

2026

Quantitative temporal logic for safe and robust planning, learning, and control

<https://hdl.handle.net/2144/53032>

"Downloaded from OpenBU. Boston University's institutional repository."

BOSTON UNIVERSITY
COLLEGE OF ENGINEERING

Dissertation

**QUANTITATIVE TEMPORAL LOGIC FOR SAFE AND ROBUST
PLANNING, LEARNING, AND CONTROL**

by

AHMAD G. AHMAD

B.S., Palestine Polytechnic University, 2018

M.S., Boston University, 2021

Submitted in partial fulfillment of the

requirements for the degree of

Doctor of Philosophy

2026

© 2026 by
AHMAD G. AHMAD
All rights reserved

Approved by

First Reader

Calin Belta, PhD
Brendan Iribe Endowed Professor of Electrical and Computer Engineering and Computer Science
University of Maryland, College Park

Second Reader

Roberto Tron, PhD
Associate Professor of Mechanical Engineering
Associate Professor of Systems Engineering

Third Reader

Wenchao Li, PhD
Associate Professor of Electrical and Computer Engineering
Associate Professor of Systems Engineering

Fourth Reader

Alyssa Pierson, PhD
Assistant Professor of Mechanical Engineering
Assistant Professor of Systems Engineering

Fifth Reader

Gioele Zardini, PhD
Rudge (1948) and Nancy Allen Career Development Professor
Assistant Professor of Civil and Environmental Engineering
Massachusetts Institute of Technology

Logic is the art of making correct discriminations... it is a system of rules that direct reason towards correct thinking, preventing mistakes where they might occur

— Abu Nasr Muhammad Al-Farabi

Acknowledgments

I want to express my deepest gratitude to my advisors, Prof. Calin Belta and Prof. Roberto Tron, for their unwavering support, patience, and guidance throughout my doctoral journey. Their insights and mentorship have been instrumental in shaping this dissertation and my development as a researcher. I extend my sincere thanks to my dissertation committee for their thoughtful feedback and guidance.

I am grateful to the Boston University Robotics Lab community for providing an inspiring research environment. Special thanks to Kasra Ghasemi, Suhail Alsalehi, Abdelrahman Abdelgawad, and Bassel Mabsout for their friendship and collaboration. I would also like to thank my collaborators Kevin Leahy, Zachary Serlin, Ho Chit Siu, Makai Mann, and Cristian Vasile for their insightful contributions to this work. A heartfelt thanks to Prof. Karim Tahboub, who introduced me to robotics and control theory during my Bachelor's degree, sparking the research interests that led me to this path.

Finally, I owe profound gratitude to my family. To my parents, Hanan and Ghandi, for their unconditional love and support. To my siblings, Riham, Tasneem, Danyiah, Mohammed, and Omar, for their constant encouragement. And most importantly, to my beloved wife, Dania, whose patience and unwavering support have been my anchor through this doctoral journey. Thank you for always believing in me.

Ahmad G. Ahmad

PhD Candidate

Division of Systems Engineering

QUANTITATIVE TEMPORAL LOGIC FOR SAFE AND ROBUST PLANNING, LEARNING, AND CONTROL

AHMAD G. AHMAD

Boston University, College of Engineering, 2026

Major Professors: Calin Belta, PhD

Brendan Iribe Endowed Professor of Electrical and
Computer Engineering and Computer Science
University of Maryland, College Park

Roberto Tron, PhD

Associate Professor of Mechanical Engineering
Associate Professor of Systems Engineering

ABSTRACT

Temporal logics enable formal specification of robotic tasks with explicit timing constraints. Signal Temporal Logic (STL) operates over continuous-time signals with real-valued predicates and has established quantitative robustness measures, while Time Window Temporal Logic (TWTL) expresses sequential tasks with bounded time horizons through explicit time windows but previously lacked quantitative semantics. Traditional min-max robustness for STL focuses on critical time points, creating non-smooth optimization landscapes. This dissertation introduces the first quantitative semantics for TWTL, developing both traditional min-max and Arithmetic-Geometric Mean (AGM) robustness. AGM evaluates specification satisfaction holistically across all time points while maintaining soundness guarantees.

We make three main contributions. First, we develop AGM robustness semantics for TWTL using arithmetic means for mixed-sign values and geometric means for uniform-

sign values. We prove soundness, introduce interval semantics for partial trajectories, and present efficient incremental monitoring algorithms. Similarly, for STL AGM robustness, we also introduce interval semantics for partial trajectories to monitor the robustness value.

Second, we present RRT^η, integrating AGM robustness into sampling-based motion planning for STL specifications. We introduce interval semantics for trajectory prefixes, Direction of Increasing AGM Satisfaction (DIAS) vectors for gradient-like guidance, and Fulfillment Priority Logic (FPL) for principled multi-objective composition. We prove probabilistic completeness and asymptotic optimality of the planning algorithm. Experiments demonstrate AGM-based methods discover feasible solutions where traditional robustness approaches fail, with computational advantages through FPL-based composition.

Third, we develop Accelerated Proximal Policy Optimization (APPO), combining a hybrid policy architecture with TWTL-based reward shaping for reinforcement learning under delayed rewards. We introduce temporal logic progress measures for credit assignment and prove monotonic improvement with bounded optimality gap $(2\zeta\gamma\alpha^2)/(1-\gamma)^2$, where α is the mixing parameter, γ is the discount factor, and ζ bounds the expected advantage, and optimal policy preservation under general value function approximation. Experiments demonstrate successful learning for tasks with sparse, delayed rewards where standard RL approaches struggle, including game-playing and sequential manipulation tasks.

AGM robustness provides a unifying framework across temporal logic formalisms and application domains, with smooth optimization landscapes, formal guarantees, and demonstrated benefits on diverse robotic systems.

Contents

1	Introduction	1
1.1	Motivation and Context	1
1.2	Broader Impact and Vision	2
1.3	Central Thesis and Contributions	3
1.3.1	Quantitative Semantics and Monitoring Algorithms	4
1.3.2	Sampling-Based Motion Planning with STL AGM Robustness	5
1.3.3	Reinforcement Learning with Delayed Rewards	6
1.3.4	Unified Framework Across Domains	7
1.4	Research Context and Related Work	7
1.5	Dissertation Organization	9
1.6	Notation and Conventions	10
1.7	Scope and Limitations	11
2	Background and Preliminaries	12
2.1	Signal Temporal Logic	12
2.1.1	Syntax and Semantics	13
2.2	Time Window Temporal Logic	14
2.2.1	Syntax and Semantics	14
2.2.2	Relationship between STL and TWTL	16
2.3	Traditional Min-Max Robustness for STL	16
2.3.1	Definition and Properties	17
2.3.2	Advantages and Limitations	17

2.4	Arithmetic-Geometric Mean Robustness for STL	18
2.4.1	AGM Operators	18
2.4.2	AGM Robustness Definition	19
2.4.3	Advantages of AGM Robustness	20
2.5	Dynamical Systems	21
3	Quantitative Semantics and Runtime Monitoring	23
3.1	Background and Related Work	23
3.1.1	Temporal Logics for Dynamical Systems	23
3.1.2	Robustness Measures for STL	25
3.1.3	Time Window Temporal Logic	25
3.1.4	Runtime Monitoring	26
3.1.5	Contributions of This Work	26
3.2	TWTL Traditional Robustness	26
3.2.1	Definition and Soundness	27
3.2.2	Limitations	29
3.3	TWTL Arithmetic-Geometric Mean Robustness	30
3.3.1	Definition and Soundness	30
3.3.2	Computational Complexity	32
3.4	STL AGM Robustness Interval Semantics	32
3.4.1	Preliminaries	33
3.4.2	Incremental Modification Functions	33
3.4.3	Incremental Runtime Monitoring Algorithm	34
3.4.4	Theoretical Properties	35
3.4.5	Complexity Analysis	36
3.5	TWTL Robustness Interval Semantics	37
3.5.1	Interval Semantics for Traditional TWTL Robustness	37

3.5.2	Interval Semantics for TWTL AGM Robustness	38
3.5.3	Soundness and Convergence	39
3.6	Case Study: Planar Robot Navigation	40
3.7	Summary	41
4	RRTⁿ: Sampling-Based Motion Planning with AGM Robustness	44
4.1	Introduction	44
4.2	Background and Related Work	45
4.2.1	Sampling-Based Motion Planning	45
4.2.2	Temporal Logic Specifications in Motion Planning	46
4.2.3	AGM Robustness for STL	47
4.2.4	Multi-Objective Composition	48
4.2.5	Contributions of This Work	48
4.3	Problem Formulation	50
4.4	Direction of Increasing AGM Satisfaction	51
4.4.1	Base Cases and Temporal Operators	51
4.4.2	Boolean Operators and Composition Challenge	52
4.5	Composition Strategies for Multi-Objective Balancing	54
4.5.1	Stochastic Composition Baseline	54
4.5.2	FPL-Based Principled Composition	55
4.6	Steering Functions	57
4.6.1	DIAS-Guided Steering	57
4.6.2	Optimization Considerations	58
4.7	RRT ⁿ Algorithm	59
4.7.1	Tree Structure and Vertex Attributes	59
4.7.2	Main Algorithm	59
4.7.3	Update Procedure with AGM Robustness	60

4.8	Theoretical Analysis	61
4.8.1	Probabilistic Completeness	61
4.8.2	Asymptotic Optimality	62
4.9	Case Studies	63
4.9.1	Unicycle Mobile Robot	63
4.9.2	7-DOF KUKA iiwa Manipulator	66
4.10	Summary	70
5	Accelerated Proximal Policy Optimization with TWTL Reward Shaping	76
5.1	Introduction	76
5.2	Background and Related Work	77
5.2.1	Policy Gradient Methods	77
5.2.2	Temporal Logics in Reinforcement Learning	78
5.2.3	Offline Reinforcement Learning	79
5.2.4	Reward Shaping with TWTL Semantics	80
5.2.5	Trajectory Prediction for Reward Shaping	81
5.2.6	Contributions of This Work	82
5.3	Problem Formulation	83
5.3.1	Value Functions and Advantages	83
5.3.2	Concrete-Time Rewards from TWTL Specifications	84
5.4	Background: Proximal Policy Optimization	85
5.5	Hybrid Policy Architecture	86
5.5.1	Policy Mixing Architecture	86
5.5.2	Performance Improvement Guarantees	88
5.5.3	Iterative Improvement Guarantees	90
5.6	TWTL-Based Reward Shaping	91
5.6.1	Trajectory Prediction for Robustness Evaluation	92

5.6.2	Potential-Based Reward Shaping	93
5.6.3	Integration with Hybrid Architecture	94
5.7	Algorithm: Accelerated PPO	95
5.7.1	Clipped Objective with Hybrid Policy	95
5.7.2	Main Algorithm	96
5.7.3	Implementation Details	96
5.8	Experimental Validation	97
5.9	Conclusion and Future Work	98
6	Conclusion and Future Directions	100
6.1	Summary of Contributions	100
6.1.1	Quantitative Semantics and Monitoring	100
6.1.2	Sampling-Based Motion Planning	101
6.1.3	Reinforcement Learning with Delayed Rewards	103
6.1.4	Unifying Framework	104
6.2	Limitations and Assumptions	104
6.2.1	Discrete-Time and Finite Horizons	104
6.2.2	Differentiability Requirements	105
6.2.3	Trajectory Prediction Quality	105
6.2.4	Computational Complexity	106
6.2.5	Specification Design	106
6.3	Future Research Directions	107
6.3.1	Probabilistic and Stochastic Specifications	107
6.3.2	Dynamic Obstacles and Time-Varying Environments	108
6.3.3	Integration with Neural Network Controllers	108
6.3.4	Computational Efficiency and Approximation	109
6.3.5	Real-World Validation and Deployment	110

6.4	Broader Implications	111
6.4.1	Bridging Theory and Practice	111
6.4.2	The Role of Quantitative Semantics	111
6.4.3	Specification as Communication	112
6.5	Closing Remarks	112
A	Quantitative Semantics Proofs	114
A.1	Correctness of Incremental Modification Functions	114
A.2	STL AGM Robustness Interval Soundness	116
A.3	STL AGM Robustness Interval Chain Inclusion	117
	References	119
	Curriculum Vitae	125

List of Figures

- 3·1 Demonstration of TWTL robustness and monitoring. (top-left) Depiction of the valuation of words \mathbf{o}_1 , \mathbf{o}_2 , and \mathbf{o}_3 w.r.t. time. The evolution of the min-max robustness (dashed-line with triangles) and the AGM robustness (solid-lines with circles) for words \mathbf{o}_1 , \mathbf{o}_2 (bottom-left), and \mathbf{o}_3 are shown in the top-right, bottom-left, and bottom-right figures, respectively. 29
- 3·2 Demonstration of monitoring the TWTL robustness and AGM TWTL robustness of precomputed planar robot runs, \mathbf{o}_1 and \mathbf{o}_1 , against satisfying formula (3.41). Words \mathbf{o}_1 and \mathbf{o}_2 in an xy – planar environment are shown in the blue and green traces in the left figure, respectively. The valuations of the intervals $[\rho]$ and $[\eta]$ for \mathbf{o}_1 , and \mathbf{o}_2 at every $t \in \{2, 10, 15, \dots, 40, 42\}$ are depicted in the middle, and right figures, respectively, where $[\rho]$ is depicted in dashed-lines with triangles and the $[\eta]$ is depicted in solid-lines with circles. 41

