

RT-Radar: Maneuver-Aware Real-Time Beam Scheduling for Multi-Target Tracking Tasks

Jinsoo Park^{*†}, Hanwoong Park[†], Youngjin Cho[†], Jinkyu Lee^{‡§}

^{*}Department of Computer Science and Engineering, Sungkyunkwan University, Republic of Korea

[†]Radar R&D Center, Hanwha Systems Co., Ltd., Republic of Korea

[‡]Department of Computer Science and Engineering, Yonsei University, Republic of Korea,
waldo0322@skku.edu, hwpark4415@hanwha.com, guswind8@hanwha.com, jinkyu.lee@yonsei.ac.kr

Abstract—Modern Phased-Array Radars (PARs) integrate surveillance and tracking functions on a single antenna timeline, introducing a fundamental scheduling challenge: optimizing the finite beam-time budget to balance tracking capacity and accuracy. Although the optimal tracking period is dictated by dynamic target maneuverability, prevailing Radar Resource Management (RRM) paradigms decouple these physical dynamics from the real-time beam scheduler, inevitably causing resource waste or track loss. Tightly integrating maneuver-aware adaptation within a hard real-time (HRT) framework has remained elusive due to technical challenges inherent to radar systems. These include: (i) managing complex, path-dependent tracking dynamics, known as hysteresis, to ensure tracking stability while optimizing resource utilization; (ii) formally verifying transient safety during immediate, zero-delay reconfiguration under non-preemptive constraints; and (iii) navigating the non-monotonic, NP-hard optimization space to efficiently balance capacity and accuracy. In this paper, we propose RT-Radar, the first framework to achieve maneuver-aware period adaptation while guaranteeing HRT safety under non-preemptive scheduling. We make three key contributions. First, we develop a novel asymmetric control policy that proactively manages hysteretic tracking dynamics. Second, we establish a rigorous transient schedulability analysis for non-preemptive EDF that formally guarantees the safety of zero-delay reconfiguration. Third, we design an anomaly-aware optimization strategy, guided by bottleneck load analysis and accelerated by an efficient $O(n)$ incremental calculation, to navigate the non-monotonic space. Evaluation on an industrial-grade high-fidelity PAR simulator shows that RT-Radar significantly outperforms established RRM algorithms, achieving up to 27.3% higher tracking success ratio during overload and reducing average tracking error by up to 60.3% during underload.

I. INTRODUCTION

Surveillance and Tracking Radar (STR) systems are the cornerstone of mission-critical applications, such as defense and air traffic control, demanding high-fidelity, real-time awareness of dynamic environments. Modern Phased-Array Radars (PARs) leverage software-defined beamforming to interleave surveillance and multi-target tracking functions on a single antenna timeline. This integration introduces a fundamental real-time scheduling challenge: optimizing the finite beam-time budget to balance the inherent trade-off between (G1) *tracking capacity*, the number of targets tracked, and (G2) *tracking accuracy*, the precision of state estimates. The *tracking period* primarily governs this trade-off; shorter periods

enhance accuracy but increase utilization, potentially reducing capacity. As tracking stability relies on temporally correct updates, these systems mandate hard real-time (HRT) guarantees.

The prevailing paradigm in Radar Resource Management (RRM) is fundamentally misaligned with physical reality: the optimal tracking period is dictated by the target's real-time *maneuverability*, quantified by acceleration. However, current RRM frameworks [1]–[19] rely on static periods or decoupled adaptation. They optimize the beam-time budget based on externally determined deadlines, failing to integrate real-time physical dynamics into the HRT scheduling loop. As targets maneuver dynamically and unpredictably, this decoupling inevitably leads to resource waste through over-sampling or track loss through under-sampling.

To effectively co-optimize G1 and G2, a paradigm shift is necessary: tightly integrating maneuver-aware period adaptation within the real-time beam scheduler. However, achieving such integration has remained elusive due to severe, interconnected technical challenges T1–T3 inherent to STR systems.

- T1. *Hysteretic tracking dynamics*. Accurately mapping the target's real-time maneuverability to the required tracking period is essential for optimizing both G1 and G2. However, this mapping is complicated by the path-dependent tracking dynamics. We identify *hysteresis* in the tracking error, a phenomenon where the error behavior differs significantly between maneuver onset and offset. This asymmetry invalidates simple reactive adaptation based solely on acceleration magnitude. The challenge is to develop an asymmetric control policy that proactively manages this hysteresis to ensure tracking stability while optimizing resource utilization.
- T2. *Safe zero-delay reconfiguration*. To maintain G2 against unpredictable target maneuvers, the beam scheduler must apply adapted periods immediately, with zero delay. This immediacy, however, risks inducing transient overload, threatening G1. The challenge is to formally verify *transient safety*: guaranteeing no deadline misses during the zero-delay reconfiguration. Established real-time adaptation mechanisms, such as mode change protocols (MCPs) [20]–[23], introduce coordination delays incompatible with this immediacy. Furthermore, existing adaptive RRM, including online optimization [24]–[28] and

[§]Jinkyu Lee is the corresponding author.

learning-based policies [29]–[31], lack formal guarantees.

T3. *Non-monotonic optimization space.* The overarching challenge lies in determining the optimal set of tracking periods that maximizes G2 by utilizing available slack during underload, and maximizes G1 while minimizing G2 degradation during overload. This optimization is NP-hard [32] and critically exacerbated by non-preemptive constraints, which induce a *non-monotonic* schedulability space. In this space, reducing utilization may paradoxically increase the actual schedulability load. This anomaly renders utilization-based optimization strategies fundamentally inadequate and unsafe. The challenge is to design a low-overhead, anomaly-aware optimization strategy to navigate this complex space efficiently.

In this paper, we propose RT-Radar, the first real-time beam scheduling framework that dynamically optimizes tracking periods based on real-time target maneuverability while providing rigorous HRT guarantees under non-preemptive scheduling, thereby co-optimizing G1 and G2. RT-Radar is a closed-loop cyber-physical system (CPS) comprising a *Period Adaptation Module* (PAM) and a *Real-Time Beam Scheduler* (RTBS), specifically designed to address T1–T3.

To address T1, PAM employs a novel asymmetric control policy that ensures rapid stabilization during maneuver onset and immediate resource reclamation during maneuver offset. For T2, we develop a rigorous transient schedulability analysis for non-preemptive Earliest Deadline First (NP-EDF). This analysis fills a critical gap in existing real-time theory by formally guaranteeing the safety of zero-delay reconfiguration. To resolve T3, RTBS employs an anomaly-aware optimization strategy guided by bottleneck load analysis. We accelerate this strategy by developing an efficient $O(n)$ incremental load calculation, which enables low-latency decision-making.

We implemented RT-Radar and evaluated it using an industrial-grade high-fidelity PAR simulator that faithfully replicates live-target operational field data. The results demonstrate that RT-Radar significantly outperforms established RRM algorithms, achieving up to 27.3% higher tracking success ratio during overload conditions and reducing the average tracking error by up to 60.3% during underload conditions.

In summary, this paper makes the following contributions:

- We identify critical insights into tracking dynamics, including hysteresis, motivating the necessity of maneuver-aware period adaptation (Sec. II).
- We propose RT-Radar, the first system design to co-optimize G1 and G2 for multi-target tracking (Sec. III).
- We develop a novel asymmetric control policy that proactively manages hysteresis, ensuring rapid stabilization and immediate resource reclamation (Sec. IV).
- We establish a rigorous transient schedulability analysis for NP-EDF that formally guarantees the safety of immediate, zero-delay reconfiguration (Sec. V-B).
- We design an anomaly-aware optimization strategy guided by bottleneck load analysis to navigate the non-monotonic NP-EDF space (Sec. V-C).



Fig. 1: The operational concept of an STR system.

- We demonstrate the effectiveness of RT-Radar through experiments on a high-fidelity PAR simulator (Sec. VI).

II. TARGET SYSTEM AND MOTIVATION

This section details our target system and its unique real-time constraints. We then motivate RT-Radar through a case study identifying critical insights into tracking dynamics.

A. Target System: Surveillance and Tracking Radar

An STR is a mission-critical system responsible for detecting and tracking multiple targets (Fig. 1). It executes two primary functions: *surveillance*, discovering new targets, and *tracking*, maintaining known trajectories. Modern PARs leverage electronic beam steering, implementing these distinct functions as software-defined *beam tasks* interleaved on a single antenna timeline. Tracking tasks require periodic updates for stability, while surveillance tasks utilize the remaining time, leading to contention for the finite beam-time budget.

Following standard practice in RRM [33], we model the tracking of a single target as a dedicated periodic task, τ_i . This periodicity is crucial for the stability of state estimation algorithms, such as Kalman filters. Each task generates a sequence of jobs; the k -th job of τ_i , denoted by $J_{i,k}$, executes as a *dwell*—an atomic, non-preemptive sequence of transmission, reception, and processing operations. We identify three fundamental constraints that distinguish radar resource allocation from conventional real-time scheduling problems:

- C1. *Non-preemptivity.* Dwells execute atomically, inherently inducing non-preemptive blocking. This complicates schedulability analysis even under EDF, rendering simple utilization bounds insufficient and mandating explicit load analysis that accounts for blocking time.
- C2. *Dynamic and unpredictable workload.* The active task set evolves unpredictably as targets arrive/depart. Furthermore, as detailed below, tracking periods must be dynamically adapted to target maneuvers, leading to spontaneous fluctuations in system load.
- C3. *Mission criticality.* The system mandates HRT constraints. Temporal correctness is essential for stable state estimation; a deadline miss can cause unrecoverable error and track loss. During saturation, the system must ensure predictable degradation based on mission priority (MP).

B. Motivation: The Need for Maneuver-aware Adaptation

The fundamental scheduling challenge in STR systems, complicated by C1–C3, is optimizing the finite beam-time budget to balance the inherent trade-off between G1 and

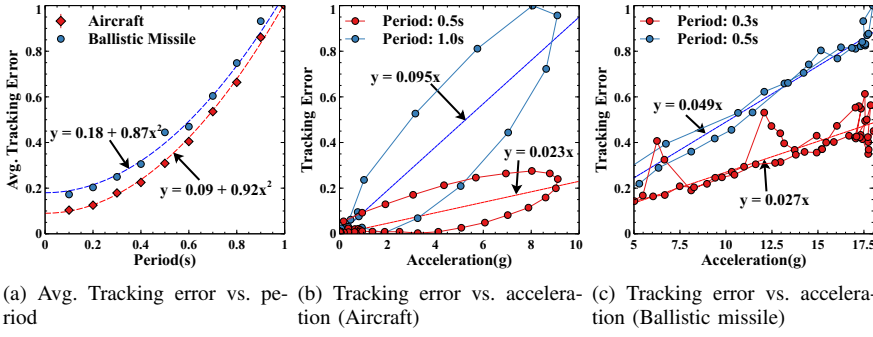


Fig. 2: Measured relationship between tracking period, target acceleration, and tracking error.

G2. This trade-off is primarily governed by the tracking period. For a specific target, τ_i , the optimal tracking period depends critically on its real-time maneuverability, quantified by acceleration. Conceptually, from a task-level perspective, let T_i , a_i , and e_i denote the tracking period, acceleration, and resulting tracking error of τ_i , respectively. Tracking involves predicting the target's future state; T_i defines the prediction horizon, while a_i represents the unmodeled dynamics causing deviation from the prediction, thereby increasing e_i .

Conventional STR systems, however, manage these periods using static assignments or decoupled adaptation schemes. This approach is inherently suboptimal as it fails to integrate real-time dynamics directly into the HRT scheduling loop. To effectively co-optimize G1 and G2 in response to dynamic physical changes, we shift our focus from this coarse-grained task-level perspective to a fine-grained, *job-level* adaptation. We dynamically adjust the tracking period for the subsequent job $J_{i,k+1}$, denoted by $T_{i,k+1}$. This adjustment is based on the target's latest maneuverability, denoted by $a_{i,k}$, which is estimated immediately upon completion of the current job $J_{i,k}$.

