz bound:

$$\rho(\varphi, \hat{\mathbf{y}}, n) \geq \rho(\phi, \mathbf{x}, t) - \sup_{t \in I_n} \|\mathbf{x}(t) - \hat{\mathbf{y}}[n]\|_\infty \geq 2\epsilon - \epsilon = \epsilon.$$

Hence $\rho(\varphi, \hat{\mathbf{y}}, n) > \epsilon$. □

Remark. The 2ϵ margin in the backward direction is a standard consequence of using a fixed spatial guard: the continuous specification must be satisfied with *double* the guard width in order for the discrete monitor to reliably detect it. This is the price of the hardware-awareness in FAR-STL; it is precisely the guard ϵ that prevents the monitor from declaring "safe" when the signal is within ϵ of a predicate boundary. Lemma 1 shows that FAR-STL does not add expressiveness but adds a *verified, hardware-grounded connection* between continuous STL and the discrete filtered monitor—something not provided by standard STL or passive uncertainty-aware variants.

Example 3 (Monitoring with improved FAR-STL Robustness $\tilde{\rho}$)**.** *Recall the longitudinal safety policies for the ACC system. In the FAR-STL framework,* monitoring uses the improved FAR-robustness $\tilde{\rho}(\varphi, \hat{\mathbf{y}}, n)$ *from Definition 10:*

$$\tilde{\rho}(\varphi, \hat{\mathbf{y}}, n) = \rho(\varphi, \hat{\mathbf{y}}, n) \cdot \tau,$$

where $\rho(\varphi, \hat{\mathbf{y}}, n)$ is the (base) FAR-STL robustness over the valid aligned trace $\hat{\mathbf{y}}$ (Definition 9) and $\tau \in (0, 1]$ is the trust factor from (25) *for the hardware configuration and filter. Verdicts follow Definition 11: e.g.* unsafe *when $\neg$prefix(n) and $\tilde{\rho}(\varphi, \hat{\mathbf{y}}, n) \leq \epsilon$;* currently unsafe *when prefix(n) and $\tilde{\rho}(\varphi, \hat{\mathbf{y}}, n) \leq \epsilon$.*

1) ***Underlying STL policy*** *ϕ_1:* $\square(v_1 \geq 0 \wedge v_2 \geq 0)$*;* ***monitored FAR-STL formula*** *φ_1 as in Example 2.*
   Monitor action: *For each $n \in \mathcal{D}_{\varphi_1}$, compute*

$$\tilde{\rho}(\varphi_1, \hat{\mathbf{y}}, n) = \rho(\varphi_1, \hat{\mathbf{y}}, n) \cdot \tau,$$

where $\rho(\varphi_1, \hat{\mathbf{y}}, n)$ uses the ϵ-guarded predicates on $\hat{v}_1, \hat{v}_2$ from that instantiation. If $\tilde{\rho}(\varphi_1, \hat{\mathbf{y}}, n) \leq \epsilon$, the monitor

flags a violation (Definition 11), indicating (under the bisimilarity assumptions of Lemma 1) that the continuous policy ϕ_1 may be violated—e.g. one vehicle may be reversing.

2) ***Underlying STL policy ϕ_{gap}:*** *timed gap recovery as in Example 1;* ***monitored FAR-STL formula φ_{gap}*** *as in* (27)*, where the ϵ guard is subsumed by the canonical predicate semantics.*
Monitor action: *For each $n \in \mathcal{D}_{\varphi_{gap}}$, form a single-channel trace $\hat{\mathbf{y}} = \hat{g}$ (or embed $\hat{g}$ in the first component of $\hat{\mathbf{y}}$) and compute*

$$\tilde{\rho}(\varphi_{gap}, \hat{g}, n) = \rho(\varphi_{gap}, \hat{g}, n) \cdot \tau.$$

If $\tilde{\rho}(\varphi_{gap}, \hat{g}, n) \leq \epsilon$, the monitor reports a violation at step n. In both cases, the monitor runs online *on the filtered, aligned trace. Alerts use $\tilde{\rho}$ compared against the spatial guard ϵ (Definition 11).*

V. Joint Synthesis and Soundness

A. Why Joint Synthesis Is Necessary

In a decoupled design, an engineer (i) selects a filter based on SNR requirements, (ii) manually computes ϵ and τ for that filter, and (iii) writes an STL formula with those constants hard-coded. Steps (i)–(ii) are *outside* the formal framework: there is no guarantee that the chosen ϵ is correct for the chosen filter, and changing the filter requires repeating all steps manually. More critically, no formal bridge connects the discrete monitor verdict to the original continuous-time specification.

In FAR-STL, $\epsilon(M, F_s, \sigma, \delta_p)$ and $\tau(M, F_s, \text{SNR})$ are *symbolic functions of design variables*. This enables the synthesis loop below to automatically evaluate the correct formula instantiation for *every* candidate in a filter library, compare them by verified robustness, and return the optimal design with a formal soundness certificate.

B. Synthesis Pipeline

The pipeline (Fig. 6) is formulated as the joint optimisation

$$\begin{aligned} \max_{F_s,\, h} \quad & J(F_s, h) = \min_{\mathbf{x} \in \mathcal{X}} \inf_{n \in \mathcal{D}_\varphi} \rho(\varphi, \hat{\mathbf{y}}_{\mathbf{x}}, n) \cdot \tau(F_s, \text{SNR}_{\text{gain}}) \\ \text{s.t.} \quad & k \leq \nu \cdot F_s \quad \text{(temporal feasibility)} \\ & \text{SNR}_{\text{gain}} \geq \text{SNR}_{\text{target}} - \text{SNR}_{\text{in}} \quad \text{(noise rejection)} \\ & \tilde{\rho}(\varphi, \hat{\mathbf{y}}_{\mathbf{x}}, n) > \epsilon,\ \forall \mathbf{x} \in \mathcal{X}, \forall n \in \mathcal{D}_\varphi \quad \text{(soundness guard)} \\ & F_s \in [F_{\min}, F_{\max}] \quad \text{(hardware limits)} \end{aligned} \tag{30}$$

where $\mathcal{X}$ is a set of representative operational traces and ν is the maximum tolerable filter latency (reaction budget).

1) Phase 1: DSP Synthesis Layer: Algorithm 1 performs a grid search over (F_s, M) pairs. For each candidate, it (i) computes the group delay k and prunes configurations where $k > \nu \cdot F_s$ (*Early Temporal Alignment Pruning*), (ii) synthesises a Type I FIR filter using a windowed-sinc method, (iii) computes the SNR gain and prunes if the target is not met, and (iv) stores the qualifying entry $(h, F_s, \text{SNR}_{\text{gain}}, M, \delta_p, \tau, k)$ in the library $\mathcal{H}$.