4.1	<p>Illustration of the Direction of Increasing Satisfaction (DIAS) for a spatial predicate $\mu = \{q \mid \ q - c\ ^2 \leq r^2\}$ with center $c = [3.5, 3.5]^\top$ and radius $r = 1$. (a) The robustness landscape $\eta(q, \mu)$ showing the satisfaction region (red circle) where $\eta > 0$. (b) The gradient field $\nabla_q \eta(q, \mu)$ pointing toward the direction of increasing satisfaction. (c) The state transition field $\mathbf{J}_f(q) \cdot u$ resulting from the control input $u = [0.5, 0.5]^\top$ and discrete-time dynamics with Jacobian $\mathbf{J}_f = \mathbf{I}_2$. (d) The DIAS field $\chi_\eta(q, \mu) = \nabla \eta(q, \mu)^\top \cdot \mathbf{J}_f$, which is nonzero only in regions where $\nabla \eta(q, \mu)^\top \cdot (q_{k+1} - q_k) > 0$ (condition satisfied).</p>	53
4.2	<p>Performance comparison on unicycle sequential reach-avoid task. (a) Lower bounds: Traditional robustness (blue, normalized) fails with $\underline{\eta} \approx -0.35$; choose-blend (red) achieves $\underline{\eta} \approx 0.93$; FPL (green) reaches $\underline{\eta} \approx 0.95$. (b) Upper bounds show similar trends. (c) Gap metric: FPL converges to gap < 0.05 at 400 iterations, choose-blend at 800 iterations, demonstrating $2\times$ speedup. (d) Tree construction showing successful path (color indicates temporal progression) visiting Region 1, navigating around obstacle via upper arc, and terminating in Region 2.</p>	74

4.3	Performance comparison on KUKA cascading choice problem. (a) Lower bound evolution: FPL (green) shows rapid, structured convergence; traditional (blue) and choose-blend (red) show slower exploration. (b) Upper bound evolution shows similar trends. (c) Gap metric: FPL reaches near-zero gap (< 0.1) within 1000 iterations; traditional and choose-blend plateau at 1.2 and 0.8 respectively. (d-f) Ghost trail visualization from multiple viewpoints showing reference trajectory with 10 time-sampled configurations (opacity indicates temporal progression). The robot successfully executes the A-then-E path with red spheres marking joint positions throughout motion.	75
5.1	The Actor-Critic RL framework. The Actor’s architecture, policy π_θ , consists of an offline policy, π_ρ , and an adaptive policy, π_β , where the two policies are mixed using the parameters of the FCL, α . The critic consists of an MLP that approximates the value function. The task predictor LSTM network and the reward shaping are depicted in cyan.	87
5.2	Training performance comparison across different variants of PPO in LunarLander-v2 and Pendulum environments. For LunarLander-v2 (left y-axis), we compare vanilla PPO against variants with reward shaping and mixing. The results show that combining reward shaping with policy mixing achieves faster learning and better performance compared to baseline PPO and reward shaping alone. For Pendulum (right y-axis, dashed lines), we include PPO with and without mixing for reference. Training steps are shown in millions on the x-axis.	98

List of Abbreviations

AGM	Arithmetic-Geometric Mean
APPO	Accelerated Proximal Policy Optimization
CBF	Control Barrier Function
DIAS	Direction of Increasing AGM Satisfaction
DIS	Direction of Increasing Satisfaction
DOF	Degrees of Freedom
FK	Forward Kinematics
FPL	Fulfillment Priority Logic
GAE	Generalized Advantage Estimation
IK	Inverse Kinematics
IRTM	Incremental Robustness Tracking Monitor
LSTM	Long Short-Term Memory
LTL	Linear Temporal Logic
MDP	Markov Decision Process
MILP	Mixed-Integer Linear Program
MTL	Metric Temporal Logic
PPO	Proximal Policy Optimization
RL	Reinforcement Learning
RRT	Rapidly-exploring Random Tree
RRT*	RRT-star
RRT ^η	RRT-eta
STL	Signal Temporal Logic
TD	Temporal Difference
TL	Temporal Logic
TRPO	Trust Region Policy Optimization
TWTL	Time Window Temporal Logic

Chapter 1

Introduction

1.1 Motivation and Context

The increasing autonomy and complexity of robotic systems demands formal methods that can specify, verify, and synthesize correct-by-construction behaviors. From autonomous vehicles navigating urban environments to robotic manipulators performing delicate assembly tasks, modern applications require robots to satisfy complex spatiotemporal requirements with quantifiable guarantees of correctness. Traditional planning and control approaches, while effective for trajectory tracking and stabilization, often struggle to encode and reason about high-level task specifications involving temporal ordering, explicit timing constraints, and sequential objectives.

Temporal logics have emerged as powerful specification languages for cyber-physical systems, enabling the formal expression of complex behavioral requirements (Baier and Katoen, 2008). Signal Temporal Logic (STL) and its variants provide rich formalisms for specifying properties over continuous signal (Maler and Nickovic, 2004), while Time Window Temporal Logic (TWTL) offers efficient representations of sequential tasks with explicit time bound (Vasile et al., 2017a). These logics bridge the gap between high-level human-interpretable task descriptions and low-level control implementations, enabling automated synthesis of correct behaviors.

A critical challenge in leveraging temporal logic specifications for robotic systems lies in quantifying specification satisfaction beyond binary yes-no decisions. While Boolean semantics determine whether a trajectory satisfies a specification, they provide no infor-

mation about satisfaction quality, robustness to disturbances, or distance to satisfaction boundaries. This limitation becomes particularly acute in three scenarios. First, during motion planning and control synthesis, optimization algorithms require continuous objective functions to guide search toward high-quality solutions. Second, in online monitoring and runtime verification, systems must assess partial trajectories before complete specification evaluation becomes possible. Third, in reinforcement learning settings with delayed rewards, agents need immediate feedback signals that correlate with eventual specification satisfaction.

The quantitative semantics problem asks: how can we measure the degree to which a trajectory satisfies a temporal logic specification? Traditional robustness metrics based on min-max operations provide soundness guarantees—positive robustness implies Boolean satisfaction—but suffer from brittleness. They focus exclusively on the most critical time points and subformulae, ignoring the broader context of satisfaction across entire trajectories. This creates non-smooth optimization landscapes with sharp decision boundaries that hinder gradient-based methods and sampling-based planners. Moreover, when composing multiple temporal objectives, simple selection mechanisms fail to balance competing requirements adequately.

1.2 Broader Impact and Vision

The techniques developed in this dissertation aim to increase the autonomy and reliability of robotic systems through formal specifications and quantitative guarantees. Potential application domains include:

Autonomous Vehicles: Temporal logic specifications can encode traffic rules, safety requirements, and mission objectives. AGM robustness provides continuous measures for optimization of driving policies while maintaining safety guarantees.

Medical Robotics: Surgical robots and rehabilitation devices require precise spatiotem-

poral control with stringent safety requirements. Formal specifications with quantitative semantics enable verification and synthesis of high-confidence behaviors.

Manufacturing and Logistics: Robotic manipulation and warehouse automation involve complex sequential tasks with timing constraints. The frameworks developed here can enable more flexible and robust automation.

Search and Rescue: Emergency response robots must satisfy complex objectives under time pressure while adapting to uncertain environments. The combination of formal specifications and learning-based adaptation could improve performance in such scenarios.

However, increased autonomy also raises important considerations regarding safety, transparency, and accountability. Formal specifications provide interpretability—a human can understand what behavior the robot is supposed to exhibit—but they cannot guarantee that specifications correctly capture human intent in all situations. The quantitative robustness measures provide continuous feedback about satisfaction quality, but threshold selection for “sufficient” robustness remains a design choice requiring domain expertise and safety analysis.

As robotic systems become more autonomous, ensuring their behaviors align with human values and societal norms becomes increasingly critical. Formal methods and temporal logic specifications provide one tool in this broader challenge, but they must be integrated with complementary approaches including human oversight, fail-safe mechanisms, and ongoing monitoring during deployment.

1.3 Central Thesis and Contributions

This dissertation develops a unified framework for temporal logic specifications in robotics, centered on Arithmetic-Geometric Mean (AGM) robustness as a quantitative semantics that addresses the limitations of traditional approaches. We establish that AGM robustness provides a mathematically principled measure that evaluates specification satisfaction holis-

tically across all time points and subformulae, creating smoother optimization landscapes while maintaining soundness guarantees. Building on this foundation, we develop practical algorithms for runtime monitoring, sampling-based motion planning, and reinforcement learning that leverage AGM robustness to achieve superior performance in complex robotic tasks.

The central thesis of this work is:

Arithmetic-Geometric Mean robustness provides a unifying quantitative semantics for temporal logic specifications that enables efficient online monitoring, improves sampling-based motion planning through smoother exploration landscapes, and accelerates reinforcement learning in delayed reward settings through principled reward shaping.

This thesis is validated through four main contributions spanning theory, algorithms, and applications.

1.3.1 Quantitative Semantics and Monitoring Algorithms

We develop traditional min-max robustness from which we derive AGM robustness semantics for TWTL specifications (Ahmad et al., 2023), establishing their soundness through formal proofs. Unlike traditional min-max robustness that focuses on critical points, AGM robustness aggregates satisfaction information across all time points using arithmetic means when values have mixed signs and geometric means when values share signs. This creates continuous, smoother measures of satisfaction quality.

For online monitoring of partial trajectories, we introduce interval semantics that bound possible robustness values over all completions of incomplete trajectories. We develop efficient incremental monitoring algorithms that update robustness intervals as new observations arrive, with complexity linear in formula size per observation, compared to quadratic in trajectory length for non-incremental approaches. For typical planning scenarios with a

trajectory length 20 – 50 time units and formula complexity 10 – 20, this reduces monitoring overhead by one order of magnitude of operations per trajectory evaluation.

The monitoring algorithms maintain soundness: for any partial trajectory and any possible completion, the true robustness value is guaranteed to lie within the computed interval bounds. As trajectories extend, intervals shrink, converging to exact robustness values when trajectory horizon reaches formula horizon. These properties enable informed decision-making during incremental planning.

1.3.2 Sampling-Based Motion Planning with STL AGM Robustness

We present RRT^η , a sampling-based motion planning algorithm that integrates AGM robustness for STL specifications. The key insight is that AGM’s holistic evaluation across time points and subformulae provides more consistent guidance for tree exploration compared to traditional robustness’s sharp decision boundaries.

Our contributions include three components. First, we develop Direction of Increasing AGM Satisfaction (DIAS) vectors that leverage AGM’s smooth gradients to guide steering during tree expansion. These vectors naturally encode specification structure without the abrupt changes characteristic of traditional robustness gradients. Second, we introduce principled multi-objective composition using Fulfillment Priority Logic (FPL) (Mabsout et al., 2025), enabling systematic balancing of competing temporal requirements. FPL-based composition computes gradient-based weights that prioritize less-fulfilled objectives, providing $1.5 - 2\times$ computational advantages over stochastic methods. Third, we prove that RRT^η maintains probabilistic completeness and asymptotic optimality guarantees of RRT^* despite the more complex robustness measure.

Experimental validation on unicycle and 7-DOF manipulator systems demonstrates substantial advantages. On sequential reach-avoid tasks, traditional robustness-based planning fails to discover satisfying trajectories (negative lower bounds throughout planning) while AGM-based methods achieve high-quality solutions with robustness $\eta_L \approx 0.93-0.95$.

For high-dimensional systems, augmented state representation with inverse kinematics-based sampling achieves computational speedup for tight workspace constraints, reducing planning time until convergence.

1.3.3 Reinforcement Learning with Delayed Rewards

We develop Accelerated Proximal Policy Optimization (APPO) (Ahmad et al., 2025), addressing the fundamental challenge of delayed rewards where control actions leading to successful outcomes may not receive immediate feedback. APPO combines two innovations: a hybrid policy architecture and TWTL-based reward shaping.

The hybrid architecture mixes an offline policy trained on expert demonstrations with an online policy optimized through PPO, maintaining the offline policy as an active guide throughout training rather than using it merely for initialization. We prove monotonic improvement guarantees over both the offline policy and previous iterations, with bounded performance gap $(2\zeta\gamma\alpha^2)/(1-\gamma)^2$ where α is the learned mixing parameter, γ is the discount factor, and ζ bounds the expected advantage. The mixing parameter automatically adapts based on relative policy quality, providing robustness to offline demonstration quality.

For reward shaping, we use TWTL robustness with trajectory prediction to provide immediate feedback about specification satisfaction. Using potential-based shaping (Ng et al., 1999), we prove optimal policy preservation while transforming sparse delayed rewards into dense immediate signals.

Experimental validation on robotic control benchmarks demonstrates substantial improvements. On the lunar lander task with a delayed reward achieved at the end of a successful landing, APPO achieves better performance compared to vanilla PPO.

1.3.4 Unified Framework Across Domains

A key contribution of this dissertation is demonstrating AGM robustness as a unifying mathematical framework applicable across diverse robotic domains. The same core measure provides benefits in motion planning (smoother exploration), reinforcement learning (dense reward signals), and runtime monitoring (efficient incremental computation). This unification suggests AGM robustness captures fundamental properties of specification satisfaction that transcend specific application contexts.