Measurement-based case study. To validate our approach and uncover the underlying tracking dynamics, we conducted a case study using an industrial-grade high-fidelity PAR simulator. This simulator integrates the identical software stack deployed in operational STR systems and is validated through Software-in-the-Loop (SITL) execution, replicating live-target testing results using field data. We analyzed two representative tactical threats: an aircraft and a ballistic missile. We evaluated the actual tracking error $e_{i,k}$ measured for each job, defined as the positional error against ground truth, across various fixed periods T_i and levels of dynamic maneuverability $a_{i,k}$. Fig. 2 presents the results as normalized tracking error.

We synthesize the critical insights from this study.

- I1. Tracking error is highly sensitive to both period and acceleration, following the relationship $e \propto aT^2$.

Fig. 2(a) shows that the average tracking error increases convexly (near-quadratically) with the fixed period. Figs. 2(b) and 2(c) demonstrate that the instantaneous job-level tracking error increases monotonically with the real-time acceleration. Crucially, the curves corresponding to longer periods exhibit

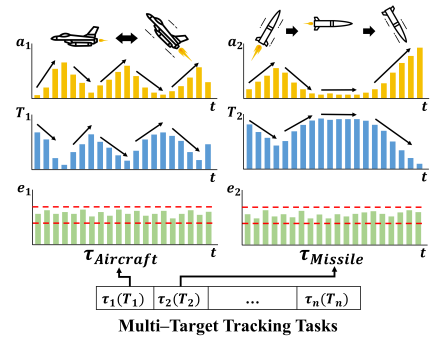


Fig. 3: The core concept of RT-Radar: maneuver-aware period adaptation.

steeper slopes, indicating that tracking error accumulates much faster during maneuvers when updates are infrequent.

- I2. Tracking error exhibits significant asymmetry, known as hysteresis, with respect to acceleration trends.

Figs. 2(b) and 2(c) reveal the critical hysteresis: the job-level tracking error during increasing acceleration (*maneuver onset*; upper path) is substantially higher than during decreasing acceleration (*maneuver offset*; lower path), even at the same acceleration magnitude. This asymmetry arises from the inherent dynamics of tracking filters, to be explained in Sec. IV-B.

Implications. These insights establish a clear mandate for maneuver-aware period adaptation, illustrated in Fig. 3. They fundamentally challenge the conventional decoupled RRM paradigm and dictate the core design requirements for RT-Radar. First, I1 confirms a quantifiable relationship governing tracking error. By monitoring the latest acceleration, we can determine the precise period required to maintain tracking stability for the subsequent job. This insight establishes the foundation for maneuver-aware adaptation. Implementing this foundation necessitates an HRT-compatible *transformation model* capable of analytically mapping estimated acceleration and required accuracy bounds to the appropriate tracking period. Second, I2 reveals complex, hysteretic dynamics in the job-level tracking error. This path-dependency implies that reacting solely to the magnitude of acceleration is insufficient; the adaptation policy must proactively and asymmetrically respond to the *acceleration trend* at a job-level granularity. Managing this path-dependent behavior therefore requires a specialized asymmetric control policy: maneuver onset demands *aggressive stabilization* through rapid period reduction for safety, whereas maneuver offset permits *immediate resource reclamation* through rapid period extension for efficiency.

C. Transformation Model

The transformation model serves as the physics-to-cyber bridge, translating physical dynamics (acceleration) and QoS requirements (accuracy constraints) into the temporal constraints (periods) required by the scheduler. To implement the maneuver-aware adaptation subject to I1–I2 within the HRT constraints of C3, the underlying model must support predictable scheduling decisions. Its primary role is not the

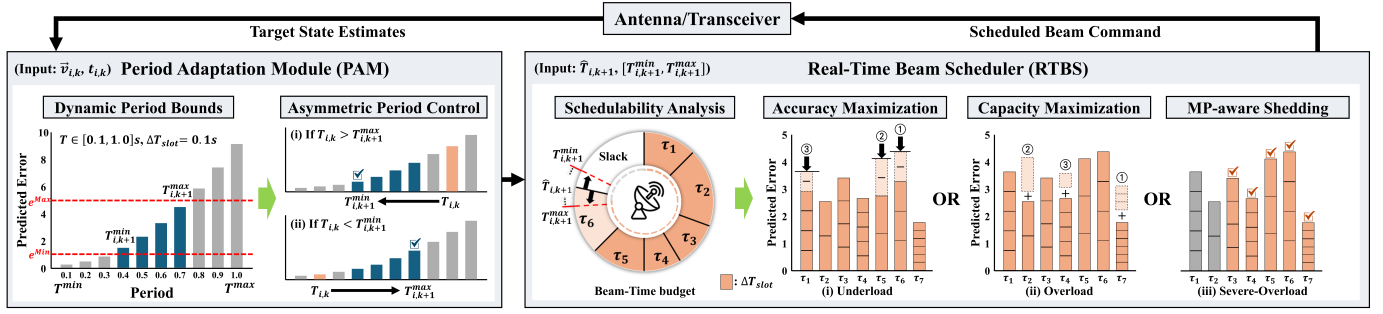


Fig. 4: System architecture and workflow of RT-Radar.

high-fidelity estimation of the actual error, but rather the rapid derivation of a safe period amenable to formal analysis. Therefore, an HRT-compatible transformation model must satisfy three essential properties: (i) computational predictability, requiring low computational complexity and a strictly bounded derivation latency; (ii) determinism, mandating a closed-form, analytical derivation from measurable quantities; and (iii) analyzability, necessitating a physics-based foundation to enable quantitative reasoning about tracking accuracy.

Existing approaches often fail to satisfy these properties simultaneously. Online optimization [24]–[28] or iterative solvers [34] violate (i) and (ii) due to high overhead and variable latency. Empirical or heuristic schemes [35]–[37] fail (iii), precluding formal analysis. Learning-based policies [29]–[31] typically violate all three due to unpredictable overhead, lack of closed-form solutions, and opacity to formal verification.

The adopted model. To satisfy (i)–(iii), we adopt a physics-based tracking error model [38], validated by I1:

$$\hat{e}_i \approx \frac{1}{2} a_i T_i^2. \quad (1)$$

Rooted in standard Newtonian kinematics, this model quantifies the positional deviation of a maneuvering target from a constant-velocity prediction, thereby isolating the maneuver-induced error—the dominant factor in resource allocation. We denote the error as $\hat{e}$ to emphasize that it represents the *predicted error*, distinct from the actual measured error. Crucially, prioritizing this systematic kinematic bias over stochastic noise provides a *deterministic bound* on the error, essential for verifiable HRT scheduling decisions. Furthermore, solving for T_i yields the closed-form period $T_i = \sqrt{2\hat{e}_i/a_i}$. Consequently, this model uniquely satisfies all required properties: (i) $O(1)$ complexity; (ii) closed-form; and (iii) physics-based.

III. RT-RADAR: SYSTEM DESIGN

In this section, we define the system goals and introduce the RT-Radar architecture designed to achieve these goals.

A. System Goals and Overview

RT-Radar is a real-time beam scheduling framework designed to manage the inherent trade-off between tracking capacity and accuracy on a single antenna timeline. We aim to achieve the following two conflicting goals:

G1. Maximizing tracking capacity. RT-Radar maximizes the number of targets or the MP-weighted sum successfully

tracked while strictly meeting all deadlines and minimum QoS accuracy requirements.

G2. Maximizing overall tracking accuracy. RT-Radar minimizes system-wide tracking error by utilizing available beam-time slack to shorten periods as much as possible, without compromising schedulability.

Achieving G1 and G2 simultaneously requires a fundamental paradigm shift: tightly integrating job-level maneuver-aware adaptation within an HRT framework. This integration elevates the tracking period from an externally determined input to the critical control knob for actively managing the capacity/accuracy trade-off. However, as detailed in Sec. I, this integration faces severe technical challenges T1–T3, stemming from the conflicts between complex physical dynamics and stringent real-time constraints. To unlock the full potential of maneuver-aware adaptation, we must address T1–T3.

B. Architecture and Workflow

The RT-Radar architecture shown in Fig. 4 is a closed-loop CPS that dynamically optimizes tracking periods by integrating physical dynamics (target maneuvers) with the cyber state (beam-time budget). To address T1–T3, RT-Radar comprises two synergistic modules forming a pipeline from physical sensing to real-time scheduling decisions. The workflow cycles continuously, triggered by each beam job’s completion.

Period Adaptation Module (PAM) addresses T1 by translating real-time physical dynamics into actionable scheduling parameters. Upon completion of $J_{i,k}$, it estimates the target’s real-time acceleration (Sec. IV-A) and leverages the transformation model in Eq. (1) to determine the dynamic period bounds $T_{i,k+1} \in [T_{i,k+1}^{\min}, T_{i,k+1}^{\max}]$ for the subsequent job $J_{i,k+1}$. These bounds are based on the estimated acceleration and pre-defined system-wide QoS requirements, denoted by $e^{\min}$ and $e^{\max}$, addressing I1. PAM then selects an initial candidate period $\hat{T}_{i,k+1}$ within these bounds. Crucially, this selection employs a proactive asymmetric policy designed to manage the hysteresis, ensuring rapid stabilization and immediate resource reclamation (Sec. IV-B).

Real-Time Beam Scheduler (RTBS) addresses T2 and T3 by guaranteeing safe zero-delay reconfiguration and navigating the non-monotonic optimization space. It receives the initial candidate periods from PAM and determines the final, optimal set of periods. To ensure mission criticality during dynamic workloads, RTBS performs a run-time schedulability analysis

(Sec. V-B). This analysis verifies both steady-state and transient schedulability under non-preemptive constraints. Based on this analysis, RTBS determines the system's load state and then invokes the corresponding optimization strategy:

- Underload: If schedulable with $\hat{T}_{i,k+1}$, the system focuses on G2 without compromising G1. RTBS utilizes the available slack to iteratively shorten periods towards $T_{i,k+1}^{\min}$.
- Overload: If overloaded but feasible, the system prioritizes G1 while minimizing G2 degradation. RTBS iteratively extends periods towards $T_{i,k+1}^{\max}$ until schedulability is restored.
- Severe-overload: If unschedulable even at $T_{i,k+1}^{\max}$, period extension is insufficient to restore G1. RTBS invokes an MP-aware policy, progressively shedding the lowest MP tasks to ensure predictable degradation.

Navigating this non-monotonic schedulability space to find the optimal periods is NP-hard. This complexity is exacerbated by non-preemptive constraints, which render utilization-based strategies fundamentally inadequate and unsafe. To ensure low-latency decision-making, RTBS employs an efficient, anomaly-aware optimization strategy: the Efficiency-First (EF) strategy. This strategy, guided by novel efficiency metrics derived from bottleneck load analysis, explicitly handles anomalies, such as plateaus (Sec. V-C).

IV. MANEUVER-AWARE PERIOD ADAPTATION

This section details how PAM achieves T1, focusing on real-time acceleration estimation and the asymmetric control policy for managing the hysteresis.

A. Real-Time Target Acceleration Estimation

The PAM computes a robust, real-time acceleration estimate $a_{i,k}$ for target τ_i in $O(1)$ time. The process is triggered immediately after $J_{i,k}$ completes. We leverage high-fidelity state estimates from an Interacting Multiple Model (IMM) tracking filter [39], the standard solution for maneuvering targets. To bridge the gap between stochastic sensor measurements and the deterministic nature of our transformation model in Eq. (1), the filter employs standard Constant Velocity (CV) and Constant Acceleration (CA) models. Furthermore, we ensure the filter's output quality via *velocity smoothing* (moving average) to dampen noise and *measurement gating* based on residual thresholds to reject outliers. From this robust output, the PAM utilizes the two most recent state vectors. Let $\vec{v}_{i,k}$ and $t_{i,k}$ denote the estimated velocity vector and measurement time for $J_{i,k}$, respectively. The average acceleration vector $\vec{a}_{i,k}$ over the interval $\Delta t \triangleq t_{i,k} - t_{i,k-1}$ is computed via finite difference: $\vec{a}_{i,k} \triangleq (\vec{v}_{i,k} - \vec{v}_{i,k-1})/\Delta t$. The magnitude $a_{i,k} = \|\vec{a}_{i,k}\|$ represents the estimated maneuverability.