Algorithm 1 DSP Synthesis Layer (Phase 1)

Input: SNR_{in}, $\text{SNR}_{\text{target}}$, $F_{s,\min}$, $F_{s,\max}$, F_{signal}, $M_{\max}$, ω_c, λ_1, λ_2, ν
Output: Library $\mathcal{H} = \{(h, F_s, \text{SNR}_{\text{gain}}, M, \delta_p, \tau, k)_i\}$

$\mathcal{H} \leftarrow \emptyset$
for $F_s = F_{s,\min}, \ldots, F_{s,\max}$ **do**
  for $M = 2, 4, \ldots, M_{\max}$ **do** ▷ Type I: even order
    $k \leftarrow \lfloor M/2 \rfloor + \lceil T_{\text{calc}} \cdot F_s \rceil$
    if $k > \nu \cdot F_s$ **then continue** ▷ Temporal pruning
    end if
    $h \leftarrow \text{WindowedFIR}(M, \omega_c, F_s)$
    $\text{SNR}_{\text{gain}} \leftarrow 10 \log_{10}(1/\sum_m h[m]^2)$
    if $\text{SNR}_{\text{gain}} < \text{SNR}_{\text{target}} - \text{SNR}_{\text{in}}$ **then continue**
    end if
    $\delta_p \leftarrow \max_{\omega \in \Omega_{\text{pass}}} |1 - |H(e^{j\omega})||$
    $\tau = \exp\left(-\lambda_1 \frac{\text{SNR}_{\text{target}} - \text{SNR}_{\text{in}}}{\text{SNR}_{\text{gain}}} - \lambda_2 \frac{F_{\text{signal}}}{F_s}\right)$
    $\mathcal{H} \leftarrow \mathcal{H} \cup \{(h, F_s, \text{SNR}_{\text{gain}}, M, \delta_p, \tau, k)\}$
  end for
end for
return $\mathcal{H}$

Algorithm 2 Logical Verification Layer (Phase 2)

Input: Library $\mathcal{H}$, property φ, trace set $\mathcal{X}$
Output: Optimal configuration $\theta^* = (h, F_s, M)^*$

$\tilde{\rho}^* \leftarrow -\infty, \quad \theta^* \leftarrow \text{null}$
for each $\theta_j \in \mathcal{H}$ **do**
  $\tilde{\rho}_{\text{temp}} \leftarrow +\infty$
  for each $\mathbf{x} \in \mathcal{X}$ **do**
    $\hat{\mathbf{x}} \leftarrow \mathbf{x} * h_j$ ▷ FIR convolution
    $\hat{\mathbf{y}}_{\mathbf{x}}[n] \leftarrow \hat{\mathbf{x}}[n]$, for $n \geq k_j$ ▷ Valid aligned trace
    for each $n \in \mathcal{D}_\varphi$ **do**
      $\rho \leftarrow \text{EvalRobustness}(\varphi, \hat{\mathbf{y}}_{\mathbf{x}}, n)$
      $\tilde{\rho}_{\text{temp}} \leftarrow \min(\tilde{\rho}_{\text{temp}},\ \rho \cdot \tau_j)$
    end for
  end for
  if $\tilde{\rho}_{\text{temp}} > \tilde{\rho}^*$ **and** $\tilde{\rho}_{\text{temp}} > \epsilon_j$ **then**
    $\tilde{\rho}^* \leftarrow \tilde{\rho}_{\text{temp}}, \quad \theta^* \leftarrow \theta_j$
  end if
end for
return θ^*

2) Phase 2: Logical Verification Layer: Algorithm 2 evaluates each library entry against the property φ on a set of representative traces $\mathcal{X}$. For each configuration θ_j and each trace $\mathbf{x} \in \mathcal{X}$, it filters $\mathbf{x}$, extracts the valid aligned trace $\hat{\mathbf{y}}_{\mathbf{x}}$ for $n \geq k_j$, evaluates FAR-robustness over $\mathcal{D}_\varphi$, and forms the worst-case trust-weighted robustness $\min_{\mathbf{x} \in \mathcal{X}} \inf_{n \in \mathcal{D}_\varphi} \rho(\varphi, \hat{\mathbf{y}}_{\mathbf{x}}, n) \cdot \tau_j$. Configuration θ_j is retained as the new optimum only if this worst-case value improves the incumbent and satisfies the soundness guard on every trace in $\mathcal{X}$.

C. Soundness, Termination, and Completeness

We establish three properties of the FAR-STL framework via the (Δ^*, ϵ)-bisimilarity relation defined in Theorem 1 (Section IV).

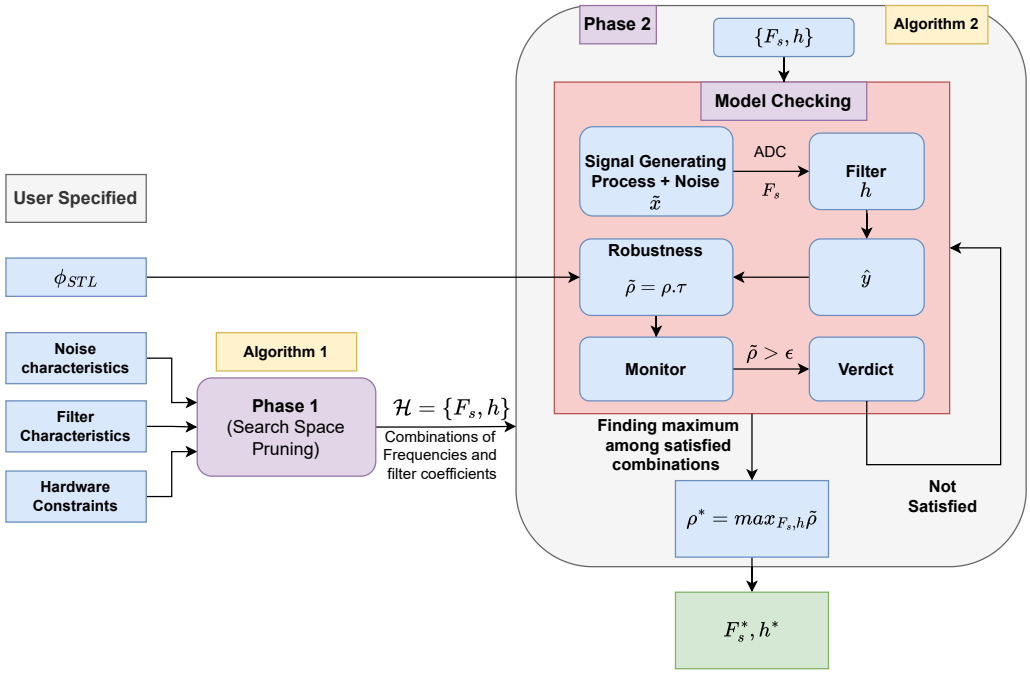

Fig. 6: FAR-STL two-phase synthesis pipeline. Phase 1 (Algorithm 1) builds a library $\mathcal{H}$ of hardware-feasible filter configurations. Phase 2 (Algorithm 2) evaluates each library entry and returns the optimal design θ^*.

The bound (29) is satisfied by construction when the filter parameters are chosen to satisfy the Phase 1 constraints: the $L\Delta^*$ term is controlled by F_s, the δ_p term by the filter design, and the σ term by the SNR gain.