1.4 Research Context and Related Work

This dissertation builds upon three interconnected areas of research: formal methods for robotics, sampling-based motion planning, and reinforcement learning.

Temporal logics have emerged as powerful tools for formalizing high-level robotic tasks (Baier and Katoen, 2008). Linear Temporal Logic (LTL) has been widely employed for planning and synthesis problems (Vasile et al., 2020; Belta et al., 2017; Kress-Gazit et al., 2018), enabling specification of qualitative properties such as “eventually visit region A” or “always avoid obstacle B.” To express tasks with explicit timing constraints, logics such as Signal Temporal Logic (STL) (Maler and Nickovic, 2004) and Metric Temporal Logic (MTL) (Koymans, 1990) operate over continuous-time signals with real-valued predicates. These logics admit quantitative robustness semantics that measure how strongly a trajectory satisfies a specification (Donzé and Maler, 2010; Fainekos and Pappas, 2009), enabling optimization-based synthesis approaches. Time Window Temporal Logic (TWTL) (Vasile et al., 2017a) provides an alternative concrete-time formalism that efficiently expresses sequential tasks through explicit time windows and admits efficient automata-based verification. However, unlike STL and MTL, TWTL previously lacked quantitative semantics for measuring satisfaction degree.

Quantitative semantics have enabled robustness-maximizing approaches for both mo-

tion planning and control synthesis. For STL specifications, robustness measures have been integrated into sampling-based planners (Vasile et al., 2017b), trajectory optimization (Sadraddini and Belta, 2015; Raman et al., 2014), and control barrier function synthesis (Ahmad et al., 2022; Yang et al., 2025). The traditional min-max robustness focuses on the most critical time points and constraints, creating non-smooth optimization landscapes. Recent work introduced Arithmetic-Geometric Mean (AGM) robustness for STL (Mehdipour et al., 2019), which evaluates satisfaction holistically using arithmetic and geometric means, producing smoother gradients for optimization. However, AGM robustness has not been developed for TWTL, nor has it been systematically integrated into motion planning frameworks with formal convergence guarantees.

In the context of reinforcement learning, temporal logics have been increasingly used to specify complex tasks beyond simple reward functions. Approaches range from automata-guided learning (Li et al., 2017; Icarte et al., 2022) to reward shaping from temporal logic specifications (Asarkaya et al., 2021; Balakrishnan and Deshmukh, 2019). Policy gradient methods, particularly Proximal Policy Optimization (PPO) (Schulman et al., 2017), have demonstrated strong performance across diverse domains. However, PPO can struggle in environments with delayed rewards where the temporal gap between actions and consequences creates sparse gradient signals. Offline reinforcement learning methods leverage expert demonstrations (Ball et al., 2023; Ravari et al., 2024; Hu et al., 2023), but typically treat offline data as a fixed initialization rather than an active component throughout training. Combining temporal logic specifications with offline data to accelerate learning in delayed-reward settings remains an open challenge.

This dissertation addresses these gaps by developing quantitative semantics for TWTL, integrating AGM robustness into sampling-based planning with formal guarantees, and combining temporal logic reward shaping with hybrid policy architectures for reinforcement learning. The following chapters present detailed related work for each contribution

area.

1.5 Dissertation Organization

The remainder of this dissertation is organized as follows:

Chapter 2: Background and Preliminaries establishes foundational concepts including dynamic systems, temporal logic syntax and semantics (STL and TWTL), and traditional quantitative robustness measures. We present the standard min-max robustness semantics that serve as baseline for comparison throughout the dissertation.

Chapter 3: Quantitative Semantics and Runtime Monitoring develops traditional and AGM robustness semantics for TWTL and AGM robustness monitor for STL specifications. We introduce interval semantics for reasoning about partial trajectories and present efficient incremental monitoring algorithms. Formal proofs establish soundness, chain inclusion properties, and convergence guarantees.

Chapter 4: Sampling-Based Motion Planning from STL Specifications presents the RRTⁿ framework for motion planning with AGM robustness. We develop DIAS vectors for informed tree exploration, introduce FPL-based composition for multi-objective balancing, and prove probabilistic completeness and asymptotic optimality. Case studies on unicycle, and 7-DOF manipulator systems validate the approach, demonstrating superior performance over traditional robustness-based planning.

Chapter 5: Reinforcement Learning with Delayed Rewards introduces APPO for learning under delayed rewards using TWTL specifications. We develop the hybrid policy architecture combining offline and online learning, establish monotonic improvement guarantees, and present TWTL-based reward shaping with optimality preservation proofs. Experimental validation on inverted pendulum and lunar lander environments demonstrates effectiveness for delayed reward.

Chapter 6: Conclusion and Future Directions synthesizes contributions across do-

mains, discusses limitations of current approaches, and proposes directions for future research including probabilistic specifications, and dynamic obstacles for motion planning.

1.6 Notation and Conventions

Throughout this dissertation, we adopt the following notational conventions:

Sets and Spaces: \mathcal{X} , \mathcal{U} , and \mathcal{O} denote the state space, the control space, and the observation space, respectively. We use calligraphic letters for spaces and Roman letters for elements: $x \in \mathcal{X}$, $u \in \mathcal{U}$, $o \in \mathcal{O}$.

Trajectories and Words: Bold notation denotes sequences: $\mathbf{x}_{t_1, t_2} = x_{t_1} x_{t_1+1} \cdots x_{t_2}$ is a state trajectory, $\mathbf{o}_{t_1, t_2} = o_{t_1} o_{t_1+1} \cdots o_{t_2}$ is an observation word. We use subscripts to denote time indices and superscripts to denote episodes or iterations.

Temporal Logic: We use ϕ (lowercase phi) for TWTL formulae and φ (lowercase varphi) for STL formulae. Greek letters μ , π denote atomic propositions. The satisfaction relation is \models , and $\|\phi\|$ denotes formula time horizon.

Robustness Measures: We use ρ (varrho) for traditional min-max robustness of TWTL, ρ (rho) for traditional min-max robustness of STL, and η (eta) for AGM robustness of both TWTL and STL. Interval semantics are denoted by brackets: $[\eta] = [\underline{\eta}, \overline{\eta}]$ represents a robustness interval with lower bound $\underline{\eta}$ and upper bound $\overline{\eta}$.

Policies and Values: In reinforcement learning contexts, π denotes a policy (function from states to action distributions), V^π denotes state value function, Q^π denotes state-action value function, and A^π denotes advantage function. We use θ for policy parameters, with π_θ denoting a parameterized policy.

Probabilities and Expectations: $\mathbb{P}(\cdot)$ denotes probability, $\mathbb{E}[\cdot]$ denotes expectation. When the distribution is clear from context, we write $\mathbb{E}_{u \sim \pi}$ to emphasize expectation over actions sampled from policy π .

Optimization: $\arg \max$, $\arg \min$ denote arguments that maximize or minimize func-

tions. We use ∇ for gradients and $\|\cdot\|$ for norms (Euclidean unless otherwise specified).

1.7 Scope and Limitations

This dissertation focuses on discrete-time systems with finite time horizons in deterministic or stochastic but model-free settings. While the temporal logic frameworks (STL, TWTL) can in principle handle continuous-time signals and infinite horizons, our algorithms and theoretical results assume discrete-time observations and finite planning/learning horizons.

For motion planning, we consider systems with Lipschitz continuous dynamics for which sampling-based methods are appropriate. We do not address hybrid systems with discrete mode switches, though extensions to such systems would be natural future work. The experimental validation focuses on 2-D mobile robots and robot arms in relatively simple environments; real-world deployment would require addressing additional challenges including perception uncertainty, environmental disturbances, and hardware constraints.

For reinforcement learning, we assume access to a simulator or environment for sample collection, and we assume availability of some offline demonstrations for the hybrid architecture (though we show graceful degradation with poor demonstration quality). We do not address purely offline RL where no environment interaction is possible, nor do we address the problem of learning from human preferences without explicit specifications.

The AGM robustness framework requires differentiable predicates for gradient-based methods, limiting applicability to specifications with non-differentiable predicates (e.g., mode switches, logical constraints). Extensions to broader classes of specifications remain open research questions.

Chapter 2

Background and Preliminaries

This chapter establishes the formal foundations for temporal logic specifications and quantitative semantics. We begin with Signal Temporal Logic (STL), a widely-used formalism for specifying properties of continuous-time signals with explicit timing constraints. We then introduce Time Window Temporal Logic (TWTL), which provides an efficient syntax for expressing sequential tasks common in robotics applications. After presenting the traditional min-max robustness semantics for STL, we introduce the Arithmetic-Geometric Mean (AGM) robustness measure, which addresses key limitations of min-max semantics by providing smoother optimization landscapes. Finally, we formalize the class of dynamical systems considered in this dissertation.

2.1 Signal Temporal Logic

Signal Temporal Logic (STL) (Maler and Nickovic, 2004) provides a formal language for specifying temporal properties of real-valued signals. Unlike Linear Temporal Logic (LTL), which operates over discrete sequences of Boolean valuations, STL can express specifications with explicit, concrete time bounds.

2.1.1 Syntax and Semantics

Consider discrete-time signals where signal valuations $s_t \in \mathbb{R}$ are sampled at discrete time points $t \in \mathbb{Z}_{\geq 0}$. An STL formula φ over signals is defined recursively as follows:

$$\varphi ::= \mu \mid \neg\varphi \mid \varphi_1 \wedge \varphi_2 \mid \varphi_1 \vee \varphi_2 \mid \mathbf{G}_{[a,b]}\varphi_1 \mid \mathbf{F}_{[a,b]}\varphi_1 \quad (2.1)$$

where μ is an atomic predicate of the form $h(s_t) > \sigma$ with $h : \mathbb{R} \rightarrow \mathbb{R}$ a linear function and $\sigma \in \mathbb{R}$ a threshold. The operators \neg , \wedge , and \vee denote negation, conjunction, and disjunction, respectively. The temporal operators $\mathbf{G}_{[a,b]}$ (Globally, or Always) and $\mathbf{F}_{[a,b]}$ (Finally, or Eventually) specify time bounds $a, b \in \mathbb{Z}_{\geq 0}$ with $a < b$. We denote the set of all STL formulae as Φ_{STL} .

A signal sequence $s_{t_1, t_2} := s_{t_1} s_{t_1+1} \cdots s_{t_2}$ represents the signal over the time interval $[t_1, t_2]$. The Boolean satisfaction relation $s_{t_1, t_2} \models \varphi$ is defined recursively as follows:

$$s_t \models \mu \quad \text{iff } h(s_t) > \sigma \quad (2.2)$$

$$s_{t_1, t_2} \models \neg\varphi \quad \text{iff } s_{t_1, t_2} \not\models \varphi \quad (2.3)$$

$$s_{t_1, t_2} \models \varphi_1 \wedge \varphi_2 \quad \text{iff } (s_{t_1, t_2} \models \varphi_1) \wedge (s_{t_1, t_2} \models \varphi_2) \quad (2.4)$$

$$s_{t_1, t_2} \models \varphi_1 \vee \varphi_2 \quad \text{iff } (s_{t_1, t_2} \models \varphi_1) \vee (s_{t_1, t_2} \models \varphi_2) \quad (2.5)$$

$$s_{t_1, t_2} \models \mathbf{G}_{[a,b]}\varphi \quad \text{iff } \forall t \in [t_1 + a, t_1 + b], s_{t_1, t} \models \varphi \quad (2.6)$$

$$s_{t_1, t_2} \models \mathbf{F}_{[a,b]}\varphi \quad \text{iff } \exists t \in [t_1 + a, t_1 + b], s_{t_1, t} \models \varphi \quad (2.7)$$

Definition 2.1.1 (STL Time Horizon). *The time horizon $\|\varphi\|$ of an STL formula φ represents the minimum time required to evaluate the formula. It is defined recursively as:*

$$\|\varphi\| := \begin{cases} 0 & \text{if } \varphi = \mu \\ \max(\|\varphi_1\|, \|\varphi_2\|) & \text{if } \varphi \in \{\varphi_1 \wedge \varphi_2, \varphi_1 \vee \varphi_2\} \\ \|\varphi_1\| & \text{if } \varphi = \neg\varphi_1 \\ b + \|\varphi_1\| & \text{if } \varphi \in \{\mathbf{G}_{[a,b]}\varphi_1, \mathbf{F}_{[a,b]}\varphi_1\} \end{cases} \quad (2.8)$$

Example 2.1.1 (Obstacle Avoidance). *Consider a robot navigating in a plane with position $(x, y) \in \mathbb{R}^2$. The signal $s_t = (x_t, y_t)$ represents the robot's position at time t . An obstacle avoidance specification requiring the robot to maintain distance $d_{safe} = 1.0$ from an obstacle centered at $(x_c, y_c) = (5, 5)$ for the entire trajectory of length $T = 20$ can be expressed as:*

$$\varphi_{avoid} := \mathbf{G}_{[0,20]}\mu_{safe} \quad (2.9)$$

where the predicate μ_{safe} is defined as:

$$h(s_t) = \sqrt{(x_t - x_c)^2 + (y_t - y_c)^2} - d_{safe} > 0 \quad (2.10)$$

The formula φ_{avoid} has time horizon $\|\varphi_{avoid}\| = 20$.