B. Dynamic Period Bounds and Asymmetric Control

The PAM translates $a_{i,k}$ into scheduling parameters for $J_{i,k+1}$: the dynamic period bounds $[T_{i,k+1}^{\min}, T_{i,k+1}^{\max}]$ and the initial candidate period $\hat{T}_{i,k+1}$. This mechanism inverts the transformation model in Eq. (1) to translate the system-wide QoS error requirements, $e^{\min}$ and $e^{\max}$, into a safe temporal range, thereby addressing I1. RT-Radar operates with discrete

time slots of duration ΔT_{slot} . The continuous periods derived from the model must be discretized while strictly respecting the QoS constraints. Using the latest acceleration $a_{i,k}$ and the pre-defined system-wide period limits, denoted by $T^{\min}$ and $T^{\max}$, the per-job period bounds are computed as follows:

$$T_{i,k+1}^{\min} = \max \left(\Delta T_{\text{slot}} \cdot \left\lceil \frac{\sqrt{2 \cdot e^{\min}/a_{i,k}}}{\Delta T_{\text{slot}}} \right\rceil, T^{\min} \right), \quad (2)$$

$$T_{i,k+1}^{\max} = \min \left(\Delta T_{\text{slot}} \cdot \left\lfloor \frac{\sqrt{2 \cdot e^{\max}/a_{i,k}}}{\Delta T_{\text{slot}}} \right\rfloor, T^{\max} \right). \quad (3)$$

Rationale for asymmetric control. While the dynamic bounds define the safe operating range, selecting the initial candidate period $\hat{T}_{i,k+1}$ must address the asymmetric error behavior (hysteresis). This hysteresis stems from the inherent dynamics of adaptive tracking filters such as the IMM. During maneuver onset (acceleration increase), the filter shifts probability mass from the non-maneuvering model to the maneuvering model. The resulting convergence latency and model mismatch induce significant prediction errors [40]–[42]. Conversely, during maneuver offset (acceleration decrease), the filter already utilizes the maneuvering model, enabling a smoother transition and lower errors. Consequently, maneuver onset demands a much faster stabilization response compared to maneuver offset, thus necessitating a shorter $T_{i,k+1}$.

Asymmetric control algorithm. Relying solely on the RTBS's global optimization cannot guarantee the immediate, localized response required during these critical transient phases; any delay in stabilization risks track loss before global optimization converges. Furthermore, prompt resource reclamation is essential for efficiency. To address I2, PAM employs a proactive, $O(1)$ asymmetric control policy. This policy acts as a local *preconditioner*, stabilizing the track state locally to provide the RTBS with a safer and more efficient starting point for global optimization. The algorithm determines the initial candidate period $\hat{T}_{i,k+1}$ using the previously scheduled period $T_{i,k}$ and the new bounds defined in Eqs. (2) and (3):

$$\hat{T}_{i,k+1} = \begin{cases} T_{i,k+1}^{\min}, & \text{if } T_{i,k} > T_{i,k+1}^{\max} \quad (\text{Stabilization}), \\ T_{i,k+1}^{\max}, & \text{if } T_{i,k} < T_{i,k+1}^{\min} \quad (\text{Reclamation}), \\ T_{i,k}, & \text{otherwise} \quad (\text{Steady State}). \end{cases} \quad (4)$$

If the previous period $T_{i,k}$ exceeds the new maximum safe bound $T_{i,k+1}^{\max}$, indicating a significant increase in maneuverability during maneuver onset, the policy immediately jumps to the shortest safe period $T_{i,k+1}^{\min}$ to prioritize safety. Conversely, if $T_{i,k}$ is shorter than the new minimum required bound $T_{i,k+1}^{\min}$, indicating a significant decrease in maneuverability during maneuver offset, the policy immediately jumps to the longest safe period $T_{i,k+1}^{\max}$ to prioritize efficiency and reclaim resources. Finally, if $T_{i,k}$ remains within the new dynamic bounds, the policy retains the current period. This localized, proactive approach effectively addresses I2 by ensuring stability and optimizing efficiency, facilitating the subsequent global scheduling framework.

V. SCHEDULING FRAMEWORK FOR RT-RADAR

In this section, we present the RTBS framework, detailing the solutions for T2 and T3 essential for safe and efficient real-time beam scheduling.

A. Formal Models

We formalize the system and workload characteristics using standard real-time calculus notation.

Definition 1 (System and execution model): We model the STR system as a *uniprocessor* platform representing the antenna timeline. The system employs NP-EDF scheduling, meaning that jobs are prioritized by absolute deadlines and execute without interruption.

Definition 2 (Task and job model): The system manages a dynamic set of n beam tasks, $\mathcal{T} = \{\tau_1, \dots, \tau_n\}$. Each task τ_i has static parameters: execution time C_i and mission priority (MP) w_i . C_i represents the dwell time, which is constant and predictable, akin to WCET, due to the synchronous processing architecture of radar systems. τ_i generates a sequence of jobs $\{J_{i,k}\}$. The k -th job $J_{i,k}$ of τ_i has dynamic parameters: period $T_{i,k}$, release time $r_{i,k}$, and absolute deadline $d_{i,k}$. We assume an implicit-deadline model, where $D_{i,k} = T_{i,k}$. Temporal constraints are recursive: $r_{i,k} = d_{i,k-1}$ and $d_{i,k} = r_{i,k} + T_{i,k}$.

Definition 3 (Maneuver-aware period adaptation model): We utilize the physics-based model of Eq. (1) for maneuver-aware adaptation. Upon completion of job $J_{i,k}$, the system estimates the acceleration $a_{i,k}$. This estimation is used to determine the scheduling parameters for the subsequent job $J_{i,k+1}$. Given system-wide QoS constraints ($e^{\min}$ and $e^{\max}$), dynamic period bounds $[T_{i,k+1}^{\min}, T_{i,k+1}^{\max}]$ for $J_{i,k+1}$ are calculated as described in Sec. IV-B. The scheduled period $T_{i,k+1}$ must satisfy $T_{i,k+1}^{\min} \leq T_{i,k+1} \leq T_{i,k+1}^{\max}$. The predicted error for $J_{i,k+1}$ is then defined as: $\hat{e}_{i,k+1} = \frac{1}{2}a_{i,k}T_{i,k+1}^2$.

Definition 4 (Dynamic reconfiguration model): The system state evolves through reconfiguration points $\{t_m\}$. Let $\mathcal{T}_m$ be the active task set during $[t_m, t_{m+1})$. A reconfiguration point t_{m+1} occurs *only* upon a job completion. This constraint reflects the inherent STR operational flow and is crucial for the subsequent non-preemptive analysis in Lemma 2. At t_{m+1} , the system transitions $\mathcal{T}_m \rightarrow \mathcal{T}_{m+1}$ due to task addition/removal or period adaptation. An optimization executes to co-optimize G1 and G2, modeled as the highest priority task deferring beam execution. Its WCET, denoted by C^{OP} , depends on the specific optimization path ($C_{\text{under}}^{\text{OP}}$, $C_{\text{over}}^{\text{OP}}$, $C_{\text{severe}}^{\text{OP}}$), as detailed in Sec. V-D. We employ the *zero-delay update rule* to apply the resulting optimal periods $\{T_i^{\text{OP}}\}$. Let $J_{i,e}$ be the earliest pending or unreleased job of τ_i . Upon optimization completion, its absolute deadline is immediately updated: $d_{i,e} = r_{i,e} + T_{i,e}^{\text{OP}}$.

B. Schedulability Analysis

To address T2, we develop a schedulability analysis that accounts for both steady-state and transient dynamics under NP-EDF. Our analysis builds upon the established condition for conventional STR systems with pre-defined static periods.

Lemma 1 (Static NP-EDF schedulability, from [43]): Consider a set of implicit-deadline periodic tasks $\mathcal{T} = \{\tau_i = (C_i, T_i)\}_{i=1}^n$, indexed in non-decreasing order of periods. $\mathcal{T}$ is schedulable under NP-EDF, if the NP-EDF load $\mathbf{L}_h$ for every τ_h satisfies:

$$\forall h \quad \mathbf{L}_h = \mathbf{U}_h + \mathbf{B}_h \leq 1, \quad (5)$$

where $\mathbf{U}_h = \sum_{i=1}^h \frac{C_i}{T_i}$ is the utilization load, and $\mathbf{B}_h = \frac{B_h}{T_h}$ is the blocking load, with $B_h = \max(\{0\} \cup \{C_i \mid T_i > T_h\})$ being the maximum blocking time for τ_h .

Proof: We prove by contradiction. Suppose a deadline miss occurs at time t_0 . Consider the longest interval of length L ending at t_0 , during which the processor continuously executes jobs with deadlines no later than t_0 , except possibly for one blocking job executing at the beginning of the interval with a deadline strictly after t_0 . The total processor demand $W(L)$ within this interval comprises: (i) the execution time of jobs released and having deadlines within the interval, and (ii) the execution time of the blocking job (if any). Since the blocking job has a deadline after t_0 and is already executing at $t_0 - L$, its release time must precede $t_0 - L$; under the implicit-deadline model, this implies its period satisfies $T_i > L$. Let h be the largest task index satisfying $T_h \leq L$, so the blocking job's period exceeds T_h and its execution time is bounded by B_h . The demand is thus bounded by:

$$W(L) \leq \sum_{i=1}^h \left\lfloor \frac{L}{T_i} \right\rfloor C_i + B_h.$$

A deadline miss implies $W(L) > L$. Applying the inequality $\lfloor x \rfloor \leq x$, we obtain $L < L \cdot \sum_{i=1}^h \frac{C_i}{T_i} + B_h = L \cdot \mathbf{U}_h + B_h$. Dividing by L and using $T_h \leq L$ yields $1 < \mathbf{U}_h + \frac{B_h}{L} \leq \mathbf{U}_h + \frac{B_h}{T_h} = \mathbf{U}_h + \mathbf{B}_h$, which contradicts Eq. (5). ■

While Lemma 1 verifies steady-state schedulability, it is insufficient for RT-Radar. Def. 4 implies that immediate deadline alterations can induce a transient overload that static analysis misses. We therefore develop a novel transient analysis to guarantee safety during these dynamic reconfigurations.

Lemma 2 (Transient NP-EDF schedulability): Consider a transition from a schedulable task set $\mathcal{T}_m$ to $\mathcal{T}_{m+1}$ at t_{m+1} , incurring WCET overhead C^{OP} corresponding to the optimization path taken. Let $t_{m+1}^{\text{OP}} = t_{m+1} + C^{\text{OP}}$ be the time when beam execution resumes. For each task $\tau_i \in \mathcal{T}_{m+1}$, let $\mathcal{D}_i = d_{i,e} - t_{m+1}^{\text{OP}}$ be the residual deadline of $J_{i,e}$ relative to t_{m+1}^{OP} , and $\mathcal{E}\mathcal{D}_i = \min(\mathcal{D}_i, T_i)$ be its effective deadline. Assuming tasks are indexed in non-decreasing order of $\mathcal{D}_i$, the system remains schedulable after the transition if:

- 1) (Immediate safety) $\forall i : \mathcal{D}_i > 0$,
- 2) (Steady-state safety) $\mathcal{T}_{m+1}$ satisfies Eq. (5), and
- 3) (Transient safety) The transient load $\mathbf{TL}_h$ for τ_h satisfies:

$$\forall h \quad \mathbf{TL}_h = \sum_{i=1}^h \frac{C_i}{\mathcal{E}\mathcal{D}_i} \leq 1. \quad (6)$$

Proof: We prove by contradiction. Condition 1 guarantees that no deadline miss occurs during the optimization interval $[t_{m+1}, t_{m+1}^{\text{OP}}]$. Assume the conditions hold, yet a deadline miss

occurs at time $t > t_{m+1}^{\text{OP}}$. Following the standard busy interval analysis [43], let t' be the latest time before t such that there are no pending requests arriving before t' with deadlines at or before t . Crucially, Def. 4 ensures that no non-preemptive blocking is carried over from $\mathcal{T}_m$ into the interval $[t_{m+1}, t]$.