Theorem 2 (Soundness, Termination, and Completeness)**.** *Let $\hat{\mathbf{y}}$ be (Δ^*, ϵ)-bisimilar to $\mathbf{x}$. Let ϕ be an STL formula and φ the corresponding FAR-STL formula.*

1) ***Soundness.*** *$\rho(\varphi, \hat{\mathbf{y}}, n) > \epsilon \implies \mathbf{x}(t) \models \phi$ for all $t \in I_n$.*
2) ***Termination.*** *Algorithm 2 terminates in $O(|\mathcal{H}| \cdot |\mathcal{D}_\varphi|)$ steps.*
3) ***Completeness.*** *$\rho(\phi, \mathbf{x}, t) > 2\epsilon$ for all $t \in I_n \implies \rho(\varphi, \hat{\mathbf{y}}, n) > \epsilon$.*

Proof. 1) **(Soundness.)** Assume $\rho(\varphi, \hat{\mathbf{y}}, n) > \epsilon$. By the 1-Lipschitz property of the STL robustness function [8], and by the bisimilarity bound (29), for all $t \in I_n$:

$$|\rho(\phi, \mathbf{x}, t) - \rho(\varphi, \hat{\mathbf{y}}, n)| \leq \sup_{t \in I_n} \|\mathbf{x}(t) - \hat{\mathbf{y}}[n]\|_\infty \leq \epsilon.$$

Therefore $\rho(\phi, \mathbf{x}, t) \geq \rho(\varphi, \hat{\mathbf{y}}, n) - \epsilon > 0$, which implies $\mathbf{x}(t) \models \phi$.

2) **(Termination.)** The library $|\mathcal{H}|$ is finite because both the grid of F_s values and the range of filter orders $[2, M_{\max}]$ are finite (bounded by hardware limits $F_{\max}$ and $M_{\max}$). The domain $|\mathcal{D}_\varphi|$ is finite because N (signal length) and H_φ are finite for bounded-horizon formulas. Each iteration of the inner loop in Algorithm 2 performs a fixed-cost robustness evaluation. Therefore the outer loop terminates after $|\mathcal{H}|$ iterations and the inner loop terminates after $|\mathcal{D}_\varphi|$ iterations.

3) **(Completeness.)** Assume $\rho(\phi, \mathbf{x}, t) > 2\epsilon$ for all $t \in I_n$. By the 1-Lipschitz property and the bisimilarity bound:

$$\begin{aligned}\rho(\varphi, \hat{\mathbf{y}}, n) &\geq \rho(\phi, \mathbf{x}, t) - \sup_{t \in I_n} \|\mathbf{x}(t) - \hat{\mathbf{y}}[n]\|_\infty \\ &\geq 2\epsilon - \epsilon = \epsilon, \text{ so } \rho(\varphi, \hat{\mathbf{y}}, n) > \epsilon.\end{aligned}$$

□

Remark on the soundness guard. The monitor only issues a verdict when $\tilde{\rho} = \rho \cdot \tau > \epsilon$ (line 12 of Algorithm 2). Because $\tau \leq 1$, this condition is *stricter* than $\rho > \epsilon$: it additionally penalises configurations with low SNR margin or insufficient sampling density. The guard, therefore, provides conservative verdicts when hardware fidelity is poor.

Example 4 (Synthesis trace)**.** *Consider the longitudinal ACC contracts from Example 1. Policy ϕ_1 requires both vehicles to move forward; policy ϕ_{gap} bounds a 100 ms window with a timed gap-recovery obligation ($\square_{[0,0.1]}$ and $\Diamond_{[0.5,1.5]}$ on the inter-vehicle gap). Example 2 rewrites these as FAR-STL formulas φ_1 over $\hat{v}_i$ and φ_{gap} over the scalar aligned gap $\hat{g}$, with spatial guard ϵ and valid domains $\mathcal{D}_{\varphi_1}$, $\mathcal{D}_{\varphi_{gap}}$ after group delay k. Example 3 is the operational reading: at each valid n the monitor forms $\tilde{\rho}(\varphi, \hat{\mathbf{y}}, n) = \rho(\varphi, \hat{\mathbf{y}}, n) \cdot \tau$ and compares $\tilde{\rho}$ to ϵ.*

We illustrate the joint objective (30) *for $\varphi = \varphi_1$ first. Intuitively, J rewards configurations whose* worst *trace and time step (minimum over $\mathbf{x} \in \mathcal{X}$ and $n \in \mathcal{D}_{\varphi_1}$) still leaves a large trust-weighted margin above the hardware guard; Phase 1 discards candidates that cannot meet SNR or latency before any robustness is computed. The same library $\mathcal{H}$ is then reused for φ_{gap}: only $\mathcal{D}_{\varphi_{gap}}$, the monitored channel (scalar $\hat{g}$), and*

the horizon parameters ($n_{win}, n_{start}, n_{end}$) change—the filter (h, F_s, M) and thus k, SNR_{gain}, and τ are shared.

Let $SNR_{in} = 75\,dB$, $SNR_{target} = 82\,dB$, and $\nu = 150\,ms$. The Phase 1 SNR rule requires $SNR_{gain} \geq SNR_{target} - SNR_{in} = 7\,dB$ so that, under the white-noise model (15)*, the filtered chain is trusted to suppress noise enough for the chosen ϵ. The temporal rule is $k \leq \nu F_s$: at $F_s = 100\,Hz$ the budget is $0.15 \times 100 = 15$ ticks—every candidate below must align the FIR output within that many samples (cf. line 4 of Algorithm 1).*

Phase 1 (library construction). *Config A ($F_s = 100\,Hz$, $M = 4$): $k = \lfloor M/2 \rfloor + \lceil T_{calc} F_s \rceil = 2 + 1 = 3$ ticks (30 ms), which is temporally feasible ($3 \leq 15$). However $SNR_{gain} = 6.2\,dB$ falls short of the 7 dB requirement, so A is* rejected before robustness evaluation*—a short filter is fast, but does not justify the noise model assumed in Example 2. Config B ($M = 12$): $k = 6 + 1 = 7$ ticks (70 ms), $SNR_{gain} = 11.2\,dB \geq 7\,dB$, hence $\tau_B = 0.92$ from the Phase 1 trust update (Algorithm 1, lines 95–97; same spirit as Definition 10).* Accepted *into $\mathcal{H}$. Config C ($M = 24$): $k = 12 + 2 = 14$ ticks (140 ms), still under the 15-tick cap; $SNR_{gain} = 14.8\,dB$ and $\tau_C = 0.98$.* Accepted*. Thus $\mathcal{H}$ contains B and C but not A.*

Phase 2 (logical verification on φ_1). *Given a recorded ACC manoeuvre, Algorithm 2 convolves the raw channels with each h_j, drops the transient so that $\hat{\mathbf{y}}[n]$ is valid for $n \geq k_j$, and evaluates $\rho(\varphi_1, \hat{\mathbf{y}}, n)$ on $\mathcal{D}_{\varphi_1}$ (Definition 9). For each θ_j, the inner loop tracks the bottleneck step*

$$\tilde{\rho}^{(j)}_{temp} = \inf_{n \in \mathcal{D}_{\varphi_1}} \rho(\varphi_1, \hat{\mathbf{y}}, n) \cdot \tau_j,$$

matching the objective in (30)*. A candidate is eligible only if that same minimum still exceeds ϵ_j (line 12): equivalently, at every n, the trust-weighted robustness stays above the spatial guard, as in Example 3. Suppose that, for the trace at hand, both B and C pass this test. Holding the discrete signal fixed, a larger τ scales every $\rho \cdot \tau$ upward, so $\tilde{\rho}^{(C)}_{temp} > \tilde{\rho}^{(B)}_{temp}$ whenever the underlying $\rho(\varphi_1, \hat{\mathbf{y}}, n)$ are comparable—here $\tau_C = 0.98 > \tau_B = 0.92$ dominates the comparison. Phase 2 therefore returns $\theta^* =$ Config C.*