Example 2.1.2 (Sequential Reach-Avoid). *Building on Example 2.1.1, suppose we require the robot to visit a goal region $\mathcal{G} = \{(x, y) \mid (x - 10)^2 + (y - 10)^2 \leq 1\}$ within the time window $[5, 15]$ while continuously avoiding the obstacle. This can be expressed as:*

$$\varphi_{task} := \mathbf{F}_{[5,15]}\mu_{goal} \wedge \mathbf{G}_{[0,20]}\mu_{safe} \quad (2.11)$$

where μ_{goal} is defined by $h(s_t) = 1 - \sqrt{(x_t - 10)^2 + (y_t - 10)^2} > 0$. The time horizon is $\|\varphi_{task}\| = \max(15, 20) = 20$.

2.2 Time Window Temporal Logic

Time Window Temporal Logic (TWTL) (Vasile et al., 2017a) provides an alternative temporal logic formalism specifically designed for specifying sequential tasks with explicit timing constraints. TWTL offers advantages for robotics applications through its expressive syntax for sequential compositions and its efficient translation to automata.

2.2.1 Syntax and Semantics

Consider a discrete-time system with state $x \in \mathcal{X} \subset \mathbb{R}^d$ and observable output $o_t = l(x_t)$ at time t , where $l : \mathcal{X} \rightarrow 2^\Pi$ is a labeling function and Π is a set of atomic propositions. An atomic proposition $\pi_A \in \Pi$ takes Boolean value \top at output o_t if $o_t \in A$ where $A := \{o \mid h(o) > \sigma\}$ with $h : 2^\Pi \rightarrow \mathbb{R}^d$ and $\sigma \in \mathbb{R}^d$, and \perp otherwise.

A TWTL formula ϕ is defined recursively as:

$$\phi ::= \mathbf{H}^d \pi_A \mid \mathbf{H}^d \neg \pi_A \mid \phi_1 \wedge \phi_2 \mid \phi_1 \vee \phi_2 \mid \neg \phi \mid \phi_1 \cdot \phi_2 \mid [\phi]_{[a,b]} \quad (2.12)$$

where \mathbf{H}^d is the hold operator with duration $d \in \mathbb{Z}_{\geq 0}$, \cdot is the concatenation operator, and $[\cdot]_{[a,b]}$ is the within operator with time bounds $a, b \in \mathbb{Z}_{\geq 0}$ and $a \geq b$.

An output word $o_{t_1, t_2} := o_{t_1} o_{t_1 + \Delta t} \cdots o_{t_2}$ represents the sequence of outputs over the time interval $[t_1, t_2]$ with sampling time Δt . The Boolean satisfaction relation $o_{t_1, t_2} \models \phi$ is defined recursively as:

$$o_{t_1, t_2} \models \mathbf{H}^d \pi_A \quad \text{iff } (o_t \in A, \forall t \in [t_1, t_2]) \wedge (t_2 - t_1 \geq d\Delta t) \quad (2.13)$$

$$o_{t_1, t_2} \models \mathbf{H}^d \neg \pi_A \quad \text{iff } (o_t \notin A, \forall t \in [t_1, t_2]) \wedge (t_2 - t_1 \geq d\Delta t) \quad (2.14)$$

$$o_{t_1, t_2} \models \phi_1 \wedge \phi_2 \quad \text{iff } (o_{t_1, t_2} \models \phi_1) \wedge (o_{t_1, t_2} \models \phi_2) \quad (2.15)$$

$$o_{t_1, t_2} \models \phi_1 \vee \phi_2 \quad \text{iff } (o_{t_1, t_2} \models \phi_1) \vee (o_{t_1, t_2} \models \phi_2) \quad (2.16)$$

$$o_{t_1, t_2} \models \neg \phi \quad \text{iff } o_{t_1, t_2} \not\models \phi \quad (2.17)$$

$$o_{t_1, t_2} \models \phi_1 \cdot \phi_2 \quad \text{iff } \exists t = \arg \min_{t \in [t_1, t_2]} \{o_{t_1, t} \models \phi_1\} \wedge (o_{t + \Delta t, t_2} \models \phi_2) \quad (2.18)$$

$$o_{t_1, t_2} \models [\phi]_{[a,b]} \quad \text{iff } (\exists t \geq t_1 + a \text{ s.t. } o_{t, t_1 + b} \models \phi) \wedge (t_2 - t_1 \geq b) \quad (2.19)$$

Definition 2.2.1 (TWTL Time Horizon). *The time horizon $\|\phi\|$ of a TWTL formula ϕ is defined recursively as:*

$$\|\phi\| := \begin{cases} \max(\|\phi_1\|, \|\phi_2\|) & \text{if } \phi \in \{\phi_1 \wedge \phi_2, \phi_1 \vee \phi_2\} \\ \|\phi_1\| & \text{if } \phi = \neg \phi_1 \\ \|\phi_1\| + \|\phi_2\| + \Delta t & \text{if } \phi = \phi_1 \cdot \phi_2 \\ d\Delta t & \text{if } \phi = \mathbf{H}^d \pi_A \\ b & \text{if } \phi = [\phi_1]_{[a,b]} \end{cases} \quad (2.20)$$

Definition 2.2.2 (Feasible TWTL Formula (Vasile et al., 2017a)). *A TWTL formula ϕ is called feasible if each time window's deadline is sufficiently large to accommodate its enclosed task. Formally, for every within operator $[\phi']_{[a,b]}$ appearing in ϕ , the window dura-*

tion $b - a$ must be at least as large as the time horizon $\|\phi'\|$ of the enclosed subformula.

We denote the set of all feasible TWTL formulae as Φ_{TWTL} .

Example 2.2.1 (Lunar Lander Sequential Task). *Consider the lunar lander environment with state $x = (p_x, p_y, \dot{p}_x, \dot{p}_y, \psi, \dot{\psi}) \in \mathbb{R}^6$ representing position, velocity, angle, and angular velocity. The landing sequence requires hovering, aligning, descending, and finally landing, each with specific duration and timing constraints. This can be expressed using TWTL as:*

$$\phi_{landing} := [\mathbf{H}^{100}\pi_{hover}]_{[0,150]} \cdot [\mathbf{H}^{150}\pi_{align}]_{[100,300]} \cdot [\mathbf{H}^{150}\pi_{descend}]_{[250,450]} \cdot [\mathbf{H}^{50}\pi_{land}]_{[400,500]} \quad (2.21)$$

where each π represents a predicate over the appropriate state variables defining hovering altitude, alignment tolerance, descent velocity, and landing conditions. The time horizon is $\|\phi_{landing}\| = 1403\Delta t$. The concatenation operator \cdot ensures the correct sequential ordering of subtasks.

2.2.2 Relationship between STL and TWTL

While both STL and TWTL express temporal properties with explicit time bounds, they differ in expressiveness and computational complexity. STL's **G** and **F** operators can express properties over any subset of the time horizon, whereas TWTL's concatenation operator naturally enforces sequential ordering.

2.3 Traditional Min-Max Robustness for STL

Beyond Boolean satisfaction, quantitative semantics provide a measure of how robustly a signal satisfies a temporal logic specification. The traditional robustness metric for STL, introduced by Donzé and Maler (Maler and Nickovic, 2004) and Fainekos and Pappas (Fainekos and Pappas, 2009), uses min-max operations to compute a real-valued measure of satisfaction degree.

2.3.1 Definition and Properties

The STL robustness $\rho(s_{t_1, t_2}, \varphi)$ of a signal sequence s_{t_1, t_2} with respect to an STL formula φ is defined recursively as:

$$\rho(s_t, \mu) := h(s_t) - \sigma \quad (2.22)$$

$$\rho(s_{t_1, t_2}, \neg\varphi) := -\rho(s_{t_1, t_2}, \varphi) \quad (2.23)$$

$$\rho(s_{t_1, t_2}, \varphi_1 \wedge \varphi_2) := \min\{\rho(s_{t_1, t_2}, \varphi_1), \rho(s_{t_1, t_2}, \varphi_2)\} \quad (2.24)$$

$$\rho(s_{t_1, t_2}, \varphi_1 \vee \varphi_2) := \max\{\rho(s_{t_1, t_2}, \varphi_1), \rho(s_{t_1, t_2}, \varphi_2)\} \quad (2.25)$$

$$\rho(s_{t_1, t_2}, \mathbf{G}_{[a, b]}\varphi) := \min_{t \in [t_1 + a, t_1 + b]} \rho(s_{t_1, t}, \varphi) \quad (2.26)$$

$$\rho(s_{t_1, t_2}, \mathbf{F}_{[a, b]}\varphi) := \max_{t \in [t_1 + a, t_1 + b]} \rho(s_{t_1, t}, \varphi) \quad (2.27)$$

Theorem 2.3.1 (Soundness of STL Robustness (Donzé and Maler, 2010)). *For any signal sequence s_{t_1, t_2} and STL formula φ , the following implications hold:*

$$\rho(s_{t_1, t_2}, \varphi) > 0 \Rightarrow s_{t_1, t_2} \models \varphi \quad (2.28)$$

$$\rho(s_{t_1, t_2}, \varphi) < 0 \Rightarrow s_{t_1, t_2} \not\models \varphi \quad (2.29)$$

The robustness value provides a signed distance to the decision boundary: positive values indicate satisfaction with magnitude reflecting the safety margin, while negative values indicate violation with magnitude reflecting the severity.

2.3.2 Advantages and Limitations

The min-max robustness semantics offers several advantages. First, it provides a quantitative measure that extends Boolean satisfaction, enabling optimization-based control synthesis. Second, the soundness property (Theorem 2.3.1) ensures that positive robustness implies specification satisfaction, making it suitable for safety-critical applications. Third, robustness can guide synthesis algorithms toward solutions with higher safety margins.

However, min-max robustness has significant limitations for planning and optimization

applications. The robustness value is determined entirely by the most critical time point or subformula, effectively masking contributions from all other parts of the signal and specification. This creates a non-smooth optimization landscape with sharp decision boundaries where small changes in the signal can produce abrupt shifts in the critical constraint.

Example 2.3.1 (Limitations of Min-Max Robustness). *Consider two robot trajectories satisfying the obstacle avoidance specification φ_{avoid} from Example 2.1.1. Trajectory $s^{(1)}$ maintains distances $\{1.01, 1.01, 1.01, \dots\}$ from the obstacle at each time step, while trajectory $s^{(2)}$ maintains distances $\{5.0, 5.0, 1.01, 5.0, \dots\}$. Both trajectories have identical robustness $\rho(s^{(1)}, \varphi_{\text{avoid}}) = \rho(s^{(2)}, \varphi_{\text{avoid}}) = 0.01$ because the minimum distance dominates the computation. However, intuitively, trajectory $s^{(2)}$ provides substantially better safety margins overall, with only one critical moment, whereas trajectory $s^{(1)}$ operates near the constraint boundary throughout execution. The min-max semantics fails to distinguish these qualitatively different safety profiles.*

2.4 Arithmetic-Geometric Mean Robustness for STL

To address the limitations of min-max robustness, Mehdipour et al. (Mehdipour et al., 2019) introduced the Arithmetic-Geometric Mean (AGM) robustness for STL. Rather than focusing solely on critical points, AGM robustness evaluates satisfaction holistically across all time points and subformulae, creating smoother optimization landscapes while maintaining soundness guarantees.

2.4.1 AGM Operators

The AGM robustness builds on two key aggregation functions that generalize min and max operations. For robustness values $r_1, \dots, r_N \in \mathbb{R}$, we first define the positive and negative part extraction:

$$[r]_+ := \begin{cases} r & \text{if } r > 0 \\ 0 & \text{otherwise} \end{cases}, \quad [r]_- := -[-r]_+ \quad (2.30)$$

where $r = [r]_+ + [r]_-$.

The AGM disjunction operator is defined as:

$$\text{AGM}_{\vee}(r_1, \dots, r_N) := \begin{cases} -\sqrt[N]{\prod_{i=1}^N (1 - r_i)} + 1 & \text{if } \forall i \in \{1, \dots, N\}, r_i < 0 \\ \frac{1}{N} \sum_{i=1}^N [r_i]_+ & \text{otherwise} \end{cases} \quad (2.31)$$

The AGM conjunction operator is defined as:

$$\text{AGM}_{\wedge}(r_1, \dots, r_N) := \begin{cases} \sqrt[N]{\prod_{i=1}^N (1 + r_i)} - 1 & \text{if } \forall i \in \{1, \dots, N\}, r_i > 0 \\ \frac{1}{N} \sum_{i=1}^N [r_i]_- & \text{otherwise} \end{cases} \quad (2.32)$$

These operators use geometric mean when all values share the same sign (capturing compound effects) and arithmetic mean otherwise (averaging the contributing parts). This design provides smooth transitions while preserving key properties of min and max operations.

2.4.2 AGM Robustness Definition

Definition 2.4.1 (Normalized STL Formula). *An STL formula φ is normalized if all predicates μ are defined over normalized functions: $\mu := h_{norm}(s_t) > \sigma_{norm}$ where $h_{norm} : \mathbb{R} \rightarrow [-1, 1]$ and $\sigma_{norm} \in [-1, 1]$.*

Throughout the remainder of this dissertation, we assume all STL formulae are normalized unless explicitly stated otherwise.