i) *Case 1* ($t' \geq t_{m+1}^{\text{OP}}$): The interval is entirely governed by $\mathcal{T}_{m+1}$. A deadline miss implies that $\mathcal{T}_{m+1}$ violates the static condition, which contradicts Condition 2.

ii) *Case 2* ($t' < t_{m+1}^{\text{OP}}$): Since $\mathcal{T}_m$ was schedulable and C^{OP} executes at the highest priority per Def. 4, the overload must be induced by $\mathcal{T}_{m+1}$ starting at t_{m+1}^{OP} . Let $\delta = t - t_{m+1}^{\text{OP}}$. A deadline miss implies that the total processor demand $W(\delta)$ strictly exceeds δ . This demand is generated by tasks in $\mathcal{T}_{m+1}$ whose residual deadlines satisfy $\mathcal{D}_i \leq \delta$. Let h be the largest task index such that $\mathcal{D}_h \leq \delta$. The demand is bounded by:

$$W(\delta) \leq \sum_{i=1}^h \left(\left\lfloor \frac{\delta - \mathcal{D}_i}{T_i} \right\rfloor + 1 \right) C_i.$$

If $W(\delta) > \delta$, applying the inequality $\lfloor x \rfloor \leq x$ yields a necessary condition for the deadline miss:

$$\delta < \sum_{i=1}^h \left(\frac{\delta - \mathcal{D}_i}{T_i} + 1 \right) C_i.$$

Dividing both sides by δ and rearranging terms yields:

$$1 < \sum_{i=1}^h \left(1 + \frac{T_i - \mathcal{D}_i}{\delta} \right) \frac{C_i}{T_i}.$$

We analyze the contribution of each task τ_i :

- If $\mathcal{D}_i \leq T_i$ (pending job): Since $\mathcal{D}_i \leq \delta$,

$$\left(1 + \frac{T_i - \mathcal{D}_i}{\delta} \right) \frac{C_i}{T_i} \leq \left(1 + \frac{T_i - \mathcal{D}_i}{\mathcal{D}_i} \right) \frac{C_i}{T_i} = \frac{C_i}{\mathcal{D}_i}.$$

- If $\mathcal{D}_i > T_i$ (unreleased job):

$$\left(1 + \frac{T_i - \mathcal{D}_i}{\delta} \right) \frac{C_i}{T_i} < \frac{C_i}{T_i}.$$

Combining these bounds yields:

$$1 < \sum_{i=1}^h \frac{C_i}{\min(\mathcal{D}_i, T_i)} = \sum_{i=1}^h \frac{C_i}{\mathcal{E}\mathcal{D}_i},$$

which contradicts Condition 3. ■

By integrating the static (Lemma 1) and transient (Lemma 2) conditions, we establish the rigorous schedulability theorem for dynamic RT-Radar, effectively addressing T2.

Theorem 1 (Dynamic NP-EDF schedulability): A system characterized by the dynamic reconfiguration model (Def. 4) undergoing transitions $\{\mathcal{T}_m \rightarrow \mathcal{T}_{m+1}\}$ is schedulable for all time $t \geq t_0$ if:

- 1) The initial task set $\mathcal{T}_0$ satisfies Lemma 1, and
- 2) Every transition at t_{m+1} ($m \geq 0$), satisfies Lemma 2.

Proof: By induction on m . Base case ($m = 0$) holds by Condition 1. Assuming the system is schedulable up to t_{m+1} , Condition 2 guarantees the schedulability of $\mathcal{T}_{m+1}$ after t_{m+1} . Thus, the system remains schedulable. ■

C. Optimization Strategy

The RTBS addresses T3 by determining the optimal periods at each reconfiguration point t_m , subject to Theorem 1. This subsection formalizes the optimization problem, analyzes its complexity, and develops an efficient optimization strategy.

Definition 5 (RTBS optimization problem, $RTBS^{\text{OP}}$): At every reconfiguration point t_m , given $\mathcal{T} = \{\tau_i\}$ and current accelerations $\{a_i\}$. Let $\mathcal{P}_i$ be the discrete periods available for T_i within $[T_i^{\min}, T_i^{\max}]$. $RTBS^{\text{OP}}$ seeks a period vector $\mathbf{T} = (T_1, \dots, T_n)$ minimizing the system-wide average predicted tracking error $E(\mathbf{T})$, subject to:

- 1) Period selection: $\forall i : T_i \in \mathcal{P}_i$.
- 2) Schedulability: $\mathcal{T}$ with periods $\mathbf{T}$ satisfies Theorem 1.

The objective function $E(\mathbf{T})$ is defined as:

$$E(\mathbf{T}) = \frac{1}{n} \sum_{i=1}^n (\hat{e}_i(\mathbf{T})). \quad (7)$$

We now establish the computational intractability and structural properties of this problem.

Lemma 3: The $RTBS^{\text{OP}}$ is NP-hard.

Proof: We prove this lemma by restriction to the *Multiple-Choice Knapsack Problem (MCKP)*, which is known to be NP-hard [44]. Consider an instance of $RTBS^{\text{OP}}$ relaxed to a simple utilization bound ($\sum_{i=1}^n C_i/T_i \leq 1$). This requires selecting one period $T_i \in \mathcal{P}_i$ (item) per task (group) such that total utilization (weight) is bounded while minimizing $E(\mathbf{T})$ (maximizing value). This is equivalent to MCKP. Since a restriction is NP-hard, $RTBS^{\text{OP}}$ is also NP-hard. ■

The NP-hardness of $RTBS^{\text{OP}}$ necessitates an efficient strategy. We analyze the structural properties of the objective function and constraints to design an effective greedy heuristic.

Lemma 4 (Convexity of error vs. utilization): The predicted tracking error $\hat{e}_i$ of a task τ_i is a strictly convex function of its utilization $U_i = C_i/T_i$.

Proof: Expressing error in terms of utilization: $\hat{e}_i(U_i) = \frac{1}{2}a_i(\frac{C_i}{U_i})^2 = k_i U_i^{-2}$, where $k_i = \frac{1}{2}a_i C_i^2$. The second derivative $\frac{d^2 \hat{e}_i}{dU_i^2} = 6k_i U_i^{-4}$ is strictly positive since $k_i > 0$ and $U_i > 0$. Thus, $\hat{e}_i(U_i)$ is strictly convex. ■

While Lemma 4 suggests gradient-based approaches due to the well-behaved objective function, the NP-EDF constraints introduce significant complexity. Safely optimizing $\mathbf{T}$ requires analyzing the impact of period changes on the individual load metrics defined in Lemma 1. Crucially, the interaction between utilization $\mathbf{U}_h$ and blocking $\mathbf{B}_h$ loads behaves counter-intuitively when periods, and thus EDF priorities, change.

Lemma 5 (Non-monotonicity of individual NP-EDF load): Let $\mathbf{L}_h(\mathbf{T}) = \mathbf{U}_h(\mathbf{T}) + \mathbf{B}_h(\mathbf{T})$ denote the NP-EDF load metric for τ_h under $\mathbf{T}$. $\mathbf{L}_h(\mathbf{T})$ is non-monotonic with respect to changes in periods $\mathbf{T} \rightarrow \mathbf{T}'$.

Proof: We prove by counterexample. Period changes can alter the period-based sorted order required by Lemma 1, causing discontinuous changes in the load metric. Consider $\mathcal{T}$ with $\mathbf{T}: \tau_1 = (3, 15), \tau_2 = (4, 15), \tau_3 = (5, 20)$. Initially, $T_1 =$

$T_2 < T_3$. The load for τ_2 is $\mathbf{L}_2(\mathbf{T}) = (3/15 + 4/15) + (B_2/15)$. $B_2 = C_3 = 5$. Thus, $\mathbf{L}_2(\mathbf{T}) = 12/15 = 0.8$. Now, reduce τ_2 's period to $T'_2 = 14$. The new order is $T'_2 < T_1 < T_3$. The load for τ_2 becomes $\mathbf{L}_2(\mathbf{T}') = (4/14) + (B'_2/14)$. $B'_2 = \max(C_1, C_3) = 5$. Thus, $\mathbf{L}_2(\mathbf{T}') = 9/14 \approx 0.643$. Although the utilization of τ_2 increased, its individual NP-EDF load decreased due to the period-based reordering. Thus, the NP-EDF load metric is non-monotonic. ■

Lemma 5 reveals the critical challenge for $RTBS^{\text{OP}}$: the load of an individual task can behave anomalously due to period-based reordering. This complexity invalidates conventional optimization strategies. Utilization-based approaches are inherently unsafe as they ignore non-preemptive blocking. Furthermore, strategies relying on localized load metrics are unreliable; due to global shifts in the period-based sorted order, a locally optimal move may paradoxically increase the system's bottleneck load. To ensure safe optimization, we must focus on the metric that defines the actual schedulability boundary of the entire system.

Definition 6 (System bottleneck load): The system bottleneck load $\mathbf{L}_B(\mathbf{T})$ under period vector $\mathbf{T}$ is the maximum NP-EDF load metric across all tasks: $\mathbf{L}_B(\mathbf{T}) = \max_h \mathbf{L}_h(\mathbf{T})$.

The system is schedulable in steady-state if $\mathbf{L}_B(\mathbf{T}) \leq 1$. Therefore, $\mathbf{L}_B(\mathbf{T})$ is the definitive measure that guides our optimization strategy. Crucially, while individual loads $\mathbf{L}_h(\mathbf{T})$ are non-monotonic, $\mathbf{L}_B(\mathbf{T})$ behaves more predictably.

Lemma 6 (Monotonicity and plateaus of $\mathbf{L}_B(\mathbf{T})$): The system bottleneck load $\mathbf{L}_B(\mathbf{T})$ is monotonic with respect to changes in period $\mathbf{T} \rightarrow \mathbf{T}'$. However, $\mathbf{L}_B(\mathbf{T})$ is not strictly monotonic; period changes may not change $\mathbf{L}_B(\mathbf{T})$, resulting in plateau regions.

Proof: The schedulability condition of Lemma 1 is sustainable. Therefore, decreasing utilization through a longer period cannot degrade the system's overall schedulability margin. Since $\mathbf{L}_B(\mathbf{T})$ represents this margin, it must monotonically decrease or stay the same. To show that plateaus exist, consider $\mathcal{T}$ with $\mathbf{T}: \tau_1 = (3, 10), \tau_2 = (5, 15)$. $\mathbf{L}_1(\mathbf{T}) = 3/10 + 5/10 = 0.8$, and $\mathbf{L}_2(\mathbf{T}) = 3/10 + 5/15 \approx 0.633$, yielding $\mathbf{L}_B(\mathbf{T}) = 0.8$. If we extend τ_2 's period to $T'_2 = 20$, $\mathbf{L}_1(\mathbf{T}') = 0.8$ and $\mathbf{L}_2(\mathbf{T}') = 0.55$, yielding $\mathbf{L}_B(\mathbf{T}') = 0.8$. Utilization decreased, but the bottleneck load remained unchanged. ■

Lemma 6 guarantees that the optimization process is stable, meaning no backward moves occur, but the existence of plateaus introduces two critical anomalies that we must explicitly handle to ensure efficiency and convergence of $RTBS^{\text{OP}}$.

- “Free Lunch” (FL) anomaly: A period reduction improves accuracy without increasing the bottleneck load.
- “Ineffective Move” (IM) anomaly: A period extension degrades accuracy without decreasing the bottleneck load.

To effectively co-optimize G1 and G2, we design a novel greedy Efficiency-First (EF) strategy. This strategy requires carefully defined metrics that quantify the trade-off between the objective function $E(\mathbf{T})$ and the constraint of system bottleneck load $\mathbf{L}_B(\mathbf{T})$. The goal is to achieve the maximum

improvement in $E(\mathbf{T})$ relative to the increase in $\mathbf{L}_B(\mathbf{T})$ during underload, or the maximum reduction in $\mathbf{L}_B(\mathbf{T})$ relative to the degradation in $E(\mathbf{T})$ during overload. Furthermore, these metrics must explicitly handle FL and IM anomalies to ensure local optimality and guarantee convergence.