Takeaway. *Config C uses 93% of the reaction budget (140 ms of 150 ms) but yields the highest certified trust-weighted robustness on φ_1. The same Phase 2 loop applies to φ_{gap} on the aligned gap trace. Config B is also sound for this trace yet leaves more timing slack; the synthesis loop exposes the explicit trade-off between filter latency (alignment delay k) and hardware confidence τ.*

Remark 2 (Offline synthesis, online deployment)**.** *Algorithms 1–2 are intended to run* offline*: Phase 1 builds $\mathcal{H}$ from filter-design constraints, and Phase 2 scores candidates on a set of representative traces $\mathcal{X}$—nominal manoeuvres together with corner cases from domain expertise or falsification tools such as Breach [9] and S-TaLiRo [10]—to select $\theta^* = (F_s^*, h^*, M^*, \ldots)$ that is robust across operational scenarios. The cardinality of $\mathcal{X}$ is benchmark-dependent: the objective is to elicit low-robustness corner cases rather than to fix a universal trace count. The outcome is a fixed sampling rate F_s^*, a concrete FIR h^*, and the associated alignment delay k^*, trust τ^*, and guard ϵ^* tailored for the chosen φ. At runtime, the deployed stack samples the physical signal at F_s^*, applies h^* (with the same transient handling as in Phase 2), and feeds the aligned trace to the same quantitative STL monitor, comparing $\rho(\varphi, \hat{\mathbf{y}}, n) \cdot \tau^*$ to ϵ^* at each step. Thus, co-design amortises analysis cost offline while online monitoring reduces to streaming convolution plus standard robustness evaluation.*

VI. Results and Discussion

Beyond the ACC running example developed in Sections III–V, we evaluate FAR-STL on two additional case studies—an aerial vehicle and a cardiac electrogram monitor—before returning to the full ACC experiment. Results are reported in that same order: drone, heart, then ACC, with the latter presented last because it is the most involved.

A. Case Studies

All FAR-STL monitors share the causal outer window $\square_{[0,\, n_{\text{win}}]}$, where $n_{\text{win}} = \lfloor 0.1 F_s \rfloor$ samples denotes the 100 ms online monitoring buffer (cf. Example 2).

1) Drone: FADS kinematic plant: The drone case study uses the three-axis kinematic unmanned-aircraft model from the FADS framework [39]. Translational motion follows a point-mass double integrator $\dot{\mathbf{p}}(t) = \mathbf{v}(t)$, $\dot{\mathbf{v}}(t) = \mathbf{a}(t)$, with per-axis velocity and acceleration bounds as in FADS Definition 8.1. Here $\mathbf{v}(t) = [v_x(t), v_y(t), v_z(t)]$ and $\mathbf{a}(t) = [a_x(t), a_y(t), a_z(t)]$ are the velocity and acceleration vectors, and $z(t)$ is the altitude. A reference acceleration $\mathbf{a}_{\text{ref}}(t)$ is generated by minimum-jerk quintic splines [40] between waypoints, decoupled per axis as in [39]. The simulator produces altitude $z(t)$, groundspeed $v_g(t) = \sqrt{v_x^2 + v_y^2 + v_z^2}$, and acceleration channels at $F_s^{\text{raw}} = 10\,\text{kHz}$.

We monitor the FAA regulatory caps from FADS Section 4:

$$\varphi_{\text{alt}} = \square_{[0,\, n_{\text{win}}]}(\hat{z} \leq 121\ \text{m}), \qquad \varphi_{\text{spd}} = \square_{[0,\, n_{\text{win}}]}(\hat{v}_g \leq 44\ \text{m/s}).$$

Both are safety envelopes over the aligned altitude and speed traces within the causal outer window; Phase 2 selects the filter and sampling configuration that maximises the certified margin for each scalar channel.

2) Heart: EGM conductance model: The heart case study uses a 33-node electrogram (EGM) conductance simulator [41] that propagates action potentials through a reduced cardiac network and records atrial and ventricular EGM proxy signals. The C implementation is stepped at $\Delta t = 1\,\text{ms}$ ($F_s^{\text{raw}} = 1\,\text{kHz}$) under a healthy configuration; additive Gaussian noise is injected on the sensor channels before filtering. We monitor two timing policies from the healthy-heart specification, with $V_{a,\text{th}} = V_{v,\text{th}} = 80\,\text{mV}$. The inner $\Diamond$ bounds are discretised at the candidate monitor rate as $n^-_{\text{AV}} = \lfloor 0.18 F_s \rfloor$, $n^+_{\text{AV}} = \lfloor 0.24 F_s \rfloor$ (AV delay, 180–240 ms) and $n^-_{\text{VV}} = \lfloor 0.6 F_s \rfloor$, $n^+_{\text{VV}} = \lfloor 1.0 F_s \rfloor$ (V–V interval, 600–1000 ms):

$$\varphi_{\text{AV}} = \square_{[0,\, n_{\text{win}}]}\Big((\hat{A}_{\text{EGM}} \geq V_{a,\text{th}}) \ \rightarrow \Diamond_{[n^-_{\text{AV}},\, n^+_{\text{AV}}]} (\hat{V}_{\text{EGM}} \geq V_{v,\text{th}})\Big), \tag{31}$$

$$\varphi_{\text{VV}} = \square_{[0,\, n_{\text{win}}]}\Big((\hat{V}_{\text{EGM}} \geq V_{v,\text{th}}) \ \rightarrow \Diamond_{[n^-_{\text{VV}},\, n^+_{\text{VV}}]} (\hat{V}_{\text{EGM}} > V_{v,\text{th}})\Big). \tag{32}$$

φ_{AV} checks that a ventricular EGM crossing follows each atrial activation within the AV interval; φ_{VV} checks that a physiologically valid interval separates successive ventricular activations.

3) ACC: Intelligent Driver Model: For completeness, the ACC case study uses the Intelligent Driver Model (IDM) [42] to produce gap and velocity channels at $F_s^{\text{raw}} = 10\,\text{kHz}$. Policies follow Example 1, instantiated in FAR-STL form as in (27); the spatial guard ϵ is subsumed by the canonical predicate semantics (Definition 5) and so is not written into the thresholds:

$$\varphi_d = \square_{[0,\,n_{\text{win}}]}\Big((\hat{g} < d_{min}) \rightarrow \Diamond_{[\lfloor 0.5F_s \rfloor,\,\lfloor 1.5F_s \rfloor]}(\hat{g} \geq d_{min})\Big),$$

with threshold $d_{min} = 2\,\text{m}$; and

$$\varphi_v = \square_{[0,\,n_{\text{win}}]}(\hat{v_1} \geq 0 \wedge \hat{v_2} \geq 0).$$

This study couples a nonlinear car-following plant to two policies of contrasting temporal structure (nested $\square_{[0,\,n_{\text{win}}]}{-}{\rightarrow}{-}\Diamond$ gap recovery vs outer-window forward-motion guard) and is the most involved of the three evaluations.