Definition 2.4.2 (STL AGM Robustness (Mehdipour et al., 2019)). *Given a normalized STL formula φ and signal sequence s_{t_1, t_2} , the AGM robustness $\eta(s_{t_1, t_2}, \varphi)$ is defined recur-*

sively using (2.31) and (2.32) as:

$$\eta(s_{t_1, t_2}, \top) := +1, \quad \eta(s_{t_1, t_2}, \perp) := -1 \quad (2.33)$$

$$\eta(s_t, \mu) := \frac{1}{2}(h(s_t) - \sigma) \quad (2.34)$$

$$\eta(s_{t_1, t_2}, \neg\varphi) := -\eta(s_{t_1, t_2}, \varphi) \quad (2.35)$$

$$\eta(s_{t_1, t_2}, \varphi_1 \wedge \varphi_2) := AGM_{\wedge}(\eta(s_{t_1, t_2}, \varphi_1), \eta(s_{t_1, t_2}, \varphi_2)) \quad (2.36)$$

$$\eta(s_{t_1, t_2}, \varphi_1 \vee \varphi_2) := AGM_{\vee}(\eta(s_{t_1, t_2}, \varphi_1), \eta(s_{t_1, t_2}, \varphi_2)) \quad (2.37)$$

$$\eta(s_{t_1, t_2}, \mathbf{G}_{[a, b]} \varphi) := AGM_{\wedge}(\eta(s_{t_1, t}, \varphi) \mid t \in [t_1 + a, t_1 + b]) \quad (2.38)$$

$$\eta(s_{t_1, t_2}, \mathbf{F}_{[a, b]} \varphi) := AGM_{\vee}(\eta(s_{t_1, t}, \varphi) \mid t \in [t_1 + a, t_1 + b]) \quad (2.39)$$

Theorem 2.4.1 (Soundness of AGM Robustness (Mehdipour et al., 2019)). *For any signal sequence s_{t_1, t_2} and normalized STL formula φ , the following equivalences hold:*

$$\eta(s_{t_1, t_2}, \varphi) > 0 \Leftrightarrow \rho(s_{t_1, t_2}, \varphi) > 0 \Rightarrow s_{t_1, t_2} \models \varphi \quad (2.40)$$

$$\eta(s_{t_1, t_2}, \varphi) < 0 \Leftrightarrow \rho(s_{t_1, t_2}, \varphi) < 0 \Rightarrow s_{t_1, t_2} \not\models \varphi \quad (2.41)$$

Theorem 2.4.1 establishes that AGM robustness is sound and agrees with traditional robustness on the sign of satisfaction. However, the magnitude differs: AGM robustness aggregates contributions from all time points and subformulae rather than selecting only the critical ones.

2.4.3 Advantages of AGM Robustness

AGM robustness addresses key limitations of min-max semantics while preserving soundness. First, the holistic evaluation across all time points and subformulae creates more informative gradients for optimization-based synthesis. In Example 2.3.1, AGM robustness would assign higher values to trajectory $s^{(2)}$ because the arithmetic mean in AGM_{\wedge} rewards the larger distances at most time steps. Second, the use of arithmetic and geometric means produces continuously differentiable aggregation functions (except at sign boundaries), yielding smoother optimization landscapes than the sharp min-max operators. Third, AGM robustness naturally balances competing objectives in specifications with mul-

tuple subformulae by considering all contributions rather than focusing only on the worst case.

The computational complexity of evaluating AGM robustness is comparable to min-max robustness: both require $O(|\phi| \cdot |s|)$ time for a formula ϕ of $|\phi|$ number of operators and signal s of length $|s|$. The key difference lies not in computational cost but in the structure of the resulting optimization landscape when used for synthesis.

2.5 Dynamical Systems

We consider discrete-time dynamical systems of the form:

$$x_{k+1} = f(x_k, u_k) \quad (2.42)$$

where $x_k \in \mathcal{X} \subset \mathbb{R}^n$ is the state at time step k , $u_k \in \mathcal{U} \subset \mathbb{R}^m$ is the control input, \mathcal{X} is the state space, \mathcal{U} is the control space, and $f : \mathcal{X} \times \mathcal{U} \rightarrow \mathcal{X}$ is the dynamics function.

Definition 2.5.1 (Lipschitz Continuity). *The dynamics function f is Lipschitz continuous if there exists a constant $L \geq 0$ such that for all $(x_1, u_1), (x_2, u_2) \in \mathcal{X} \times \mathcal{U}$:*

$$\|f(x_1, u_1) - f(x_2, u_2)\|_2 \leq L \sqrt{\|x_1 - x_2\|_2^2 + \|u_1 - u_2\|_2^2} \quad (2.43)$$

where $\|\cdot\|_2$ denotes the Euclidean norm.

Lipschitz continuity ensures that small changes in state or control produce bounded changes in the next state, which is essential for convergence guarantees in sampling-based planning algorithms.

A control sequence $\mathbf{u} = u_0 u_1 \cdots u_{T-1}$ applied to system (2.42) from initial state x_0 generates a state trajectory $\xi = x_0 x_1 \cdots x_T$ satisfying the dynamics. For specifications defined over observable outputs, we define an observation function $l : \mathcal{X} \rightarrow \mathbb{R}^d$ that maps states to signal values. The signal sequence corresponding to trajectory ξ is $s_{0,T} = l(x_0)l(x_1) \cdots l(x_T)$.

In the context of temporal logic specifications, we are interested in synthesizing control sequences that generate trajectories satisfying given STL or TWTL formulae.

Chapter 3

Quantitative Semantics and Runtime Monitoring

This chapter presents our contributions to quantitative semantics and runtime monitoring for temporal logics. Building on the STL AGM robustness framework introduced in Chapter 2, we develop quantitative semantics for Time Window Temporal Logic (TWTL), enabling optimization-based planning for sequential robotic tasks. We then address the challenge of runtime monitoring: evaluating robustness for partial trajectories whose length is less than the formula’s time horizon. Our main contributions are: (1) traditional and AGM robustness measures for TWTL with soundness proofs, (2) interval semantics for STL AGM robustness that bound possible robustness values for any completion of partial signals, (3) efficient incremental monitoring algorithms achieving linear speedup over non-incremental approaches, and (4) interval semantics for both traditional and AGM TWTL robustness. These results enable real-time monitoring and informed decision-making during motion planning before complete trajectories are generated.

3.1 Background and Related Work

3.1.1 Temporal Logics for Dynamical Systems

Temporal logics (TLs) (Baier and Katoen, 2008) have been widely used to formulate high-level, expressive specifications for cyber-physical systems. Formal verification and synthesis algorithms have been employed to analyze and control such systems from TL specifications. Linear Temporal Logic (LTL) (Pnueli, 1977) has been employed to specify tasks for planning problems (Wolff et al., 2013; Kress-Gazit et al., 2018; Luo et al., 2021; Kantaros

and Zavlanos, 2020) and for formal synthesis problems for discrete-time systems (Belta et al., 2017). LTL formulae can be translated to automata, which can encode the progress towards task satisfaction. Automata-theoretic tools are typically used with finite abstractions of the system to produce policies that guarantee the satisfaction of tasks, or prove that they cannot be satisfied (Belta et al., 2017; Kloetzer and Belta, 2008). Other approaches overcome some scalability issues through sampling-based planning algorithms guided by specification automata (Vasile and Belta, 2014; Kantaros and Zavlanos, 2020; Luo et al., 2021; Vasile et al., 2020).

Signal Temporal Logic (STL) (Maler and Nickovic, 2004), Metric Temporal Logic (MTL) (Koymans, 1990), and Time Window Temporal Logic (TWTL) (Vasile et al., 2017a), unlike LTL, can express specifications with explicit, concrete-time constraints. For example, one can specify: *“Perform task A between times t_1 and t_2 ; right after that, spend t_5 time units between times t_3 and t_4 performing task B; and for all times do not perform task C.”*

The semantics of both STL and MTL are defined over real-time signals. They both have quantitative semantics, or robustness, which quantifies the degree of satisfaction of a formula by a signal (Fainekos and Pappas, 2009; Donzé and Maler, 2010). Most existing works that use STL and MTL for specifications find controllers by maximizing robustness, yielding runs of the system that robustly satisfy the specifications (Sadraddini and Belta, 2015; Raman et al., 2014; Donzé and Maler, 2010; Vasile et al., 2017b; Leung et al., 2020). The work in (Castro et al., 2013) considers planning for syntactically co-safe LTL using RRT*, where in addition to task specifications, other spatial requirements are expressed using fragment-STL and its robustness is used as the optimality criterion. In (Rodionova et al., 2021), the authors synthesize controllers for time-critical systems for which they quantify a temporal robustness measure that needs to be optimized.

3.1.2 Robustness Measures for STL

The traditional robustness metric for STL is not differentiable and is mostly determined by one value of the signal—it “masks” most of the signal’s contribution to satisfaction. These issues are addressed by the authors of (Mehdipour et al., 2019), who introduced an arithmetic and geometric mean (AGM) robustness measure for STL. The AGM robustness provides a smoother measure suitable for gradient-based optimization while maintaining soundness guarantees. This approach has been shown to improve performance in trajectory optimization and control synthesis problems.

3.1.3 Time Window Temporal Logic

TWTL (Vasile et al., 2017a) has several advantages over STL, MTL, and other concrete-time TLs. First, its syntax and semantics can express serial tasks in an efficient and explicit way. This is important in many applications, especially in robotics. Second, TWTL formulae can be efficiently translated into automata. The complexity of the translation algorithm is independent of the formula time bounds (Vasile et al., 2017a). This makes this logic suitable for automata-based synthesis and planning problems (see (Penedo et al., 2020) for a motion planning application).

TWTL, however, has lacked quantitative semantics that measure the degree of satisfaction or violation of a formula. This limitation has prevented TWTL from being used in optimization-based synthesis and planning problems that rely on maximizing robustness. Furthermore, the lack of quantitative semantics has hindered the development of runtime monitoring algorithms that can provide real-time feedback on how well a system is satisfying its specification during execution.

3.1.4 Runtime Monitoring

Runtime verification techniques provide lightweight algorithms to monitor the satisfaction of temporal logic specifications given partial system runs (Deshmukh et al., 2017). Different techniques have been employed for monitoring various temporal logics. The work in (Bonnah and Hoque, 2022) uses a rewriting technique to monitor the Boolean satisfaction of TWTL. In (Deshmukh et al., 2017), the authors introduced interval semantics to monitor the robustness degree of STL specifications. This technique considers partial runs, and with the set of all possible completions of the run, it computes the best and worst possible robustness values. The interval semantics provide sound bounds that contain the true robustness value for any completion of the partial trajectory, enabling informed decision-making before complete trajectories are available.

3.1.5 Contributions of This Work

This chapter addresses these gaps by introducing the first quantitative semantics for TWTL. We develop both traditional min-max robustness (adapted from STL) and AGM robustness for TWTL, proving soundness for both measures. We introduce interval semantics that enable runtime monitoring of robustness for partial trajectories, with efficient incremental algorithms that avoid redundant computation. These contributions enable TWTL to be used in optimization-based planning and learning applications while maintaining the logic’s advantages for expressing sequential tasks with bounded time horizons.

3.2 TWTL Traditional Robustness

We first extend the traditional min-max robustness semantics from STL to TWTL, adapting the recursive structure to handle TWTL’s unique operators including the hold, concatenation, and within operators.

3.2.1 Definition and Soundness

Given a TWTL formula ϕ and an output word o_{t_1, t_2} generated by a discrete-time system, we define the robustness degree $\rho(o_{t_1, t_2}, \phi)$ recursively as follows:

$$\rho(o_{t_1, t_2}, \mathbf{H}^d \pi_A) := \begin{cases} \min_{t \in [t_1, t_1 + d\Delta t]} h(o_t) & \text{if } t_2 - t_1 \geq d\Delta t \\ \rho_{\perp} & \text{otherwise} \end{cases} \quad (3.1)$$

$$\rho(o_{t_1, t_2}, \phi_1 \wedge \phi_2) := \min\{\rho(o_{t_1, t_2}, \phi_1), \rho(o_{t_1, t_2}, \phi_2)\} \quad (3.2)$$

$$\rho(o_{t_1, t_2}, \phi_1 \vee \phi_2) := \max\{\rho(o_{t_1, t_2}, \phi_1), \rho(o_{t_1, t_2}, \phi_2)\} \quad (3.3)$$

$$\rho(o_{t_1, t_2}, \neg\phi) := -\rho(o_{t_1, t_2}, \phi) \quad (3.4)$$

$$\rho(o_{t_1, t_2}, \phi_1 \cdot \phi_2) := \max_{t \in [t_1, t_2]} \{\min\{\rho(o_{t_1, t}, \phi_1), \rho(o_{t+\Delta t, t_2}, \phi_2)\}\} \quad (3.5)$$

$$\rho(o_{t_1, t_2}, [\phi]_{[a, b]}) := \begin{cases} \max_{t \geq t_1 + a} \{\rho(o_{t, t_1 + b}, \phi)\} & \text{if } t_2 - t_1 \geq b \\ \rho_{\perp} & \text{otherwise} \end{cases} \quad (3.6)$$

where ρ_{\perp} denotes a large negative value indicating Boolean false, and $h : 2^{\Pi} \rightarrow \mathbb{R}^d$ is the predicate function from the atomic proposition π_A .