Definition 7 (Efficiency metrics): Let $\mathbf{T}'(i)$ be the period vector resulting from changing τ_i 's period by a single step ΔT_{slot} . We define the changes in predicted error $\Delta \hat{e}(i)$ and bottleneck load $\Delta \mathbf{L}_B(i)$:

- $\Delta \hat{e}(i) = \hat{e}_i(\mathbf{T}'(i)) - \hat{e}_i(\mathbf{T})$, $\Delta \mathbf{L}_B(i) = \mathbf{L}_B(\mathbf{T}'(i)) - \mathbf{L}_B(\mathbf{T})$.

We define the benefit β_i and cost γ_i with state-dependent semantics. Although their specific formulations differ across regimes (distinguished by $\bar{\cdot}$ for underload and $\tilde{\cdot}$ for overload), they consistently represent the algorithmic gain and penalty for the optimization objective. This establishes a unified efficiency metric η_i designed to explicitly handle anomalies:

- Underload (period reduction): Maximizing accuracy.

$$\begin{aligned} \bar{\beta}_i &= -\Delta \hat{e}(i) \text{ (error reduction)}, \quad \bar{\gamma}_i = \Delta \mathbf{L}_B(i) \text{ (load increase)}, \\ \mathcal{T}_{\text{FL}} &= \{\tau_i \in \mathcal{T} \mid \bar{\gamma}_i = 0\} \text{ (set of FL moves)}, \\ \bar{\eta}_i &= \begin{cases} \bar{\beta}_i, & \text{if } \mathcal{T}_{\text{FL}} \neq \emptyset \text{ and } \tau_i \in \mathcal{T}_{\text{FL}} \text{ (prioritize FL)}, \\ -\infty, & \text{if } \mathcal{T}_{\text{FL}} \neq \emptyset \text{ and } \tau_i \notin \mathcal{T}_{\text{FL}} \text{ (exclude non-FL)}, \\ \bar{\beta}_i / \bar{\gamma}_i, & \text{if } \mathcal{T}_{\text{FL}} = \emptyset \text{ (maximize efficiency)}; \end{cases} \end{aligned} \quad (8)$$

- Overload (period extension): Restoring schedulability.

$$\begin{aligned} \tilde{\beta}_i &= -\Delta \mathbf{L}_B(i) \text{ (load reduction)}, \quad \tilde{\gamma}_i = \Delta \hat{e}(i) \text{ (error increase)}, \\ \mathcal{T}_{\text{PM}} &= \{\tau_i \in \mathcal{T} \mid \tilde{\beta}_i > 0\} \text{ (set of productive moves)}, \\ \tilde{\eta}_i &= \begin{cases} \tilde{\beta}_i / \tilde{\gamma}_i, & \text{if } \mathcal{T}_{\text{PM}} \neq \emptyset \text{ and } \tau_i \in \mathcal{T}_{\text{PM}} \text{ (maximize efficiency)}, \\ -\infty, & \text{if } \mathcal{T}_{\text{PM}} \neq \emptyset \text{ and } \tau_i \notin \mathcal{T}_{\text{PM}} \text{ (exclude IM)}, \\ 1 / \tilde{\gamma}_i, & \text{if } \mathcal{T}_{\text{PM}} = \emptyset \text{ (minimize error for convergence)}. \end{cases} \end{aligned} \quad (9)$$

These metrics ensure progress: FL moves are prioritized during underload, and the strategy minimizes the error cost to escape the plateau if stuck in an IM-only state during overload.

Efficient incremental load calculation. Calculating the bottleneck load change $\Delta \mathbf{L}_B$ is computationally intensive because a local period adjustment for τ_i globally alters the period-based sorted order required by Lemma 1, potentially affecting the utilization $\mathbf{U}_j$ and blocking $\mathbf{B}_j$ loads of other tasks τ_j (Lemma 5). While a full recomputation of all loads $\mathbf{L}_j$ incurs $O(n^2)$ time, RTBS employs an efficient $O(n)$ incremental approach to enable low-latency run-time optimization.

At each reconfiguration point t_m , we first establish the full baseline for both $\mathbf{L}_h$ and $\mathbf{TL}_h$. This initialization is mandatory because the active task set evolves and $\mathcal{D}_i$ depends on the specific t_{m+1}^{OP} . Subsequent evaluations of candidate periods utilize incremental logic based on the shift in relative indices within the sorted list. For the static schedulability test in Lemma 1, tasks are indexed by non-decreasing period. Consider a period reduction for τ_i ($T_i \rightarrow T'_i$, where $T'_i = T_i - \Delta T_{\text{slot}}$). The impact on τ_j depends on its period T_j relative to T_i and T'_i :

- Tasks with larger periods ($T_i \leq T_j$): The index of τ_i remains smaller than that of τ_j . $\mathbf{U}_j$ increases due to the increased utilization of τ_i . $\mathbf{B}_j$ remains unchanged.

$$\Delta \mathbf{U}_j(i) = \frac{C_i(T_i - T'_i)}{T_i T'_i}, \quad \Delta \mathbf{B}_j(i) = 0.$$

- Tasks with smaller periods ($T_j < T'_i$): The index of τ_i remains larger than that of τ_j . Both $\mathbf{U}_j$ and $\mathbf{B}_j$ remain unchanged.

$$\Delta \mathbf{U}_j(i) = 0, \quad \Delta \mathbf{B}_j(i) = 0.$$

- Tasks with intermediate periods ($T'_i \leq T_j < T_i$): The index of τ_i shifts from larger to smaller than that of τ_j . This changes both $\mathbf{U}_j$ (as τ_i is now included) and $\mathbf{B}_j$ (as the set of blocking tasks changes).

$$\Delta \mathbf{U}_j(i) = \frac{C_i}{T'_i}, \quad \Delta \mathbf{B}_j(i) = \frac{\max(\{0\} \cup \{C_a \mid T_a > T_j\})}{T_j} - \frac{B_j}{T_j}.$$

- Self-impact ($j = i$): The position shift affects which tasks are included in $\mathbf{U}_i$ and $\mathbf{B}_i$.

$$\Delta \mathbf{U}_i(i) = \frac{C_i(T_i - T'_i)}{T_i T'_i} - \sum_{T'_i < T_a \leq T_i} \frac{C_a}{T_a},$$

$$\Delta \mathbf{B}_i(i) = \frac{\max(\{0\} \cup \{C_a \mid T_a > T'_i\})}{T'_i} - \frac{B_i}{T'_i}.$$

The rules for period extension are symmetric. Similarly, the transient load change $\Delta \mathbf{TL}_j$ follows the same logical structure by substituting the period T with the residual deadline $\mathcal{D}$. Notably, to strictly bound the complexity to $O(n)$, we explicitly maintain the B_j values using a *suffix-maximum array* within the period-sorted task list. Upon a task's period change, a single reverse pass updates the suffix maximums strictly within the affected range, thereby eliminating the need to re-scan the entire list. In contrast, since $\mathbf{TL}_h$ in Eq. (6) is a cumulative sum without blocking terms, updates are propagated linearly to affected tasks in $O(n)$ by maintaining tasks sorted by $\mathcal{D}$.

By deriving these incremental changes for static $\Delta \mathbf{U}_j, \Delta \mathbf{B}_j$ and transient $\Delta \mathbf{TL}_j$ load metrics, we efficiently update all affected loads. This approach applies to both period changes and task admission/shedding, enabling determination of the new $\mathbf{L}_B$ in $O(n)$ time. Based on the efficiency metrics and incremental load calculation, we define our greedy EF strategy.

Definition 8 (Efficiency-First (EF) strategy): The EF strategy selects the task that maximizes the efficiency metric η_i in Def. 7, aiming to minimize the average tracking error $E(\mathbf{T})$:

$$\tau_{\text{EF}} = \arg \max_i \{\eta_i\}. \quad (10)$$

We now establish the local optimality of this strategy in navigating the complex optimization space characterized by convexity in Lemma 4 and non-monotonicity in Lemma 5.

Lemma 7 (Local optimality and convergence of EF): The EF strategy is locally optimal for minimizing $E(\mathbf{T})$ in a single iteration of period adjustment and guarantees convergence to a schedulable solution.

Proof: (i) Local optimality: the objective is to optimize $E(\mathbf{T})$ relative to $\mathbf{L}_B(\mathbf{T})$. By Lemma 4, maximizing the ratio β_i/γ_i corresponds to the steepest descent for $E(\mathbf{T})$ during underload and the gentlest ascent during overload. Def. 7

Algorithm 1 Real-time beam scheduler (RTBS)

```

1: Input:  $\mathcal{T}, \hat{\mathbf{T}}$    Output:  $\mathcal{T}^*, \mathbf{T}^*$ 
2: if  $(\mathcal{T}, \hat{\mathbf{T}})$  satisfies Theorem 1 with  $C_{\text{under}}^{\text{OP}}$  then
3:    $\mathbf{T}^* \leftarrow \text{AccuracyMaximization}(\mathcal{T}, \hat{\mathbf{T}})$ ;  $\mathcal{T}^* \leftarrow \mathcal{T}$     $\triangleright$  Alg. 2
4: else
5:    $\mathbf{T}^{\text{max}} \leftarrow \{T_i^{\text{max}} \mid \tau_i \in \mathcal{T}\}$ 
6:   if  $(\mathcal{T}, \mathbf{T}^{\text{max}})$  satisfies Theorem 1 with  $C_{\text{over}}^{\text{OP}}$  then
7:      $\mathbf{T}^* \leftarrow \text{CapacityMaximization}(\mathcal{T}, \hat{\mathbf{T}})$ ;  $\mathcal{T}^* \leftarrow \mathcal{T}$     $\triangleright$  Alg. 3
8:   else
9:      $(\mathcal{T}^*, \mathbf{T}^*) \leftarrow \text{MPawareShedding}(\mathcal{T}, \mathbf{T}^{\text{max}})$     $\triangleright$  Alg. 4
10:  end if
11: end if
12: return  $(\mathcal{T}^*, \mathbf{T}^*)$ 

```

Algorithm 2 Accuracy maximization (for underload)

```

1: Input:  $\mathcal{T}, \hat{\mathbf{T}}$    Output:  $\mathbf{T}^*$ 
2:  $\mathbf{T}^* \leftarrow \hat{\mathbf{T}}$ 
3: while True do
4:    $\mathbf{T}_{\text{next}} \leftarrow \text{null}$ 
5:    $\mathcal{C} \leftarrow \{\tau_i \in \mathcal{T} \mid T_i^* - \Delta T_{\text{slot}} \geq T_i^{\text{min}}\}$ ;  $\mathcal{S} \leftarrow \emptyset$ ;  $\mathcal{T}_{\text{FL}} \leftarrow \emptyset$ 
6:   for all  $\tau_i \in \mathcal{C}$  do
7:      $\mathbf{T}' \leftarrow \mathbf{T}^*$  with  $T'_i = T_i^* - \Delta T_{\text{slot}}$ 
8:     if  $(\mathcal{T}, \mathbf{T}')$  satisfies Theorem 1 with  $C_{\text{under}}^{\text{OP}}$  then
9:       Calculate  $\bar{\beta}_i, \bar{\gamma}_i$  by Eq. (8)
10:      if  $\bar{\gamma}_i = 0$  then  $\mathcal{T}_{\text{FL}} \leftarrow \mathcal{T}_{\text{FL}} \cup \{\tau_i\}$ 
11:      end if
12:       $\mathcal{S} \leftarrow \mathcal{S} \cup \{(\tau_i, \mathbf{T}', \bar{\beta}_i, \bar{\gamma}_i)\}$ 
13:    end if
14:  end for
15:  if  $\mathcal{S} \neq \emptyset$  then
16:    Calculate  $\bar{\eta}_i$  for  $\tau_i \in \mathcal{S}$  by Eq. (8)
17:     $(\tau_{\text{EF}}, \mathbf{T}'_{\text{EF}}) \leftarrow \arg \max_{\tau_i \in \mathcal{S}} \{\bar{\eta}_i\}$ ;  $\mathbf{T}_{\text{next}} \leftarrow \mathbf{T}'_{\text{EF}}$ 
18:  end if
19:  if  $\mathbf{T}_{\text{next}} \neq \text{null}$  then  $\mathbf{T}^* \leftarrow \mathbf{T}_{\text{next}}$ 
20:  else break
21: end if
22: end while
23: return  $\mathbf{T}^*$ 

```

explicitly prioritizes FL moves, which provide the maximum benefit $\bar{\beta}_i$ at zero cost. This prioritization ensures that the locally optimal move is always selected. (ii) Convergence: during overload, if $\mathcal{T}_{\text{PM}} \neq \emptyset$, EF selects a move that strictly reduces $\mathbf{L}_B(\mathbf{T})$. If stuck on an IM-only state where $\mathcal{T}_{\text{PM}} = \emptyset$, EF selects the move that minimizes the error increase $\bar{\gamma}_i$ (equivalent to maximizing $1/\bar{\gamma}_i$). This guarantees a change in the system configuration $\mathbf{T}$, enabling the strategy to escape the plateau and eventually reach a schedulable state. ■

D. RTBS Algorithms

This subsection details the RTBS algorithms, demonstrating how the schedulability analysis in Sec. V-B and the bottleneck-guided optimization strategy in Sec. V-C are integrated.