B. Experimental Setup

All three case studies share the same FAR-STL synthesis protocol. Each run evaluates policies on 10 s discrete traces at candidate monitor rates F_s (in Hz). Robustness to additive sensor noise is assessed at $\sigma \in \{0.01, 0.1, 0.3\}$ with zero computation latency ($T_{\text{calc}} = 0$); Phase 2 is executed independently at each σ, and all reported optima are drawn from the corresponding per-σ optimisation output files.

a) Phase 1 (filter library).: Algorithm 1 sweeps $F_s \in [200,\,1000]\,\text{Hz}$ in 10 Hz steps with Kaiser-windowed FIR filters, trust parameters $\lambda_1 = \lambda_2 = 0.5$, and SNR targets $\{5,\,10,\,15\}\,\text{dB}$. Only configurations with achieved $\text{SNR}_{\text{gain}} > \text{SNR}_{\text{target}}$ are retained. Cutoff frequency f_c is set from signal bandwidth: $f_c = 10\,\text{Hz}$ (drone), $f_c = 40\,\text{Hz}$ (heart), and $f_c = 30\,\text{Hz}$ (ACC). This yields 201, 100, and 121 Phase 1 candidates for drone, heart, and ACC, respectively.

b) Phase 2 (policy-aware grid search).: Algorithm 2 evaluates every library candidate on aligned, noisy (σ) filtered traces for each policy, maximising the minimum trust-weighted robustness over traces and time indices subject to the ϵ guard[1]. We do not use a trace set $\mathcal{X}$ which considers corner cases from domain expertise or falsification tools and instead evaluate on a single nominal trace only for each configuration from the library to ensure brevity. Nevertheless, the experiments can be easily extended to consider a set of corner cases. *Margin* denotes $\tilde{\rho}_{\min} - \epsilon$, the slack between the minimum trust-weighted robustness along the trace and the soundness guard; $\tilde{\rho}_{\min} > \epsilon$ (equivalently, positive margin) is the condition checked for a `SAFE` verdict in the FAR-STL monitor.

Wall-clock timings for the Python prototype are logged separately. Phase 1 is also benchmarked on a denser F_s grid (70 Hz–10 kHz). The STL nesting-depth micro-benchmark in Section VI-D is run only on ACC policies: the evaluation cost is driven by formula structure, not by the physical case study.

[1]The 100 ms signal buffer is used continuously to ensure causal monitoring.

C. Simulation Results

Table II lists the Phase 2 optimum at each $\sigma \in \{0.01, 0.1, 0.3\}$ for all six policies.

1) Drone results: For FADS altitude and groundspeed policies, Phase 2 finds `SAFE` verdicts across the full library of 201 candidates at every $\sigma \in \{0.01, 0.1, 0.3\}$. The optimiser consistently selects the highest-rate candidate ($F_s^* = 1000\,\text{Hz}$, $M^* = 100$) for both φ_{alt} and φ_{spd}, with margins shrinking only slightly as σ grows (Table II). Altitude monitoring achieves the largest margins in our evaluation ($\tilde{\rho}_{\min} \approx 70$), reflecting a wide separation between the simulated trajectory and the 121 m cap; groundspeed margins are smaller (≈ 25.8) but remain comfortably positive. Table III illustrates the monotonic improvement in $\tilde{\rho}_{\min}$ as F_s increases within the library ($\sigma = 0.1$).

2) Heart results: Heart policies expose stronger hardware–logic coupling than the drone caps. For φ_{VV}, 90 of 100 library candidates are `SAFE` at each σ; Phase 2 selects $F_s^* = 970\,\text{Hz}$, $M^* = 42$ with large margins ($\tilde{\rho}_{\min} \approx 35.9$ at $\sigma = 0.1$), and the same (F_s^*, M^*) is retained across all three noise levels. For φ_{AV}, only three configurations pass the ϵ guard at *every* σ—the tight 180–240 ms $\Diamond$ window inside $\square_{[0,\,n_{\text{win}}]}$ leaves little slack once group delay and discretisation are accounted for. The selected optimum ($F_s^* = 250\,\text{Hz}$, $M^* = 8$), therefore *does not* coincide with the highest sampling rate in the library: lower rate and shorter filter order reduce alignment delay while preserving enough temporal resolution to witness ventricular response within the $\Diamond$ window. Margins[2] on φ_{AV} are an order of magnitude smaller than on φ_{VV} (≈ 2.2 vs. ≈ 35.9 at $\sigma = 0.1$), highlighting how sub-second timing constraints can dominate filter selection.

3) ACC results: Phase 2 scores all 121 configurations against φ_d and φ_v; every candidate returned a `SAFE` verdict at all three σ levels. The same optimal configuration ($F_s^* = 1000\,\text{Hz}$, $M^* = 34$) is selected across σ for both policies (Table II), demonstrating stable selection under $30\times$ noise variation. Trust-weighted robustness $\tilde{\rho}_{\min}$ is much larger for gap monitoring (≈ 23) than for velocity (≈ 7.7) because the monitored predicates and robustness scales differ between signals, even though the optimiser selects the same (F_s^*, M^*). Table IV shows representative gap configurations for $\sigma = 0.1$.

4) Key observations: The three case studies reveal several consistent patterns. First, trust-weighted margins vary substantially across signals and policies even under comparable noise sweeps—from ≈ 2.2 (heart φ_{AV}) to ≈ 70 (drone altitude)—due to the different scales of the monitored predicates and robustness functions. Second, when the policy horizon is coarse relative to plant dynamics (drone, ACC), Phase 2 favours the highest F_s in the sweep; when millisecond-level $\Diamond$ windows matter (heart φ_{AV}), a mid-rate configuration wins and most of the library fails certification. Third, the soundness guard $\epsilon = L\Delta + \delta_p + \sigma$ grows with σ, so margins shrink slightly as noise increases even when (F_s^*, M^*) is unchanged across all three σ levels. Lastly, $\tilde{\rho} = \rho \cdot \tau$ compresses raw

[2]Values in table are rounded to 2 decimal places after calculation of corresponding variables. Hence, post rounding, the margin may not be exactly the difference between $\tilde{\rho}_{\min}$ and ϵ used in the table.

TABLE II: Optimal FAR-STL configurations selected by Phase 2 across three case studies (all σ levels). Altitude, groundspeed, gap, and velocity policies returned SAFE for every library candidate at every σ; heart φ_{AV} admits only three SAFE configurations at each σ (see text).