The concatenation operator (3.5) requires special attention. It computes the maximum over all possible split points t where ϕ_1 could complete and ϕ_2 could begin, taking the minimum robustness between the two subformulae at each split point. This reflects the sequential nature of concatenation: the overall robustness is limited by the weaker of the two consecutive subformulae, but we choose the split point that maximizes this minimum.

Theorem 3.2.1 (Soundness of TWTL Robustness). *For any output word o_{t_1, t_2} and TWTL formula ϕ , the following implications hold:*

$$\rho(o_{t_1, t_2}, \phi) > 0 \Rightarrow o_{t_1, t_2} \models \phi \quad (3.7)$$

$$\rho(o_{t_1, t_2}, \phi) < 0 \Rightarrow o_{t_1, t_2} \not\models \phi \quad (3.8)$$

Proof. We prove the soundness by structural induction over the formula ϕ . The base case

for TWTL is the *Hold* case, which we prove as follows.

Hold: First, if $t_2 - t_1 < d\Delta t$ then $\rho(\mathbf{o}_{t_1, t_2}, H^d \pi_A) = \rho_{\perp} < 0$ which implies that $\mathbf{o}_{t_1, t_2} \not\models \phi$. Second, if $t_2 - t_1 \geq d\Delta t$, let $\rho(\mathbf{o}_{t_1, t_2}, H^d \pi_A) > 0$. Assume that there is $t' \in [t_1, t_1 + d\Delta t]$ such that $\rho(\mathbf{o}_{t'}, \pi_A) < 0$ then we get $\rho(\mathbf{o}_{t_1, t_2}, H^d \pi_A) = \min_{t \in [t_1, t_1 + d\Delta t]} h(o_t) < 0$, which is a contradiction, then $\mathbf{o}_{t_1, t_2} \models H^d \pi_A$. Third, if $t_2 - t_1 \geq d\Delta t$, let $\rho(\mathbf{o}_{t_1, t_2}, H^d \pi_A) < 0$. Assume that for all $t' \in [t_1, t_1 + d\Delta t]$ such that $\rho(\mathbf{o}_{t'}, \pi_A) > 0$ then $\rho(\mathbf{o}_{t_1, t_2}, H^d \pi_A) = \min_{t \in [t_1, t_1 + \Delta t]} h(o_t) > 0$, which is a contradiction, then $\mathbf{o}_{t_1, t_2} \not\models H^d \pi_A$.

Then we have the following induction cases:

Conjunction: Let $\rho(\mathbf{o}_{t_1, t_2}, \phi_1 \wedge \phi_2) > 0$. Assume that one or both $\rho(\mathbf{o}_{t_1, t_2}, \phi_i) < 0$, $i = 1, 2$, then $\rho(\mathbf{o}_{t_1, t_2}, \phi_1 \wedge \phi_2) = \min\{\rho(\mathbf{o}_{t_1, t_2}, \phi_1), \rho(\mathbf{o}_{t_1, t_2}, \phi_2)\} < 0$, which is a contradiction. Then by induction $\mathbf{o}_{t_1, t_2} \models \phi_1 \wedge \phi_2$. In the case when $\rho(\mathbf{o}_{t_1, t_2}, \phi_1 \wedge \phi_2) < 0$. Assume that $\rho(\mathbf{o}_{t_1, t_2}, \phi_i) > 0$, $i = 1, 2$, then $\rho(\mathbf{o}_{t_1, t_2}, \phi_1 \wedge \phi_2) = \min\{\rho(\mathbf{o}_{t_1, t_2}, \phi_1), \rho(\mathbf{o}_{t_1, t_2}, \phi_2)\} > 0$, which is a contradiction. Then by induction $\mathbf{o}_{t_1, t_2} \not\models \phi_1 \wedge \phi_2$.

Disjunction: Follows similarly to the *conjunction* case with two key changes: (1) replace \min with \max in the robustness computation, so $\rho(\mathbf{o}_{t_1, t_2}, \phi_1 \vee \phi_2) = \max\{\rho(\mathbf{o}_{t_1, t_2}, \phi_1), \rho(\mathbf{o}_{t_1, t_2}, \phi_2)\}$, and (2) use the fact that $\max\{\rho_1, \rho_2\} > 0$ implies at least one of $\rho_1 > 0$ or $\rho_2 > 0$ (hence, at least one subformula is satisfied by the induction hypothesis), which suffices for disjunction satisfaction.

Negation: Let $\phi = \neg\varphi$ and $\rho(\mathbf{o}_{t_1, t_2}, \phi) > 0$. We get $\rho(\mathbf{o}_{t_1, t_2}, \varphi) < 0$, and by induction hypothesis $\mathbf{o}_{t_1, t_2} \not\models \varphi$; hence, $\mathbf{o}_{t_1, t_2} \models \phi$. Similarly, if $\rho(\mathbf{o}_{t_1, t_2}, \phi) < 0$ we get $\mathbf{o}_{t_1, t_2} \models \varphi$; hence, $\mathbf{o}_{t_1, t_2} \not\models \phi$.

Within: Similar to the *hold* case, with the time window $[t_1, t_2]$ replacing the single time point. We verify that there exists some $t \in [t_1, t_2]$ such that $\mathbf{o}_{t_1, t}$ satisfies ϕ_1 and $\mathbf{o}_{t+\delta t, t_2}$ satisfies ϕ_2 , following from $\rho(\mathbf{o}_{t_1, t_2}, \phi_1 \cdot \phi_2) > 0$ by the semantics of concatenation..

Concatenation: Given that the language of TWTL is assumed to be unambiguous¹, we modify the syntax of the concatenation operator as follows.

$$\mathbf{o}_{t_1, t_2} \models \phi_1 \cdot \phi_2 \iff \bigvee_{t \in [t_1, t_2]} \mathbf{o}_{t_1, t} \models \phi_1 \wedge \mathbf{o}_{t+\Delta t, t_2} \models \phi_2 \quad (3.9)$$

Thus, the soundness of $\rho(\mathbf{o}_{t_1, t_2}, \phi_1 \cdot \phi_2)$ follows directly from the soundness of the robustness of the *disjunction* and *conjunction* cases. \square

¹For any formula $\phi_1 \cdot \phi_2$, a satisfying trajectory has a unique decomposition into a prefix satisfying ϕ_1 and a suffix satisfying ϕ_2 . This ensures concatenation is well-defined and prevents multiple interpretations of the same trajectory (Vasile et al., 2017a).

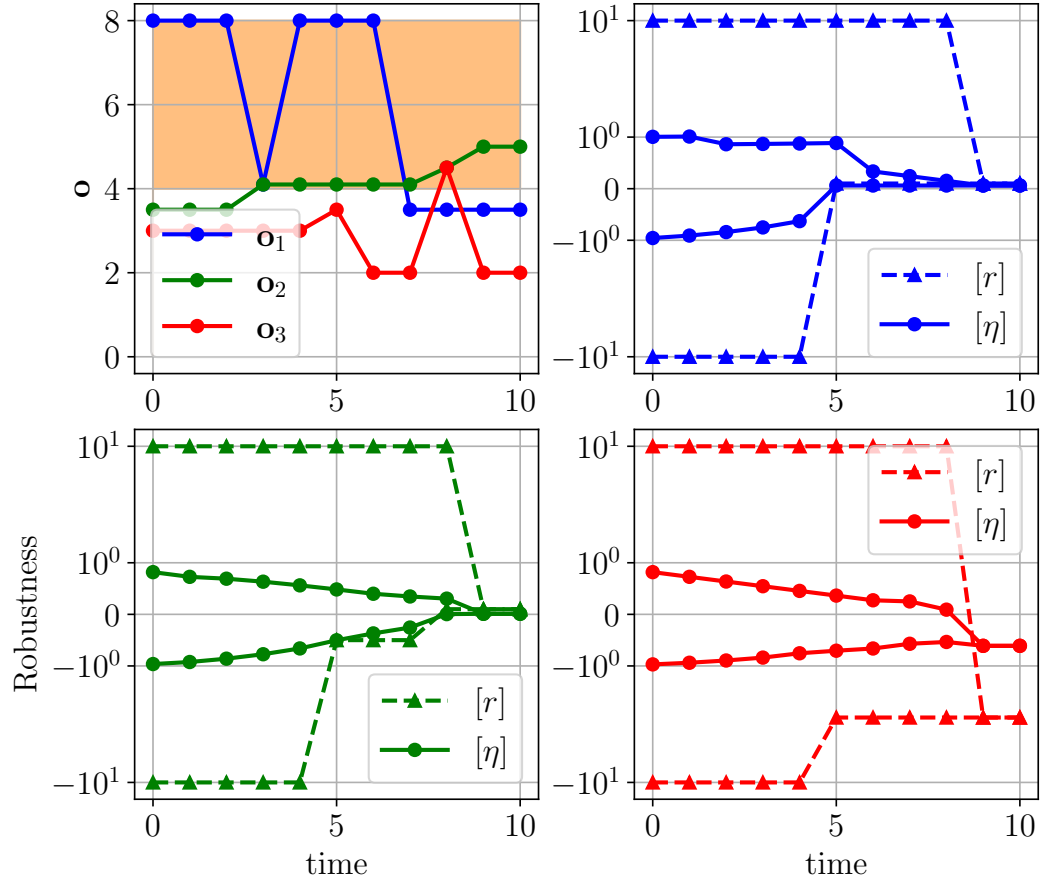


Figure 3-1: Demonstration of TWTL robustness and monitoring. (top-left) Depiction of the valuation of words \mathbf{o}_1 , \mathbf{o}_2 , and \mathbf{o}_3 w.r.t. time. The evolution of the min-max robustness (dashed-line with triangles) and the AGM robustness (solid-lines with circles) for words \mathbf{o}_1 , \mathbf{o}_2 (bottom-left), and \mathbf{o}_3 are shown in the top-right, bottom-left, and bottom-right figures, respectively.

Example 3.2.1. Consider a TWTL formula $\phi = [H^6 \pi_A]^{[0,10]}$, where $A = \{o \mid o \geq 4\}$, which reads as: “Within time 0 and time 10, hold in π_A for 6 time steps; and three output words \mathbf{o}_1 , \mathbf{o}_2 and \mathbf{o}_3 , which are depicted as blue, green, and red traces in the top-left figure of Fig. 3-1, respectively. One can see that \mathbf{o}_1 and \mathbf{o}_2 satisfy the task specification where $\rho(\mathbf{o}_1, \phi) = \rho(\mathbf{o}_2, \phi) = 0.099$, whereas \mathbf{o}_3 violates the task, where $\rho(\mathbf{o}_3, \phi) = -2$.

3.2.2 Limitations

Like STL min-max robustness, TWTL robustness focuses solely on critical points. For the hold operator, only the minimum predicate value over the duration matters. For concate-

nation, only the optimal split point and the minimum of the two subformulae at that split contribute to the robustness. This creates the same non-smooth optimization landscape and lack of holistic evaluation that motivated AGM robustness for STL.

3.3 TWTL Arithmetic-Geometric Mean Robustness

We now extend the AGM robustness framework to TWTL, enabling smoother optimization landscapes for planning with sequential task specifications.

3.3.1 Definition and Soundness

Definition 3.3.1 (Normalized TWTL Formula). *A TWTL formula ϕ is normalized if all atomic propositions π_A are defined over normalized predicate functions: $A := \{o \mid h_{norm}(o) > \sigma_{norm}\}$ where $h_{norm} : 2^\Pi \rightarrow [-1, 1]^d$ and $\sigma_{norm} \in [-1, 1]^d$.*

Throughout the remainder of this chapter, we assume all TWTL formulae are normalized unless explicitly stated otherwise.

Definition 3.3.2 (TWTL AGM Robustness). *Given a normalized TWTL formula ϕ and output word o_{t_1, t_2} , the AGM robustness $\eta(o_{t_1, t_2}, \phi)$ is defined recursively using the AGM operators (2.31) and (2.32) as:*

$$\eta(o_{t_1, t_2}, \top) := +1, \quad \eta(o_{t_1, t_2}, \perp) := -1 \quad (3.10)$$

$$\eta(o_t, \pi_A) := \frac{1}{2}(h(o_t) - \sigma) \quad (3.11)$$

$$\eta(o_{t_1, t_2}, \phi_1 \wedge \phi_2) := \text{AGM}_\wedge(\eta(o_{t_1, t_2}, \phi_1), \eta(o_{t_1, t_2}, \phi_2)) \quad (3.12)$$

$$\eta(o_{t_1, t_2}, \phi_1 \vee \phi_2) := \text{AGM}_\vee(\eta(o_{t_1, t_2}, \phi_1), \eta(o_{t_1, t_2}, \phi_2)) \quad (3.13)$$

$$\eta(o_{t_1, t_2}, \neg\phi) := -\eta(o_{t_1, t_2}, \phi) \quad (3.14)$$

$$\eta(o_{t_1, t_2}, \mathbf{H}^d \pi_A) := \begin{cases} \text{AGM}_\wedge(\eta(o_t, \pi_A) \mid t \in [t_1, t_1 + d\Delta t]) & \text{if } t_2 - t_1 \geq d\Delta t \\ -1 & \text{otherwise} \end{cases} \quad (3.15)$$

$$\eta(o_{t_1, t_2}, [\phi]_{[a, b]}) := \begin{cases} \text{AGM}_\vee(\eta(o_{t, t_1+b}, \phi) \mid t \in [t_1 + a, t_1 + b]) & \text{if } t_2 - t_1 \geq b \\ -1 & \text{otherwise} \end{cases} \quad (3.16)$$

$$\eta(o_{t_1, t_2}, \phi_1 \cdot \phi_2) := \text{AGM}_\vee(\text{AGM}_\wedge(\eta(o_{t_1, t}, \phi_1), \eta(o_{t+\Delta t, t_2}, \phi_2)) \mid t \in [t_1, t_2]) \quad (3.17)$$

The concatenation operator (3.17) aggregates contributions from all possible split points using AGM_\vee , where at each split point we use AGM_\wedge to combine the robustness of the two consecutive subformulae. This holistic evaluation considers all possible ways to satisfy the sequential composition rather than selecting only the optimal split point.