Algorithm 1. The RTBS orchestrates the optimization process at each reconfiguration point. It determines the system's load state using Theorem 1 to ensure strict adherence to HRT constraints. It assesses the initial configuration using $C_{\text{under}}^{\text{OP}}$ (Line 2), and the maximum feasible capacity using $C_{\text{over}}^{\text{OP}}$ (Line 6). Based on this assessment, RTBS invokes

Algorithm 3 Capacity maximization (for overload)

```

1: Input:  $\mathcal{T}, \hat{\mathbf{T}}$    Output:  $\mathbf{T}^*$ 
2:  $\mathbf{T}^* \leftarrow \hat{\mathbf{T}}$ 
3: while  $(\mathcal{T}, \mathbf{T}^*)$  violates Theorem 1 with  $C_{\text{over}}^{\text{OP}}$  do
4:    $\mathcal{C} \leftarrow \{\tau_i \in \mathcal{T} \mid T_i^* + \Delta T_{\text{slot}} \leq T_i^{\text{max}}\}, \mathcal{S} \leftarrow \emptyset; \mathcal{T}_{\text{PM}} \leftarrow \emptyset$ 
5:   for all  $\tau_i \in \mathcal{C}$  do
6:      $\mathbf{T}' \leftarrow \mathbf{T}^*$  with  $T_i' = T_i^* + \Delta T_{\text{slot}}$ 
7:     Calculate  $\tilde{\beta}_i, \tilde{\gamma}_i$  by Eq. (9)
8:     if  $\tilde{\beta}_i > 0$  then  $\mathcal{T}_{\text{PM}} \leftarrow \mathcal{T}_{\text{PM}} \cup \{\tau_i\}$ 
9:   end if
10:   $\mathcal{S} \leftarrow \mathcal{S} \cup \{(\tau_i, \mathbf{T}', \tilde{\beta}_i, \tilde{\gamma}_i)\}$ 
11: end for
12: Calculate  $\tilde{\eta}_i$  for  $\tau_i \in \mathcal{S}$  by Eq. (9)
13:  $(\tau_{\text{EF}}, \mathbf{T}'_{\text{EF}}) \leftarrow \arg \max_{\tau_i \in \mathcal{S}} \{\tilde{\eta}_i\}; \mathbf{T}^* \leftarrow \mathbf{T}'_{\text{EF}}$ 
14: end while
15: return  $\mathbf{T}^*$ 

```

Algorithm 4 MP-aware shedding (for severe-overload)

```

1: Input:  $\mathcal{T}, \mathbf{T}^{\text{max}}$    Output:  $\mathcal{T}^*, \mathbf{T}^*$ 
2:  $\mathcal{T}^* \leftarrow \mathcal{T}, \mathbf{T}^* \leftarrow \mathbf{T}^{\text{max}}$ 
3:  $\mathcal{T}_{\text{sorted}} \leftarrow \text{sort tasks in } \mathcal{T} \text{ by ascending MP } (w_i)$ 
4: for all  $\tau_{\text{shed}} \in \mathcal{T}_{\text{sorted}}$  do
5:   if  $(\mathcal{T}^*, \mathbf{T}^*)$  satisfies Theorem 1 with  $C_{\text{severe}}^{\text{OP}}$  then break
6:   end if
7:    $\mathcal{T}^* \leftarrow \mathcal{T}^* \setminus \{\tau_{\text{shed}}\}; \mathbf{T}^* \leftarrow \mathbf{T}^{\text{max}}$  for tasks in  $\mathcal{T}^*$ 
8: end for
9: return  $(\mathcal{T}^*, \mathbf{T}^*)$ 

```

the appropriate subroutine: accuracy maximization, capacity maximization, or MP-aware shedding (Algorithms 2, 3, or 4).

Algorithm 2. When *underloaded*, the algorithm maximizes accuracy by iteratively reducing periods in discrete steps ΔT_{slot} using the EF strategy. In each iteration, it evaluates all feasible period reductions using the $O(n)$ incremental load calculation, verifying safety with $C_{\text{under}}^{\text{OP}}$ (Lines 6–9). Crucially, it identifies “Free Lunch” (FL) moves where $\tilde{\gamma}_i = 0$ (Line 10). The efficiency metric calculation in Eq. (8) strictly prioritizes FL moves. If multiple FL moves exist, it maximizes the benefit $\tilde{\beta}_i$. Otherwise, it selects the move maximizing the error reduction to load increase ratio $\tilde{\beta}_i/\tilde{\gamma}_i$, ensuring the steepest descent of $E(\mathbf{T})$ as established in Lemma 7 (Lines 16–17).

Algorithm 3. When *overloaded*, the algorithm restores schedulability by iteratively extending periods in discrete steps ΔT_{slot} until Theorem 1 with $C_{\text{over}}^{\text{OP}}$ is satisfied (Line 3). The EF strategy addresses the critical challenge of “Ineffective Moves” (IM). It identifies productive moves $\mathcal{T}_{\text{PM}}$ that genuinely reduce the bottleneck load, where $\tilde{\beta}_i > 0$ (Line 8). The efficiency metric $\tilde{\eta}_i$ in Eq. (9) strictly focuses on $\mathcal{T}_{\text{PM}}$ if available, maximizing the load reduction to error increase ratio $\tilde{\beta}_i/\tilde{\gamma}_i$. Critically, if trapped in an IM-only state (i.e., $\mathcal{T}_{\text{PM}} = \emptyset$), $\tilde{\eta}_i$ maximizes $1/\tilde{\gamma}_i$ —minimizing the error increase—to escape the plateau (Lines 12–13). This guarantees convergence by escaping the plateau deadlock, as established in Lemma 7.

Algorithm 4. If the system remains unschedulable even at maximum periods $\mathbf{T}^{\text{max}}$ (i.e., *severely-overloaded*), this algorithm ensures predictable degradation. It systematically sheds tasks in ascending order of MP (Line 3) until a schedulable subset is identified using Theorem 1 with $C_{\text{severe}}^{\text{OP}}$ (Line 5), guaranteeing the schedulability of critical tasks.

TABLE I: Maximum observed overhead of RTBS_{EF} (ms)

n	K_{under}	$K_{\text{under}}^{\text{actual}}$	$C_{\text{under}}^{\text{OP}}$	K_{over}	$K_{\text{over}}^{\text{actual}}$	$C_{\text{over}}^{\text{OP}}$	$C_{\text{severe}}^{\text{OP}}$
60	2525	1000	34.06	1468	28	1.09	0.00
70	2943	1121	41.27	1748	47	2.12	0.00
80	3360	954	37.59	1979	96	4.80	0.10
90	3055	692	28.38	2192	162	8.63	1.02
100	2027	635	23.91	2384	208	12.06	3.13

We conclude this section by analyzing the theoretical properties and run-time complexity of the RTBS algorithm.

Theorem 2 (Termination and correctness): The RTBS algorithm (Algorithm 1) terminates in a finite number of steps and guarantees that the resulting task set $\mathcal{T}^*$ and period vector $\mathbf{T}^*$ are schedulable under NP-EDF.

Proof: (i) Termination: The search space $\mathcal{P}_i$ is finite and bounded. In Algorithms 2 and 3, periods strictly move towards T_i^{min} and T_i^{max} , respectively. Since the periods move monotonically within a finite space, the algorithms terminate. (ii) Schedulability guarantee: RTBS verifies Theorem 1 at every decision point. Algorithm 2 only accepts schedulable moves. Algorithm 3 terminates upon restoring schedulability. Algorithm 4 constructs a schedulable subset. ■

Run-time complexity. Let $K = \sum_{i=1}^n |\mathcal{P}_i|$ be the total discrete period steps. The core operation is evaluating a configuration change. The $O(n)$ incremental load calculation efficiently evaluates changes in both static and transient load metrics, reduced from $O(n^2)$. In Algorithms 2 and 3, the EF strategy evaluates up to n candidates, resulting in $O(n^2)$ complexity per iteration. Bounded by K iterations, their complexity is $O(K \cdot n^2)$. Algorithm 4 takes $O(n^2)$ time, leveraging the $O(n)$ incremental calculation across at most n steps. Thus, the overall worst-case time complexity is $O(K \cdot n^2)$.

VI. EVALUATION

This section evaluates the performance and real-time feasibility of the RT-Radar framework and quantifies the improvements in co-optimizing G1 and G2.

A. Experimental Setup

Platform and scenarios. We conducted experiments on a computer with an Intel Xeon D-2796NT CPU @ 2.00GHz and 64GB RAM. We utilized the high-fidelity PAR simulator introduced in Sec. II-B, which precisely integrates the realistic physical dynamics with the operational software stack. The base time resolution is 0.6 ms. To evaluate adaptability under dynamic workloads, we designed challenging scenarios. Dwell times C_i are uniformly distributed in [0.6, 6.0] ms. Tracking periods T_i adapt within [0.1, 1.0] s ($\Delta T_{\text{slot}} \approx 10$ ms). To guarantee safety per Lemma 2, we set the WCET bounds $\{C_{\text{under}}^{\text{OP}}, C_{\text{over}}^{\text{OP}}, C_{\text{severe}}^{\text{OP}}\} = \{45, 15, 5\}$ ms, conservatively covering the maximum observed overheads, as shown in Table I. Mission priorities ($w_i \in [1, 10]$) are assigned based on maximum acceleration capability, reflecting criticality. To stress-test adaptation, we employed a *Jerk-limited model* [45] based on live-target testing data to generate realistic high-g trajectories: aircraft (5 g–10 g) and ballistic missiles (20 g–50 g). To assess scalability and performance under varying workloads, we

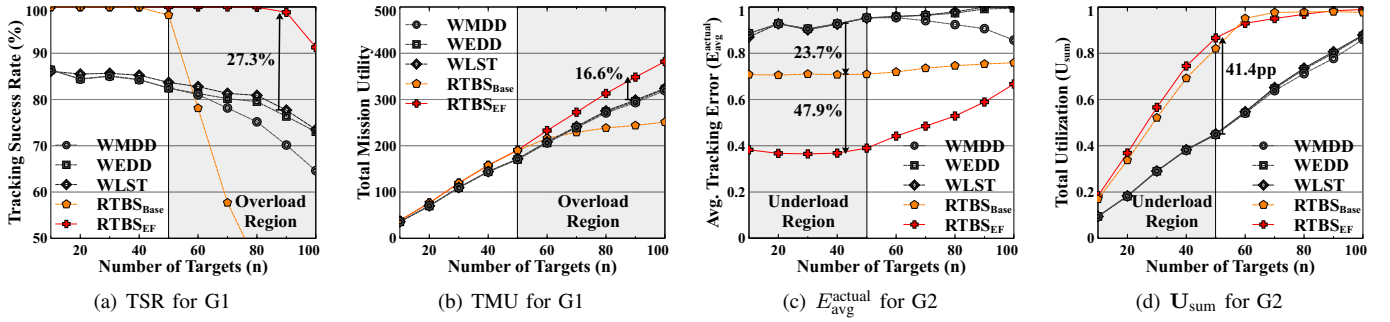


Fig. 5: Performance comparison: tracking capacity (G1) and accuracy (G2) under varying workloads.

varied the number of targets n from 10 to 100. Each data point is the average of 100 independent Monte Carlo simulations.