Case study	Policy	σ	F_s^* [Hz]	M^*	ϵ	τ	$\rho_{\min}$	$\tilde{\rho}_{\min}$	Margin
Drone	φ_{alt}	0.01	1000	100	0.523	0.682	103.00	70.26	69.74
Drone	φ_{alt}	0.10	1000	100	0.613	0.682	102.98	70.24	69.63
Drone	φ_{alt}	0.30	1000	100	0.813	0.682	102.93	70.21	69.40
Drone	φ_{spd}	0.01	1000	100	0.523	0.682	37.79	25.78	25.25
Drone	φ_{spd}	0.10	1000	100	0.613	0.682	37.76	25.76	25.15
Drone	φ_{spd}	0.30	1000	100	0.813	0.682	37.71	25.72	24.91
Heart	φ_{AV}	0.01	250	8	0.374	0.502	4.49	2.26	1.88
Heart	φ_{AV}	0.10	250	8	0.464	0.502	4.38	2.20	1.74
Heart	φ_{AV}	0.30	250	8	0.664	0.502	4.14	2.08	1.42
Heart	φ_{VV}	0.01	970	42	0.270	0.482	74.49	35.92	35.65
Heart	φ_{VV}	0.10	970	42	0.360	0.482	74.46	35.91	35.55
Heart	φ_{VV}	0.30	970	42	0.560	0.482	74.38	35.87	35.31
ACC	φ_d (gap)	0.01	1000	34	0.535	0.652	35.72	23.29	22.76
ACC	φ_d (gap)	0.10	1000	34	0.625	0.652	35.73	23.30	22.67
ACC	φ_d (gap)	0.30	1000	34	0.825	0.652	35.69	23.27	22.45
ACC	φ_v (velocity)	0.01	1000	34	0.535	0.652	11.86	7.73	7.20
ACC	φ_v (velocity)	0.10	1000	34	0.625	0.652	11.86	7.74	7.11
ACC	φ_v (velocity)	0.30	1000	34	0.825	0.652	11.88	7.75	6.92

TABLE III: Sample filter configurations for drone altitude ($\sigma = 0.1$).

F_s [Hz]	M	δ_p	τ	ϵ	$\tilde{\rho}_{\min}$	Verdict
200	20	5.27e-1	0.631	0.632	64.96	SAFE
500	50	5.12e-1	0.665	0.614	68.50	SAFE
800	80	5.29e-1	0.677	0.630	69.75	SAFE
1000	100	5.12e-1	0.682	0.613	70.24	SAFE

TABLE IV: Sample filter configurations for ACC gap signal ($\sigma = 0.1$).

F_s [Hz]	M	δ_p	τ	ϵ	$\tilde{\rho}_{\min}$	Verdict
200	8	4.12e-1	0.525	0.517	18.76	SAFE
240	8	5.16e-1	0.566	0.620	20.18	SAFE
500	18	5.05e-1	0.620	0.607	22.15	SAFE
1000	34	5.24e-1	0.652	0.625	23.30	SAFE

robustness and yields a hardware-aware margin used for SAFE verdicts; the compression factor interacts with predicate scale, explaining why altitude and gap traces can show very different $\tilde{\rho}_{\min}$ despite similar optimal F_s^*.

Overall, the evaluation shows how FAR-STL provides a statically guaranteed safety margin that depends jointly on the monitored signal, the STL encoding, and the chosen filter and sampling parameters.

D. Timing benchmarks

Filter-bank construction (Phase 1) scales roughly linearly with the number of F_s candidates in the sweep (Kaiser window, case-specific f_c, three SNR targets): about 6 s for 10 candidates, about 35 s for 100, and about 330–360 s for 1000.

Phase 2 grid search over the full library completes in a few minutes per policy on our prototype hardware for the 10 s traces reported here; runtime grows with the simulation interval (trace length N) and with library size $|\mathcal{H}|$, since each candidate requires a full robustness pass over $\mathcal{D}_\varphi$. As Phase 2 is an offline synthesis step, this cost does not affect online deployment, where the deployed stack samples the physical signal at F_s^*, applies h^* (with the same transient handling as in Phase 2), and feeds the aligned trace to the same quantitative STL monitor, comparing $\rho(\varphi, \hat{y}, n) \cdot \tau^*$ to ϵ^* at each step.

The nesting-depth benchmark is reported only for the ACC case study, because robustness evaluation time is determined by the STL policy structure rather than the physical plant. On a single channel with fixed horizon $H{=}10$, length $N{=}1000$, $F_s{=}100$ Hz, and $\tau{=}0.566$, we repeat trust-weighted robustness evaluation across nesting depths $1, 2, 4, 8, 16, 32$. Wall-clock grows exponentially from about 14 ms at depth 1 to about 2.3 s at depth 8, about 9.4 s at depth 16, and about 68.3 s at depth 32 (mean over 5 repeats). Shallow policies (as in the drone and ACC experiments) make robustness evaluation inexpensive compared with deep nesting, which is computationally costly given the recursive nature of quantitative semantics.

E. Qualitative Comparison with Existing Online Monitoring Tool: RTAMT

We qualitatively compare our approach with RTAMT, a recent reference for *online* quantitative STL monitoring in CPS and robotics [43], and it closely matches our work in terms of online monitoring, highlighting the differences in our approach. A direct quantitative benchmark is not meaningful: RTAMT is designed for monitor generation with fixed sampling, while FAR-STL jointly selects filters and sampling for hardware-aware robustness. There is no common metric or fair basis for end-to-end comparison.

Nevertheless, relative to RTAMT, FAR-STL evaluates *full* STL without such syntactic reductions (to pSTL and bfSTL), and Definition 11 adds *currently safe* and *currently unsafe* alongside *safe* and *unsafe* when the available prefix cannot resolve the full formula horizon—for unbounded properties

or for finite but large H_φ—so conservative partial verdicts are available on finite prefixes; FAR-STL additionally couples monitoring to filter and sampling parameters (F_s, $\mathcal{H}$, delay, noise) via ϵ and $\tilde{\rho}$, which RTAMT does not treat and assumes the user provides these.

Furthermore, a stack that already uses RTAMT (or similar) for deployment-time robustness evaluation can still gain from FAR-STL upstream, because choosing F_s, $\mathcal{H}$, and guard ϵ fixes what discrete trace actually reaches the monitor, and is a way to supply better-conditioned signals and explicit margins before any online engine runs.

VII. Conclusion

We have presented **FAR-STL**, a framework that makes digital filter parameters, first-class formal objects in STL monitoring. Rather than introducing a strictly more expressive logic, FAR-STL provides a *verified, hardware-aware instantiation* of STL semantics: for a given filter library, it automatically derives the correct spatial safety guard ϵ and trust factor τ for each candidate, evaluates the formula on the valid aligned trace, and returns the configuration that maximises formal robustness with a soundness certificate. The (Δ^*, ϵ)-bisimilarity theorem guarantees that a positive FAR-STL verdict on the discrete filtered trace implies satisfaction of the original continuous-time specification.

The key insight is that the decoupled design of filter and monitor is formally unsound: ϵ and τ depend on the filter parameters, so hard-coding them in an STL formula without iterating over the design space misses the optimal configuration and provides no formal bridge to the continuous ground truth. FAR-STL makes this iteration explicit and automated.