Theorem 3.3.1 (Soundness of TWTL AGM Robustness). *For any output word o_{t_1, t_2} and normalized TWTL formula ϕ , the following equivalences hold:*

$$\eta(o_{t_1, t_2}, \phi) > 0 \Leftrightarrow \rho(o_{t_1, t_2}, \phi) > 0 \Rightarrow o_{t_1, t_2} \models \phi \quad (3.18)$$

$$\eta(o_{t_1, t_2}, \phi) < 0 \Leftrightarrow \rho(o_{t_1, t_2}, \phi) < 0 \Rightarrow o_{t_1, t_2} \not\models \phi \quad (3.19)$$

Proof. We prove the soundness by structural induction over the formula ϕ . The base case for TWTL is the *Hold* case, which we prove as follows.

Hold: First, if $t_2 - t_1 < d\Delta t$ then $\eta(\mathbf{o}_{t_1, t_2}, H^d \pi_A) = -1$ which implies that $\mathbf{o}_{t_1, t_2} \not\models \phi$. Second, if $t_2 - t_1 \geq d\Delta t$, let $\eta(\mathbf{o}_{t_1, t_2}, H^d \pi_A) > 0$. Assume that there is $t' \in [t_1, t_1 + d\Delta t]$ such that $\eta(\mathbf{o}_{t'}, \pi_A) < 0$ then by (2.32) we get $\eta(\mathbf{o}_{t_1, t_2}, H^d \pi_A) = \frac{1}{|[t_1, t_2]|} \sum_{t' \in [t_1, t_1 + d\Delta t]} [\eta(\mathbf{o}_{t'}, \pi_A)]_- < 0$, which is a contradiction, then $\mathbf{o}_{t_1, t_2} \models H^d \pi_A$. Third, if $t_2 - t_1 \geq d$, let $\eta(\mathbf{o}_{t_1, t_2}, H^d \pi_A) < 0$. Assume that for all $t' \in [t_1, t_1 + d\Delta t]$ such that $\eta(\mathbf{o}_{t'}, \pi_A) > 0$ then by (2.31) we get $\eta(\mathbf{o}_{t_1, t_2}, H^d \pi_A) = \frac{1}{|[t_1, t_1 + d\Delta t]|} \sqrt{\prod_{t' \in [t_1, t_1 + d\Delta t]} (1 + \eta(\mathbf{o}_{t'}, \pi_A))} - 1 > 0$, which is a contradiction, then $\mathbf{o}_{t_1, t_2} \not\models H^d \pi_A$.

Then we have the following induction cases:

The soundness for the cases \top , \perp and π_A , and the *conjunction*, *disjunction*, and *negation* cases, follow directly from Theorem 2 in (Mehdipour et al., 2019).

Within: Similar to the *hold* case, but instead of requiring the predicate to hold for the entire duration dd , the *within* operator requires the formula to be satisfied at some point within the time window $[t_1, t_2]$. The key changes are: (1) replace the product (conjunction over all time points) with a disjunction (existence of at least one satisfying point), and (2) use AGM disjunction operator instead of AGM conjunction. If $\eta(\mathbf{o}_{t_1, t_2}, W^{[t_1, t_2]} \phi) > 0$, then by AGM disjunction semantics, there exists some $t \in [t_1, t_2]$ where $\eta(\mathbf{o}_t, \phi) > 0$, which by induction implies $\mathbf{o}_t \models \phi$.

Concatenation: Given the Boolean semantic (3.9), the soundness of $\eta(\mathbf{o}_{t_1, t_2}, \phi_1 \cdot \phi_2)$ follows directly from the soundness of the AGM robustness of the *disjunction* and *conjunction* cases. \square

Example 3.3.1. (Continued) Consider the same formula and words of Example 3.2.1.

$\eta(\mathbf{o}_1, \phi) = 0.061$, $\eta(\mathbf{o}_2, \phi) = 0.010$, where, unlike their ρ value, the AGM robustness measure η rewards words with more satisfying valuations instead of being dominated by the most critical valuations while also preserving the soundness property. The word \mathbf{o}_1 (the blue trace in top-left figure of Fig. 3.1) has more valuations that robustly contribute to satisfying the formula. In $\rho(\mathbf{o}_1, \phi)$, the 4th point of the trace, which is close to lead to violating the task, dominates the robustness computation. Even if we assume that the 4th is 2 (which would lead to violating the task), its η value would be -0.35 that is higher than its corresponding ρ value, -2 . The computation of η considers the fact that the trace has promising valuations which contribute to satisfying the task. For \mathbf{o}_3 (the red trace in the same figure), on the other hand, $\eta(\mathbf{o}_3, \phi) = -0.61$ which is higher than $\rho(\mathbf{o}_3, \phi) = -2$, is more realistic violation measure given that some valuations of \mathbf{o}_3 are close to contributing in satisfying the task.

3.3.2 Computational Complexity

Computing TWTL AGM robustness requires evaluating the recursive definition over the formula structure. For a formula ϕ with $|\phi|$ operators and an output word o of length $|o|$, the complexity is $O(|\phi| \cdot |o|)$, identical to traditional TWTL robustness. The concatenation operator requires considering all $O(|o|)$ possible split points, each requiring $O(1)$ aggregation operations. Despite the holistic evaluation, AGM robustness maintains the same computational complexity as min-max robustness.

3.4 STL AGM Robustness Interval Semantics

Runtime monitoring requires evaluating robustness for partial signals whose length is less than the formula’s time horizon. We develop interval semantics that bound all possible robustness values for any completion of the partial signal. This section presents our contribution of AGM robustness intervals for STL, which enables efficient incremental monitoring during motion planning.

3.4.1 Preliminaries

Definition 3.4.1 (Prefix and Completions). Consider a signal sequence $s_{t_1, t'}$ where $t' < \|\varphi\|$, called a prefix. A completion of $s_{t_1, t'}$ is any signal sequence s_{t_1, t_2} with $t_2 \geq \|\varphi\|$ such that $s_{t_1, t_2}(t) = s_{t_1, t'}(t)$ for all $t \in [t_1, t']$. We denote the set of all completions as $\mathfrak{C}(s_{t_1, t'}) := \{s_{t_1, t_2} \mid s_{t_1, t'} \text{ is a prefix of } s_{t_1, t_2}\}$.

Definition 3.4.2 (Interval Arithmetic for AGM Operators). Consider intervals $I_i = [\underline{I}_i, \bar{I}_i]$ for $i = 1, \dots, N$ where $\underline{I}_i \leq \bar{I}_i$. We define interval extensions of the AGM operators as:

$$\mathbf{AGM}_{\vee}(\{I_i\}_{i=1}^N) := [\mathbf{AGM}_{\vee}(\underline{I}_1, \dots, \underline{I}_N), \mathbf{AGM}_{\vee}(\bar{I}_1, \dots, \bar{I}_N)] \quad (3.20)$$

$$\mathbf{AGM}_{\wedge}(\{I_i\}_{i=1}^N) := [\mathbf{AGM}_{\wedge}(\underline{I}_1, \dots, \underline{I}_N), \mathbf{AGM}_{\wedge}(\bar{I}_1, \dots, \bar{I}_N)] \quad (3.21)$$

A singleton interval $[\eta, \eta]$ is denoted $\{\eta\}$.

3.4.2 Incremental Modification Functions

A key innovation in our approach is the development of incremental modification functions that efficiently update AGM robustness values when new observations become available, avoiding full recomputation.

Definition 3.4.3 (Incremental AGM Modification Functions). When monitoring a formula with N time points or subformulae, and given the current AGM robustness η computed from $N - 1$ values, we define functions that incorporate the N -th observation η' incrementally.

For disjunction:

$$mdf_AGM_{\vee}(\eta, N, \eta') := \begin{cases} -\sqrt[N]{(1-\eta)^{N-1} \cdot (1-\eta')} + 1 & \text{if } \eta < 0 \wedge \eta' < 0 \\ \frac{[\eta']_+}{N} & \text{if } \eta < 0 \wedge \eta' > 0 \\ \frac{(N \cdot \eta - [\eta]_+) + [\eta']_+}{N} & \text{otherwise} \end{cases} \quad (3.22)$$

For conjunction:

$$mdf_AGM_{\wedge}(\eta, N, \eta') := \begin{cases} \sqrt[N]{(1+\eta)^{N-1} \cdot (1+\eta')} - 1 & \text{if } \eta > 0 \wedge \eta' > 0 \\ \frac{[\eta']_-}{N} & \text{if } \eta > 0 \wedge \eta' < 0 \\ \frac{(N \cdot \eta + [\eta]_-) + [\eta']_-}{N} & \text{otherwise} \end{cases} \quad (3.23)$$

Lemma 3.4.1 (Correctness of Incremental Modification). *Let $\eta = \text{AGM}_\circ(r_1, \dots, r_{N-1})$ where $\circ \in \{\vee, \wedge\}$. For any new observation η' :*

$$\text{mdf_AGM}_\circ(\eta, N, \eta') = \text{AGM}_\circ(r_1, \dots, r_{N-1}, \eta') \quad (3.24)$$

Proof Sketch. The proof proceeds by case analysis on the signs of η and η' , corresponding to the three cases in equations (3.23) and (3.23).

Geometric Mean Case ($\eta < 0, \eta' < 0$ for \vee ; $\eta > 0, \eta' > 0$ for \wedge): Since η resulted from geometric mean aggregation, we have $(1 \mp \eta)^{N-1} = \prod_{i=1}^{N-1} (1 \mp r_i)$ (where \mp is $-$ for disjunction, $+$ for conjunction). Incorporating the new value η' yields $\text{AGM}_\circ(r_1, \dots, r_{N-1}, \eta') = \pm \sqrt[N]{(1 \mp \eta)^{N-1} (1 \mp \eta')} \pm 1$, which exactly matches the first case of each modification function.

Transition Case ($\eta < 0, \eta' > 0$ for \vee ; $\eta > 0, \eta' < 0$ for \wedge): When all previous values had one sign and the new value has the opposite sign, the AGM switches from geometric to arithmetic mean. Since all previous contributions were zero in the arithmetic formulation ($[r_i]_\pm = 0$ when signs differ from required), only the new value contributes: $\text{AGM}_\circ(\dots, \eta') = \frac{[\eta']_\pm}{N}$.

Arithmetic Mean Case (otherwise): When at least one previous value had the appropriate sign for arithmetic aggregation, we have $(N-1)\eta = \sum_{i=1}^{N-1} [r_i]_\pm$. However, $[\eta]_\pm$ captures the result rather than the sum of inputs, so the correct update is $\frac{(N-1)\eta - [\eta]_\pm + [\eta']_\pm}{N}$.

Arithmetic Mean Case (otherwise): When at least one previous value had the appropriate sign for arithmetic aggregation, we have $\eta = \sum_{i=1}^{N-1} [r_i]_\pm / (N-1)$. To incorporate the new value η' , we extract its contribution $[\eta']_\pm$ (applying the positive or negative part extraction based on the aggregation sign). Since η represents the average of $N-1$ values, the sum of previous contributions is $(N-1)\eta$. Adding the new contribution and re-averaging over N values yields: $\text{AGM}_\circ(\dots, \eta') = \frac{(N-1)\eta + [\eta']_\pm}{N}$.

The full proof with detailed algebraic manipulations is provided in Appendix A.1. \square

3.4.3 Incremental Runtime Monitoring Algorithm

Algorithm 1 presents our incremental runtime monitor for STL AGM robustness. The algorithm maintains an AGM robustness interval $[\eta]_\phi$ that bounds all possible robustness values for any completion of the current partial signal. At each new observation $s_{t'}$, the algorithm recursively updates the interval by traversing the formula's abstract syntax tree.