Methodology and comparisons. For G1, we evaluate the *tracking success ratio* (TSR) and *total mission utility* (TMU), defined as the sum of w_i of successful tracks. For G2, we evaluate the normalized system-wide average *actual* tracking error E_{avg}^{actual} and the total utilization $U_{sum} = \sum_{i=1}^n \frac{C_i}{T_i} \cdot E_{avg}^{actual}$ aligns with the optimization objective in Def. 5 and represents the measured performance, distinct from the predicted error used for scheduling. We compare the following approaches to isolate the contributions of PAM and RTBS:

- Conventional RRM algorithms using pre-defined static periods [6]–[10]: WMDD, WEDD, and WLST.
- A dynamic baseline implementing only PAM without the RTBS’s EF strategy: RTBS_{base}.
- The complete RT-Radar framework, integrating PAM with the RTBS’s EF strategy: RTBS_{EF}.

We also conduct an ablation study comparing the asymmetric control policy against a symmetric policy that ignores I2.

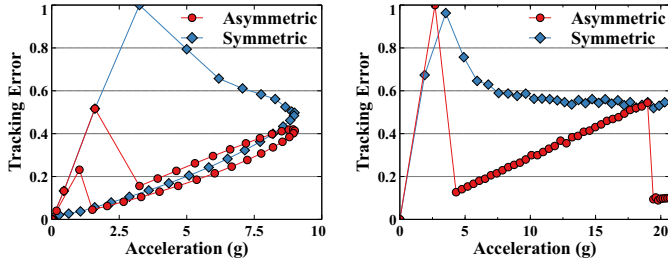
B. Experimental Results

Tracking capacity (G1). Figs. 5(a) and (b) demonstrate the substantial improvements in TSR and TMU achieved through the synergy of maneuver-aware adaptation and the EF optimization strategy, particularly during overload and severe-overload conditions. Conventional baselines degrade rapidly as n increases. RTBS_{base} initially outperforms these baselines by adapting utilization to real-time maneuverability. However, its performance collapses during severe-overload ($n \geq 70$) due to the lack of global coordination, dropping to 33.2% TSR at $n=100$. In contrast, RTBS_{EF} consistently delivers superior performance across all load conditions. At $n=90$, RTBS_{EF} achieves a 98.9% TSR, representing a significant 27.3% higher success ratio compared to the best baseline, WLST, which achieved 77.7%. This optimized capacity directly translates to higher mission utility; at $n=90$, RTBS_{EF} achieves a TMU of 348.4, which is 16.6% higher than the best baseline, WLST (298.8), and 42.8% higher than RTBS_{base} (243.9). This divergence highlights the critical role of the RTBS optimization. By efficiently redistributing the beam-time budget (Algorithm 3) and ensuring predictable degradation (Algorithm 4), RTBS_{EF} unlocks the full potential of dynamic adaptation. Furthermore, this optimization operates safely near the analytical schedulability boundary established by the rigorous run-time analysis.

Since exact NP-EDF evaluation is computationally prohibitive for online execution, a sufficient test is the only viable path to HRT guarantees. Despite this inherent conservativeness, RTBS_{EF} effectively compensates for it via substantial capacity gains. Consequently, the reported TSR represents a guaranteed lower bound, confirming a broader true schedulability region.

Tracking accuracy (G2). Crucially, RT-Radar achieves this higher capacity while simultaneously delivering superior tracking accuracy, as shown in Fig. 5(c). RTBS_{base} significantly reduces error compared to conventional baselines by dynamically ensuring minimum QoS requirements are met. At $n=40$, RTBS_{base} achieved an error of 0.707, representing a 23.7% reduction in E_{avg}^{actual} compared to the best baseline WLST, which had an error of 0.926. RTBS_{EF} dramatically amplifies these gains, reducing E_{avg}^{actual} to 0.368. This constitutes a significant 60.3% reduction compared to WLST, and a further 47.9% reduction over RTBS_{base}. This substantial improvement in accuracy is directly attributable to RTBS_{EF}’s aggressive utilization of the available beam-time budget to shorten tracking periods, as illustrated in Fig. 5(d). During underload at $n=50$, RTBS_{EF} achieves a total utilization of 86.6%, utilizing 41.4 percentage points more beam-time than WLST (45.2%). This demonstrates the framework’s effectiveness in translating utilized slack into more frequent updates, thereby enhancing tracking fidelity. It is critical to recognize that aggressively maximizing utilization under NP-EDF is inherently unsafe without formal guarantees due to non-monotonic behavior. This co-optimization is enabled by the RTBS’s sophisticated handling of non-preemptive complexities. The accuracy gains stem specifically from how the EF strategy manages the non-monotonic nature of the NP-EDF load. Guided by system bottleneck load analysis, the anomaly-aware efficiency metrics drive the optimization effectively across all workload conditions.

Effectiveness of asymmetric control. We conducted an ablation study to isolate the impact of the asymmetric control policy in handling hysteresis. As shown in Fig. 6, the symmetric policy results in severe tracking degradation during the maneuver onset phase (acceleration increase; upper path). Due to the tracking filter’s convergence latency (Sec. IV-B), the symmetric policy reacts slowly, gradually reducing the period. This delay causes rapid error accumulation, peaking at the maximum normalized error (1.0) at only 3.25 g for the aircraft. In contrast, the asymmetric policy employs aggressive



(a) Aircraft (full maneuver cycle) (b) Ballistic missile (re-entry phase)

Fig. 6: Ablation study: asymmetric vs. symmetric control.

stabilization, immediately switching to the shortest safe period (e.g., aircraft: 1.0s $\rightarrow$ 0.37s). This proactive response effectively stabilizes the track; at 3.25 g, the error is only 0.156—an 84.4% reduction. This immediacy also reduces adaptation frequency (1 vs. 8 reconfigurations). During the offset phase (lower path), the immediate reclamation demonstrates superior resource efficiency. For the aircraft at 1.5 g, the policy instantly jumped from 0.37s to 0.97s, reclaiming 61.8% of the beam-time, while the symmetric policy lagged at 0.4s.

Robustness to estimation errors. We analyzed the system’s sensitivity to acceleration estimation errors from the IMM filter. The transformation model inherently dampens this measurement noise, as $T \propto a^{-0.5}$. Thus, a conservative 10% overestimation of acceleration translates to merely a 4.6% reduction in the continuous period. Given $\Delta T_{\text{slot}} \approx 10$ ms, this reduction rarely alters the assigned discrete period. The worst-case impact occurs only if such noise forces a task with the maximum 6.0 ms dwell time to cross a quantization boundary (e.g., 110 ms $\rightarrow$ 100 ms). This increases the task’s utilization by $6.0/100 - 6.0/110$, adding a negligible 0.0055 to $\mathbf{L}_B(\mathbf{T})$. This mathematically bounded variation confirms RT-Radar’s robustness against stochastic measurement uncertainties.

Real-time feasibility. We validated the real-time feasibility of RTBS_{EF} to ensure strict adherence to HRT constraints. While the theoretical worst-case time complexity is $O(K \cdot n^2)$, the actual run-time exhibits highly load-dependent and self-limiting behavior. Table I details the maximum observed overhead C^{OP} , total search space K , and actual iterations K^{actual} for each optimization path across varying workloads. The results empirically demonstrate the EF strategy’s efficiency; notably, $K^{\text{actual}}_{\text{under}}$ remains significantly smaller than K_{under} , representing only 22.7% at $n = 90$, which confirms rapid convergence far below the theoretical bound. Crucially, $C^{\text{OP}}_{\text{under}}$ behaves non-monotonically, peaking at 41.27 ms at $n = 70$ and subsequently decreasing for $n \geq 80$. This peak marks a saturation point where a large task set and sufficient slack permit extensive optimization iterations. Two synergistic factors drive this counter-intuitive reduction: (i) tighter schedulability constraints that drastically prune the search space, reducing $K^{\text{actual}}_{\text{under}}$ by 43.4% at $n = 100$ relative to the peak, and (ii) a reduced active task set via MP-aware shedding. This behavior demonstrates that RTBS_{EF} naturally bounds the optimization overhead even under severe workloads, thereby explicitly verifying that task shedding is driven strictly by physical capacity

limits rather than algorithmic overhead. Furthermore, $C^{\text{OP}}_{\text{over}}$ and $C^{\text{OP}}_{\text{severe}}$ remain significantly lower, peaking at 12.06 ms and 3.13 ms, respectively, ensuring rapid restoration of feasibility. Consequently, these measurements validate the conservative safety bounds of 45, 15, and 5 ms established in Sec. VI-A, confirming that RT-Radar guarantees rigorous run-time safety.

VII. RELATED WORK

The prevailing RRM paradigm, spanning optimization models [1]–[5], heuristics [6]–[15], and analytical schemes [16]–[19], treats tracking periods as static inputs or pre-defined bounds. By fixing periods, these approaches preclude direct run-time optimization of the fundamental capacity/accuracy trade-off. Recent adaptive policies exhibit limitations. Rule-driven approaches [6], [14], [46] are coarse-grained, failing to capture fine-grained, asymmetric maneuver dynamics. Learning-based schedulers [47]–[49] lack the predictability for formal HRT guarantees. Furthermore, even non-preemptive admission control [16] relies on static, worst-case reservations. Crucially, existing RRM literature has not simultaneously addressed both online period optimization based on real-time maneuverability and formal verification of the transient safety during the resulting continuous reconfigurations.

This dual challenge, addressed by RT-Radar, exposes a fundamental gap in the real-time systems literature. Existing research fails to address the unique constraints inherent to STR systems: non-preemptivity, continuous reconfigurations, and immediate HRT adaptation. Elastic scheduling and exact EDF analysis [50]–[55] focus on online admission control or utilization balancing; they do not verify transient safety during continuous reconfiguration under NP-EDF. MCPs [20]–[23], designed for discrete mode transitions, introduce inherent coordination delays that are incompatible with the immediate adaptation. Consequently, a critical gap remains in verifying the safety of immediate, zero-delay reconfiguration under non-preemptive scheduling, a challenge rooted in the NP-hardness of the underlying optimization problem [32].

VIII. CONCLUSION

This paper addresses the fundamental challenge of co-optimizing tracking capacity and accuracy in STR systems. We presented RT-Radar, a novel real-time beam scheduling framework that achieves maneuver-aware period adaptation while providing formal HRT guarantees under non-preemptive constraints. RT-Radar integrates a closed-loop CPS design featuring: (i) an asymmetric control policy that proactively manages hysteretic tracking dynamics; (ii) a transient schedulability analysis for NP-EDF that formally guarantees the safety of zero-delay reconfiguration; and (iii) an anomaly-aware optimization strategy, guided by bottleneck load analysis, to navigate the non-monotonic schedulability space. Evaluation on an industrial-grade high-fidelity PAR simulator demonstrated that RT-Radar significantly outperforms established RRM algorithms. Future work includes extending this framework to multi-function radar systems and exploring joint optimization of beam scheduling and waveform adaptation.

ACKNOWLEDGMENT

This work was supported by a grant-in-aid of HANWHA SYSTEMS (C-25-008).