Limitations. The current framework considers FIR filters only; extending to IIR filters requires a different group delay characterisation. The Phase 1 grid search is exhaustive over a discretised hardware space; gradient-based optimisation could reduce synthesis time.

Future work. We plan to extend FAR-STL to (i) multi-rate sensor fusion, where signals from sensors at different sampling frequencies must be aligned before monitoring; and (ii) joint synthesis of the controller and the perception pipeline, where the filter configuration adapts to real-time changes in signal fidelity.

References

[1] IEC, "IEC 61508-1:2010 — webstore.iec.ch," https://webstore.iec.ch/en/publication/5515, 2010, [Accessed 08-08-2025].

[2] ISO, "Iso 26262:2018 road vehicles – functional safety," 2018, aSIL-D sensor chain requirements (Table 7).

[3] A. Donzé, "On signal temporal logic," in *Runtime Verification*, A. Legay and S. Bensalem, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 382–383.

[4] E. Bartocci, J. Deshmukh, A. Donzé, G. Fainekos, O. Maler, D. Ničković, and S. Sankaranarayanan, "Specification-Based Monitoring of Cyber-Physical Systems: A Survey on Theory, Tools and Applications," in *Lectures on Runtime Verification*, E. Bartocci and Y. Falcone, Eds. Cham: Springer International Publishing, 2018, vol. 10457, pp. 135–175, series Title: Lecture Notes in Computer Science. [Online]. Available: http://link.springer.com/10.1007/978-3-319-75632-5_5

[5] ETSI, "Etsi en 301 091: Short range devices; transport and traffic telematics," 2022, radar equipment operating in 76-77 GHz (LRR).

[6] FCC, "Fcc part 95 subpart m: 76-81 ghz band vehicle radar systems," 2023, eIRP, SNR, interference requirements.

[7] O. Maler and D. Nickovic, "Monitoring temporal properties of continuous signals," in *Formal Techniques, Modelling and Analysis of Timed and Fault-Tolerant Systems*, ser. LNCS, vol. 3253. Berlin, Germany: Springer, 2004, pp. 152–166.

[8] A. Donzé and O. Maler, "Robust satisfaction of temporal logic over real-valued signals," in *Formal Modeling and Analysis of Timed Systems*, ser. LNCS, vol. 6246. Berlin, Germany: Springer, 2010, pp. 92–106.

[9] A. Donzé, "Breach, a toolbox for verification and parameter synthesis of hybrid systems," in *Proceedings of the 22nd International Conference on Computer Aided Verification (CAV)*, ser. LNCS, vol. 6174. Springer, 2010, pp. 167–170.

[10] Y. Annpureddy, C. Liu, G. E. Fainekos, and S. Sankaranarayanan, "S-taliro: A tool for temporal logic falsification for hybrid systems," in *Proceedings of the 17th International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS)*, ser. LNCS, vol. 6605. Springer, 2011, pp. 254–257.

[11] A. Donzé and O. Maler, "Efficient robust monitoring of signal temporal logic properties," in *Tools and Algorithms for the Construction and Analysis of Systems*. Springer, 2010, pp. 158–173.

[12] J. V. Deshmukh, A. Donzé, S. Ghosh, X. Jin, O. Maler, and S. Sankaranarayanan, "Robust online monitoring of signal temporal logic," *Formal Methods in System Design*, vol. 51, no. 1, pp. 5–30, 2017.

[13] K. Mamouras, A. Chattopadhyay, and Z. Wang, "A compositional framework for algebraic quantitative online monitoring over continuous-time signals," *Formal Methods in System Design*, 2023.

[14] B. Melani, E. Bartocci, and M. Chiari, "A tree-shaped tableau for checking the satisfiability of signal temporal logic with bounded temporal operators," *ACM Transactions on Embedded Computing Systems (TECS)*, vol. 24, no. 5s, 2025.

[15] A. Aerts, B. Tong Minh, M. R. Mousavi, and M. A. Reniers, "Temporal logic falsification of cyber-physical systems: An input-signal-space optimization approach," in *2018 IEEE International Conference on Software Testing, Verification and Validation Workshops (ICSTW)*, 2018, pp. 214–223.

[16] G. Ernst, I. Hasuo, Z. Zhang, and S. Sedwards, "Time-staging enhancement of hybrid system falsification," in *Proceedings of the 6th International Workshop on Applied Verification of Continuous and Hybrid Systems (ARCH)*, ser. EPTCS, 2018.

[17] E. Bartocci, L. Bortolussi, M. Loreti, and L. Nenzi, "Monitoring mobile and spatially distributed cyber-physical systems," in *Proceedings of the 15th ACM-IEEE International Conference on Formal Methods and Models for System Design (MEMOCODE)*, 2017.

[18] Z. Ma, J. Qin, and J. V. Deshmukh, "Sastl: Spatial aggregation signal temporal logic for spatio-temporal properties," in *2020 ACM/IEEE 11th International Conference on Cyber-Physical Systems (ICCPS)*. IEEE, 2020, pp. 1–10.

[19] A. Momtaz, H. Abbas, and B. Bonakdarpour, "Monitoring signal temporal logic in distributed cyber-physical systems," in *Proceedings of the ACM/IEEE 14th International Conference on Cyber-Physical Systems (ICCPS)*, 2023, pp. 154–165.

[20] Y. Zhao, E. Zhu, B. Hoxha, G. Fainekos, J. V. Deshmukh, and L. Lindemann, "Distributionally robust predictive runtime verification under spatio-temporal logic specifications," *ACM Transactions on Cyber-Physical Systems*, vol. 9, no. 4, 2025.

[21] E. Visconti, E. Bartocci, M. Loreti, and L. Nenzi, "Online monitoring of spatio-temporal properties for imprecise signals," in *Proceedings of the 19th ACM-IEEE International Conference on Formal Methods and Models for System Design (MEMOCODE)*, 2021, pp. 78–88.

[22] T. Ferrere, N. Malaquias, D. Nickovic, and J. Preneel, "Interface-aware signal temporal logic," in *proceedings of the 22nd ACM International Conference on Hybrid Systems: Computation and Control*, 2019, pp. 57–66.

[23] B. Zhong, C. Jordan, and J. Provost, "Extending signal temporal logic with quantitative semantics by intervals for robust monitoring of cyber-physical systems," *ACM Transactions on Cyber-Physical Systems*, vol. 5, no. 2, 2021.

[24] B. Finkbeiner, F. Fuß, F. Klein, and J. Krčal, "A truly robust signal temporal logic: Monitoring safety properties of continuous signals," *Lecture Notes in Computer Science*, vol. 13435, pp. 315–334, 2022.

[25] C. Bellanger, P.-L. Garoche, M. Martel, and C. Picard, "Formally proved specification of non-nested STL formulas as synchronous observers," *Science of Computer Programming*, vol. 245, p. 103315, Oct. 2025. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167642325000541

[26] A. Antoniou, *Digital Signal Processing: Signals, Systems, and Filters*. McGraw-Hill Education, 2016.

[27] Z. E. Bhatti, R. Sinha, and P. S. Roop, "Observer based verification of iec 61499 function blocks," in *2011 9th IEEE International Conference on Industrial Informatics*, 2011, pp. 609–614.

[28] P. S. Roop, H. Pearce, and K. Monadjem, "Synchronous neural networks for cyber-physical systems," in *2018 16th ACM/IEEE International Conference on Formal Methods and Models for System Design (MEMOCODE)*, 2018, pp. 1–10.

[29] J. Vargas, S. Alsweiss, O. Toker, R. Razdan, and J. Santos, "An overview of autonomous vehicles sensors and their vulnerability to weather conditions," *Sensors*, vol. 21, no. 16, p. 5397, 2021.

[30] K. Watanabe, E. Kang, C.-W. Lin, and S. Shiraishi, "Runtime monitoring for safety of intelligent vehicles," in *Proceedings of the 55th Annual Design Automation Conference (DAC)*, 2018.

[31] S. Jakšić, E. Bartocci, R. Grosu, T. Nguyen, and D. Ničković, "Quantitative monitoring of stl with edit distance," *Formal Methods in System Design*, vol. 53, pp. 83–112, 2018.

[32] M. Seo and F. Kurdahi, "Efficient tracing methodology using automata processor," *ACM Transactions on Embedded Computing Systems (TECS)*, vol. 18, no. 5s, 2019.

[33] M. Ma, J. Stankovic, E. Bartocci, and L. Feng, "Predictive monitoring with logic-calibrated uncertainty for cyber-physical systems," *ACM Transactions on Embedded Computing Systems (TECS)*, vol. 20, no. 5s, 2021.

[34] L. Lindemann, X. Qin, J. V. Deshmukh, and G. J. Pappas, "Conformal prediction for stl runtime verification," in *Proceedings of the ACM/IEEE 14th International Conference on Cyber-Physical Systems (ICCPS)*, 2023, pp. 142–153.

[35] V. Lin, R. Kaur, Y. Yang, S. Dutta, Y. Kantaros, A. Roy, S. Jha, O. Sokolsky, and I. Lee, "Safety monitoring for learning-enabled cyber-physical systems in out-of-distribution scenarios," in *Proceedings of the ACM/IEEE 16th International Conference on Cyber-Physical Systems (ICCPS)*, 2025.

[36] N. Aréchiga, "Specifying safety of autonomous vehicles in signal temporal logic," in *2019 IEEE Intelligent Vehicles Symposium (IV)*, 2019, pp. 58–63.

[37] A. Papoulis and S. U. Pillai, *Probability, Random Variables, and Stochastic Processes*, 4th ed. McGraw-Hill, 2002.

[38] G. Yan, L. Jiao, Y. Li, S. Wang, and N. Zhan, "Approximate bisimulation and discretization of hybrid CSP," *CoRR*, vol. abs/1609.00091, 2016. [Online]. Available: http://arxiv.org/abs/1609.00091

[39] Y. Pant, M. Li, R. Quaye, A. Rodionova, H. Abbas, M. Ryerson, and R. Mangharam, "Fads: A framework for autonomous drone safety using temporal logic-based trajectory planning," *Transportation Research Part C: Emerging Technologies*, vol. 130, p. 103275, 09 2021.

[40] M. W. Mueller, M. Hehn, and R. D'Andrea, "A computationally efficient motion primitive for quadrocopter trajectory generation," *IEEE Transactions on Robotics*, vol. 31, no. 6, pp. 1294–1310, 2015.

[41] E. Yip, S. Andalam, P. S. Roop, A. Malik, M. L. Trew, W. Ai, and N. Patel, "Towards the emulation of the cardiac conduction system for pacemaker validation," *ACM Trans. Cyber-Phys. Syst.*, vol. 2, no. 4, Jul. 2018. [Online]. Available: https://doi.org/10.1145/3134845

[42] M. Treiber, A. Hennecke, and D. Helbing, "Congested traffic states in empirical observations and microscopic simulations," *Physical Review E*, vol. 62, no. 2, pp. 1805–1824, 2000.

[43] T. Yamaguchi, B. Hoxha, and D. Ničković, "RTAMT – runtime robustness monitors with application to cps and robotics," *International Journal on Software Tools for Technology Transfer*, vol. 26, no. 1, pp. 79–99, 2024.

Supratim Gupta is an Associate Professor (On Leave) in the Department of Electrical Engineering at the National Institute of Technology (NIT) Rourkela, India. Currently, he is also serving as a Postdoctoral Fellow in the Department of Electrical, Computer, and Software Engineering (ECSE) at the University of Auckland, New Zealand. He is an active member of the PRETZEL research group, focusing on formal methods for real-time distributed computing systems on FPGA-based platforms and collaborative research involving logically synchronous networks. He received his Ph.D. in Electrical Engineering from the Indian Institute of Technology (IIT) Kharagpur and his B.E.E. and M.E.E. degrees from Jadavpur University. His research interests encompass digital signal and image processing, embedded computing systems, computer vision, and formal verification. He holds a granted Indian patent and has authored numerous peer-reviewed publications in international journals and conferences.

Sobhan Chatterjee received an MS degree in Advanced Control and Systems Engineering in 2018 from the University of Manchester, U.K. and a Ph.D. in Electrical and Electronic Engineering in 2026 from the University of Auckland, New Zealand. His Ph.D. research examined the decomposition of neural networks to facilitate hardware implementation, improve their adherence to formal safety policies, and enhance explainability. He has also contributed to global AI projects such as the Pandemic Resilience project from the Global Partnership on AI (GPAI) and the India–New Zealand Business Council (INZBC). He currently works as a Postdoctoral Fellow at the University of Auckland, investigating safe AI-based distributed systems.

Naijun Zhan is a Boya Distinguished Professor in the School of Computer Science of Peking University. He received his bachelor's and master's degrees from Nanjing University and his Ph.D. from the Institute of Software, Chinese Academy of Sciences (ISCAS). Prior to joining Peking University, he worked at the Faculty of Mathematics and Informatics, Mannheim University, Germany, as a research fellow, and afterwards worked at ISCAS as an associate professor, a full professor, and a distinguished professor. His research interests cover the formal design of real-time, embedded, and hybrid systems, and program verification. He serves on the editorial boards of *Journal of Automated Reasoning*, *Formal Aspects of Computing*, *Journal of Logical and Algebraic Methods in Programming*, *Journal of Software*, *Journal of Electronics*, and *Journal of Computer Research and Development*, among others. He has published more than 150 papers in international leading journals and conferences and two books, and edited four conference proceedings and seven journal special issues.

Partha Roop received the Ph.D. degree in computer science from the University of New South Wales, Sydney, NSW, Australia, in 2001. He is currently a Professor with the Department of Electrical and Computer Engineering, University of Auckland, New Zealand. His research interests include the design and verification of cyber-physical systems (CPS). In particular, he is developing techniques for the design of CPS applications in automotive, robotics, intelligent transportation, and medical devices. He received the Mercator professorship from the German Research Council (DFG) in 2016 and the Humboldt Fellowship for experienced researchers in 2009. He is an editor of *IEEE Embedded Systems Letters* and the Co-Editor-in-Chief of ACM Books.