REFERENCES

- [1] P. Baptiste and R. Sadykov, "Time-indexed formulations for scheduling chains on a single machine: An application to airborne radars," *European Journal of Operational Research*, vol. 203, no. 2, pp. 476–483, 2010.
- [2] M. R. Taner, O. E. Karasan, and E. Yavuzturk, "Scheduling beams with different priorities on a military surveillance radar," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 48, no. 2, pp. 1725–1739, 2012.
- [3] C.-G. Lee, "A novel framework for quality-aware resource management in phased array radar systems," in *Proceedings of IEEE Real-Time Technology and Applications Symposium (RTAS)*, 2005, pp. 322–331.
- [4] S. Gopalakrishnan, C.-S. Shih, P. Ganti, M. Caccamo, L. Sha, and C. Lee, "Radar dwell scheduling with temporal distance and energy constraints," in *Proceedings of IEEE International Conference on Radar (RADAR)*, 2004.
- [5] T. Cheng, L. Liu, Z. Li, and S. Heng, "Real-time dwell scheduling based on a unified pulse interleaving framework for phased array radar," *Tsinghua Science and Technology*, vol. 29, no. 5, pp. 1540–1553, 2024.
- [6] N. H. Jeong, M. Kim, J. H. Choi, and K. T. Kim, "Beam scheduling of maritime multifunctional radar based on binary integration," *IEEE Access*, vol. 11, pp. 35 796–35 807, 2023.
- [7] S. Miranda, C. Baker, K. Woodbridge, and H. Griffiths, "Comparison of scheduling algorithms for multifunction radar," *IET Radar, Sonar Navigation*, vol. 1, no. 6, pp. 414–424, 2007.
- [8] B. Li, L. Tian, D. Chen, and Y. Han, "A task scheduling algorithm for phased-array radar based on dynamic three-way decision," *Sensors*, vol. 20, no. 1, p. 153, 2020.
- [9] A. J. Orman, C. N. Potts, A. K. Shahani, and A. R. Moore, "Scheduling for a multifunction phased array radar system," *European Journal of Operational Research*, vol. 90, no. 1, pp. 13–25, 1996.
- [10] S. H. Park, J. H. Kim, and G. Fox, "Effective real-time scheduling algorithm for cyber physical systems society," *Future Generation Computer Systems*, vol. 32, pp. 253–259, 2014.
- [11] Z. Li, T. Cheng, and S. Heng, "Real-time dwell scheduling algorithm for phased array radar based on a backtracking strategy," *IET Radar, Sonar Navigation*, vol. 17, no. 2, pp. 261–276, 2023.
- [12] Z. Ding, P. Przulj, Z. Qu, and P. Moo, "Radar task scheduling with gaussian random shifted start time," in *Proceedings of IEEE National Conference on Radar (RadarConf)*, 2024, pp. 1–5.
- [13] S. Gopalakrishnan, M. Caccamo, C.-S. Shih, C.-G. Lee, and L. Sha, "Finite-horizon scheduling of radar dwells with online template construction," in *Proceedings of IEEE Real-Time Systems Symposium (RTSS)*, 2004, pp. 23–33.
- [14] L. Bo, T. Linyu, C. Daqing, and L. Shiyang, "An adaptive dwell time scheduling model for phased array radar based on three-way decision," *Journal of Systems Engineering and Electronics*, vol. 31, no. 3, pp. 500–509, 2020.
- [15] Z. Boyan, L. Shaohong, Y. Wei, and J. Liliang, "An efficient scheduling method for phased array radars with limited time resources," in *IET International Radar Conference*, 2009, p. 454.
- [16] T. W. Kuo, Y. S. Chao, C. F. Kuo, and C. Chang, "Real-time dwell scheduling of component-oriented phased array radars," *IEEE Transactions on Computers*, vol. 54, no. 1, pp. 47–60, 2005.
- [17] C.-G. Lee, P.-S. Kang, C.-S. Shih, and L. Sha, "Radar dwell scheduling considering physical characteristics of phased array antenna," in *Proceedings of IEEE Real-Time Systems Symposium (RTSS)*, 2003, pp. 14–24.
- [18] C.-S. Shih, P. Ganti, S. Gopalakrishnan, M. Caccamo, and L. Sha, "Synthesizing task periods for dwells in multi-function phased array radars," in *Proceedings of IEEE National Conference on Radar (RadarConf)*, 2004, pp. 145–150.
- [19] C.-G. Lee, P.-S. Kang, C.-S. Shih, and L. Sha, "Schedulability envelope for real-time radar dwell scheduling," *IEEE Transactions on Computers*, vol. 55, no. 12, pp. 1599–1613, 2006.
- [20] M. Ahmed, N. Fisher, and D. Grosu, "A parallel algorithm for edf-schedulability analysis of multi-modal real-time systems," in *Proceedings of IEEE International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA)*, 2012, pp. 154–163.
- [21] A. Azim and S. Fischmeister, "Efficient mode changes in multi-mode systems," in *Proceedings of IEEE International Conference on Computer Design (ICCD)*, 2016, pp. 592–599.
- [22] T. Chen and L. T. X. Phan, "Safemc: A system for the design and evaluation of mode-change protocols," in *Proceedings of IEEE Real-Time Technology and Applications Symposium (RTAS)*, 2018, pp. 105–116.
- [23] R. Gifford, F. Galarza-Jimenez, L. T. Xuan Phan, and M. Zamani, "Decntr: Optimizing safety and schedulability with multi-mode control and resource allocation co-design," in *Proceedings of IEEE Real-Time Technology and Applications Symposium (RTAS)*, 2024, pp. 306–319.
- [24] J. M. Christiansen, K. E. Olsen, and G. E. Smith, "Fully adaptive radar for track update-interval control," in *Proceedings of IEEE National Conference on Radar (RadarConf)*, 2018, pp. 400–404.
- [25] R. Oechslein, S. Wieland, A. Zutter, U. Aulenbacher, and P. Wellig, "Fully adaptive resource management in radar networks," in *Proceedings of IEEE National Conference on Radar (RadarConf)*, 2020, pp. 1–6.
- [26] S. Björklund and T. Sjögren, "Cognitive radar tracking with adaptation of update interval and integration time," in *Proceedings of IEEE European Radar Conference (EURAD)*, 2022, pp. 233–236.
- [27] J. Yan, J. Dai, W. Pu, H. Liu, and M. Greco, "Target capacity based resource optimization for multiple target tracking in radar network," *IEEE Transactions on Signal Processing*, vol. 69, pp. 2410–2421, 2021.
- [28] W. W. Howard, A. F. Martone, and R. M. Buehrer, "Timely target tracking: Distributed updating in cognitive radar networks," *IEEE Transactions on Radar Systems*, vol. 2, no. 1, pp. 318–332, 2024.
- [29] C. E. Thornton, M. A. Kozy, R. M. Buehrer, A. F. Martone, and K. D. Sherbondy, "Deep reinforcement learning control for radar detection and tracking in congested spectral environments," *IEEE Transactions on Cognitive Communications and Networking*, vol. 6, no. 4, pp. 1335–1349, 2020.
- [30] P. Pulkkinen, T. Aittomäki, A. Ström, and V. Koivunen, "Time budget management in multifunction radars using reinforcement learning," in *Proceedings of IEEE National Conference on Radar (RadarConf)*, 2018, pp. 1–6.
- [31] Z. Lu, S. Kalia, M. C. Gursoy, C. K. Mohan, and P. K. Varshney, "Multi-objective reinforcement learning for cognitive radar resource management," in *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2025, pp. 1–5.
- [32] K. Jeffay, D. Stanat, and C. Martel, "On non-preemptive scheduling of periodic and sporadic tasks," in *Proceedings of IEEE Real-Time Systems Symposium (RTSS)*, 1991, pp. 129–139.
- [33] M. Peter and Z. Ding, *Adaptive radar resource management*. Academic Press, 2015.
- [34] G. Watson and W. Blair, "Tracking performance of a phased array radar with revisit time controlled using the imm algorithm," in *Proceedings of IEEE National Conference on Radar (RadarConf)*, 1994, pp. 160–165.
- [35] M. K. H. Benoudnine, A. Ouamri, and M. S. Woolfson, "Fast adaptive update rate for phased array radar using imm target tracking algorithm," in *IEEE International Symposium on Signal Processing and Information Technology (ISSPIT)*, 2006, pp. 277–282.
- [36] S. L., Coetzee, K. Woodbridge, and C. J. Baker, "Multifunction radar resource management using tracking optimisation," in *Proceedings of IEEE International Conference on Radar (RADAR)*, 2003, pp. 578–583.
- [37] C. Ting, H. Zi-shu, and T. Ting, "An imm-based adaptive-update-rate target tracking algorithm for phased-array radar," in *IEEE International Symposium on Intelligent Signal Processing and Communication Systems (ISPACS)*, 2007, pp. 854–857.
- [38] S. H. Baek, H. Seok, K. H. Park, and J. Chun, "An adaptive update-rate control of a phased array radar for efficient usage of tracking tasks," in *Proceedings of IEEE National Conference on Radar (RadarConf)*, 2010, pp. 1214–1219.
- [39] Y. Bar-Shalom, X. R. Li, and T. Kirubarajan, *Estimation with applications to tracking and navigation: theory algorithms and software*. John Wiley & Sons, 2001.
- [40] M. Efe and D. P. Atherton, "Maneuvering target tracking with an adaptive kalman filter," in *Proceedings of IEEE Conference on Decision and Control*, 1998, pp. 737–742.
- [41] F. Xiao, M. Song, X. Guo, and F. Ge, "Adaptive kalman filtering for target tracking," in *Proceedings of IEEE/OES China Ocean Acoustics (COA)*, 2016, pp. 1–5.
- [42] J. Wang, X. Wang, Y. Chen, M. Yan, and H. Lan, "Model adaptive kalman filter for maneuvering target tracking based on variational inference," *Electronics*, vol. 14, no. 10, p. 1908, 2025.

- [43] T. P. Baker, "Stack-based scheduling of realtime processes," *Real-Time Systems*, vol. 3, pp. 67–99, 1991.
- [44] H. Kellerer, U. Pferschy, and D. Pisinger, *Knapsack Problems*. Springer, 2004.
- [45] K. Mehrotra and P. R. Mahaparra, "A jerk model for tracking highly maneuvering targets," *IEEE transactions on aerospace and electronic systems*, vol. 33, no. 4, pp. 1094–1105, 1997.
- [46] A. Aubry, A. D. Maio, and L. Pallotta, "A priority-based scheduling scheme for search, track, and communications in mpars," *IEEE Transactions on Radar Systems*, vol. 2, pp. 471–481, 2024.
- [47] Z. Qu, Z. Ding, and P. Moo, "A machine learning radar scheduling method based on the est algorithm," in *Proceedings of IEEE International Conference on Cognitive Informatics Cognitive Computing (ICCI*CC)*, 2019, pp. 22–27.
- [48] S. Akbar, R. S. Adve, Z. Ding, and P. W. Moo, "Model-based drl for task scheduling in dynamic environments for cognitive multifunction radar," in *Proceedings of IEEE National Conference on Radar (RadarConf)*, 2024, pp. 1–6.
- [49] A. Hebb, R. S. Adve, Z. Qu, and Z. Ding, "Belief-rewards to learn radar scheduling policies," in *Proceedings of IEEE National Conference on Radar (RadarConf)*, 2024, pp. 1–6.
- [50] G. Buttazzo, G. Lipari, M. Caccamo, and L. Abeni, "Elastic scheduling for flexible workload management," *IEEE Transactions on Computers*, vol. 51, no. 3, pp. 289–302, 2002.
- [51] G. Qian, "An earlier time for inserting and/or accelerating tasks," *Real-Time Systems*, vol. 41, pp. 181–194, 2009.
- [52] F. Zhang and A. Burns, "Schedulability analysis for real-time systems with edf scheduling," *IEEE Transactions on Computers*, vol. 58, no. 9, pp. 1250–1258, 2009.
- [53] I. Ripoll and R. Ballester-Ripoll, "Period selection for minimal hyperperiod in periodic task systems," *IEEE Transactions on Computers*, vol. 62, no. 9, pp. 1813–1822, 2013.
- [54] M. Nasri and B. B. Brandenburg, "An exact and sustainable analysis of non-preemptive scheduling," in *Proceedings of IEEE Real-Time Systems Symposium (RTSS)*, 2017, pp. 12–23.
- [55] J. Orr, J. C. Uribe, C. Gill, S. Baruah, K. Agrawal, S. Dyke, A. Prakash, I. Bate, C. Wong, and S. Adhikari, "Elastic scheduling of parallel real-time tasks with discrete utilizations," in *Proceedings of international conference on real-time networks and systems (RTNS)*, 2020, pp. 117–127.