Algorithm 1: $\text{IRTM}(\phi, [\eta], [\eta]_{\text{aux}}, \mathbf{s}_{t'}, t', t_s)$

Input: $\phi, [\eta], [\eta]_{\text{aux}}, \mathbf{s}_{t'}, t', t_s$
Output: $[\eta]'(\phi)$

- 1 **if** $[\eta]_{\text{aux}} \neq \emptyset$ **then**
- 2 \lfloor **return** $[\eta]_{\phi}$
- 3 **else if** $\phi = \mu$ **then**
- 4 \lfloor $[\eta]_{\text{aux}} \leftarrow [h(\mathbf{s}_{t'}), h(\mathbf{s}_{t'})]$
- 5 **else if** $\phi \in \{\wedge \phi_i, \vee \phi_i\}$ **then**
- 6 $\mathcal{E} \leftarrow \emptyset$
- 7 **foreach** $i \in \{1, \dots, N\}$ **do**
- 8 \lfloor $\mathcal{E} \leftarrow \mathcal{E} \cup \{\text{IRTM}(\phi_i, [\eta], [\eta]_{\text{aux}}, \mathbf{s}_{t'}, t', t_s)\}$
- 9 **if AND then**
- 10 \lfloor $[\eta]_{\text{aux}}(\phi) \leftarrow \text{AGM}_{\wedge}(\mathcal{E})$
- 11 **else**
- 12 \lfloor $[\eta]_{\text{aux}}(\phi) \leftarrow \text{AGM}_{\vee}(\mathcal{E})$
- 13 **else if** $\phi = \mathbf{G}_{[a,b]} \phi_1$ **then**
- 14 \lfloor $[\eta]_{\text{aux}} \leftarrow \text{IRTM}(\phi_1, [\eta], \emptyset, \mathbf{s}_{t'}, t', t_s)$
- 15 \lfloor $[\eta]_{\text{aux}} \leftarrow \text{IRTM}_{\mathbf{T}}([\eta], [\eta]_{\text{aux}}, t', t_s, a, b, \|\phi\|, \mathbf{G})$ ▷ Algorithm 2
- 16 **else if** $\phi = \mathbf{F}_{[a,b]} \phi_1$ **then**
- 17 \lfloor $[\eta]_{\text{aux}} \leftarrow \text{IRTM}(\phi_1, [\eta], \emptyset, \mathbf{s}_{t'}, t', t_s)$
- 18 \lfloor $[\eta]_{\text{aux}} \leftarrow \text{IRTM}_{\mathbf{T}}([\eta], [\eta]_{\text{aux}}, t', t_s, a, b, \|\phi\|, \mathbf{F})$ ▷ Algorithm 2
- 19 **return** $[\eta]_{\text{aux}}(\phi)$

For temporal operators, the subroutine Algorithm 2 handles the time-dependent logic, using the incremental modification functions from Definition 3.4.3 to efficiently incorporate new observations.

3.4.4 Theoretical Properties

Lemma 3.4.2 (Soundness of AGM Robustness Intervals). *For any STL formula ϕ , partial signal $s_{t_1, t'}$, and completion $s \in \mathcal{C}(s_{t_1, t'})$, the AGM robustness value is contained in the computed interval:*

$$\eta(s, \phi) \in [\eta]_{s_{t'}, \phi} = [\underline{\eta}, \bar{\eta}] \quad (3.25)$$

where $[\eta]_{s_{t'}, \phi} \leftarrow \text{IRTM}(\phi, [\eta]_{s_{t'}, \phi}, \emptyset, s_{t'}, t', t_0)$.

Proof Sketch. We prove by structural induction over ϕ . For predicates, the interval is a

singleton containing the exact value. For Boolean operators, the interval arithmetic in Definition 3.4.2 preserves the containment property. For temporal operators, Algorithm 2 constructs intervals by considering best-case and worst-case completions: the lower bound assumes all future values are -1 , while the upper bound assumes all future values are $+1$. By Lemma 3.4.1, the incremental updates correctly maintain these bounds as new observations arrive. The full proof is provided in Appendix A.2. \square

Theorem 3.4.1 (Chain Inclusion Property). *Given partial signals s_{t_1, t'_1} and s_{t_1, t'_2} where $t'_1 < t'_2 < t_2$, the AGM robustness intervals satisfy:*

$$[\eta]_{s_{t'_2}, \varphi} \subseteq [\eta]_{s_{t'_1}, \varphi} \quad (3.26)$$

Proof Sketch. As the partial signal grows from s_{t_1, t'_1} to s_{t_1, t'_2} , the set of possible completions shrinks: $\mathfrak{C}(s_{t_1, t'_2}) \subseteq \mathfrak{C}(s_{t_1, t'_1})$. By Lemma 3.4.2, the interval contains all possible robustness values of completions. Fewer possible completions yield narrower robustness value ranges, resulting in the inclusion property. The full proof is in Appendix A.3 \square

Corollary 3.4.1 (Convergence to Exact Value). *When the partial signal extends to or beyond the formula horizon, $t' \geq \|\varphi\|$, the AGM robustness interval converges to a singleton:*

$$[\eta]_{s_{t'}, \varphi} = \{\eta(s_{t_1, t'}, \varphi)\} \quad (3.27)$$

3.4.5 Complexity Analysis

For a formula φ with $|\varphi|$ operators and a signal of length $|s|$, each invocation of Algorithm 1 processes a single new observation by traversing the formula's abstract syntax tree in $O(|\varphi|)$ time. The incremental modification functions (Definition 3.4.3) execute in $O(1)$ time by Lemma 3.4.1. Therefore, monitoring a complete signal requires $O(|s| \cdot |\varphi|)$ total operations.

In contrast, a non-incremental implementation that recomputes the entire robustness interval from scratch at each time step requires $O(|s| \cdot |\varphi|)$ operations per update, yielding $O(|s|^2 \cdot |\varphi|)$ total complexity. Our incremental approach achieves a linear speedup factor of $O(|s|)$. For typical planning scenarios with $|s| \approx 20 - 50$ and $|\varphi| \approx 10 - 20$, this reduces monitoring overhead from thousands to hundreds of operations per trajectory evaluation.

3.5 TWTL Robustness Interval Semantics

We now present interval semantics for runtime monitoring of both traditional and AGM robustness for TWTL. While the STL interval semantics in Section 3.4 provided the foundation, TWTL's unique operators (particularly concatenation) require specialized treatment.

3.5.1 Interval Semantics for Traditional TWTL Robustness

Given a partial output word $o_{t_1, t'}$ where $t' < \|\phi\|$, we define the traditional robustness interval $[\rho](o_{t_1, t'}, \phi)$ recursively as:

$$[\rho](o_{t_1, t_2}, \mathbf{H}^d \pi_A) := \begin{cases} \{\rho(o_{t_1, t_2}, \mathbf{H}^d \pi_A)\} & \text{if } t_2 - t_1 \geq d\Delta t \\ [\rho_{\perp}, \min_{t \in [t_1, t_1 + d\Delta t]} h(o_t)] & \text{otherwise} \end{cases} \quad (3.28)$$

$$[\rho](o_{t_1, t_2}, \phi_1 \wedge \phi_2) := \min([\rho](o_{t_1, t_2}, \phi_1), [\rho](o_{t_1, t_2}, \phi_2)) \quad (3.29)$$

$$[\rho](o_{t_1, t_2}, \phi_1 \vee \phi_2) := \max([\rho](o_{t_1, t_2}, \phi_1), [\rho](o_{t_1, t_2}, \phi_2)) \quad (3.30)$$

$$[\rho](o_{t_1, t_2}, \phi_1 \cdot \phi_2) := \max_{t \in [t_1, t_2]} (\min([\rho](o_{t_1, t}, \phi_1), [\rho](o_{t+\Delta t, t_2}, \phi_2))) \quad (3.31)$$

$$[\rho](o_{t_1, t_2}, [\phi]_{[a, b]}) := \begin{cases} \{\rho(o_{t_1, t_2}, [\phi]_{[a, b]})\} & \text{if } t_2 - t_1 \geq b \\ [\max_{t \geq t_1 + a} \{\rho(o_{t, t_1 + b}, \phi)\}, \rho_{\top}] & \text{otherwise} \end{cases} \quad (3.32)$$

where ρ_{\top} denotes a large positive value representing the best-case completion, and the min and max operations on intervals are defined in (3.21) and (3.20) (specialized to min/max rather than AGM operators).

3.5.2 Interval Semantics for TWTL AGM Robustness

The AGM robustness interval $[\eta](o_{t_1,t'}, \phi)$ for partial TWTL words follows a similar recursive structure, using the AGM interval arithmetic from Definition 3.4.2:

$$[\eta](o_{t_1,t'}, \phi_1 \wedge \phi_2) := \text{AGM}_\wedge([\eta](o_{t_1,t'}, \phi_1), [\eta](o_{t_1,t'}, \phi_2)) \quad (3.33)$$

$$[\eta](o_{t_1,t'}, \phi_1 \vee \phi_2) := \text{AGM}_\vee([\eta](o_{t_1,t'}, \phi_1), [\eta](o_{t_1,t'}, \phi_2)) \quad (3.34)$$

$$[\eta](o_{t_1,t'}, \mathbf{H}^d \pi_A) := [\underline{\eta}, \bar{\eta}] \quad (3.35)$$

For the hold operator (3.35), the lower and upper bounds are computed based on the observed partial word and worst-case/best-case assumptions about future observations:

$$\bar{\eta} := \begin{cases} \sqrt{d+1} \sqrt{\prod_{t \in [t_1, t']} (1 + \eta(\mathbf{o}_t, \pi_A)) (1 + \eta_{\max})^{|[t', d]|}} - 1; & \text{if } \forall t \in [t_1, t'], \eta(\mathbf{o}_t, \pi_A) > 0 \wedge (t' - t_1) < d \\ \frac{1}{d+1} \sum_{t \in [t_1, t']} [\eta(\mathbf{o}_t, \pi_A)]_-; & \text{if } \exists t \in [t_1, t'], \eta(\mathbf{o}_t, \pi_A) < 0 \wedge (t' - t_1) < d \\ \eta(\mathbf{o}_{t_1, t'}, H^d \pi_A); & \text{If } (t' - t_1) \geq d \end{cases} \quad (3.36)$$

$$\underline{\eta} := \begin{cases} \frac{|[t', d]|}{|[t_1, d]|} \eta_{\min}; & \text{if } \forall t \in [t_1, t'], \eta(\mathbf{o}_t, \pi_A) > 0 \wedge (t' - t_1) < d \\ \frac{1}{d+1} \left(\sum_{t \in [t_1, t']} [\eta(\mathbf{o}_t, \pi_A)]_- + |[t', d]| \eta_{\min} \right); & \text{if } \exists t \in [t_1, t'], \eta(\mathbf{o}_t, \pi_A) < 0 \wedge (t' - t_1) < d \\ \eta(\mathbf{o}_{t_1, t'}, H^d \pi_A); & \text{If } (t' - t_1) \geq d \end{cases}$$

where $\eta_{\min} = -1$ and $\eta_{\max} = +1$ represent the bounds on normalized predicates.

For the within operator:

$$[\eta](o_{t_1, t'}, [\phi]_{[a, b]}) := \begin{cases} \{\eta(o_{t_1, t'}, [\phi]_{[a, b]})\} & \text{if } t' - t_1 \geq b \\ \mathbf{AGM}_V([\eta](o_{t, t_1 + b}, \phi) \mid t \in [t_1 + a, t']) & \text{otherwise} \end{cases} \quad (3.37)$$

For the concatenation operator:

$$[\eta](o_{t_1, t'}, \phi_1 \cdot \phi_2) := \mathbf{AGM}_V(\mathbf{AGM}_\wedge([\eta](o_{t_1, t}, \phi_1), [\eta](o_{t, t'}, \phi_2)) \mid t \in [t_1, t']) \quad (3.38)$$

Example 3.5.1. (Continued) Given the TWTL unnormalized formula $\phi = [H^6 \pi_A]^{[0, 10]}$ with time horizon $\|\phi\| = 10$, we demonstrate the proposed runtime monitors by observing the robustness intervals $[\rho]$ and $[\eta]$ of partial words of \mathbf{o}_1 , \mathbf{o}_2 , and \mathbf{o}_3 (see the top-left figure of Fig. 3-1). Consider the time series $\tau = \{0, \dots, 10\}$. For each partial word we compute $[\rho]$ and $[\eta]$ at every $t \in \tau$ as the partial words $\mathbf{o}_1(t)$, $\mathbf{o}_2(t)$, and $\mathbf{o}_3(t)$ become available. The evolution of the intervals $[\rho]$ and $[\eta]$ for \mathbf{o}_1 , \mathbf{o}_2 , and \mathbf{o}_3 are depicted in the top-right, bottom-left, and bottom-right figures of Fig. 3-1, respectively, where the evolution of $[\rho]$ is depicted in dashed-lines with triangles and the evolution of $[\eta]$ is depicted in solid-lines with circles. Notice how the intervals become tighter as the partial word grows, where eventually when $t' = \|\phi\|$, $[\rho]$ and $[\eta]$ converge to the true ρ and η values, respectively. In this example we consider $\rho_\top = 10$ and $\rho_\perp = -10$; one can notice how the evolution of $[\eta]$ is smoother than the evolution of $[\rho]$, due to using the AGM in the computation of η . Note that we normalize the TWTL formula before computing η robustness values. Thus, in Fig. 3-1, η stays within $[-1, 1]$.

3.5.3 Soundness and Convergence

Theorem 3.5.1 (Soundness of TWTL Robustness Intervals). *For any TWTL formula ϕ , partial output word $o_{t_1, t'}$, and completion $o \in \mathfrak{C}(o_{t_1, t'})$:*

$$\rho(o, \phi) \in [\rho](o_{t_1, t'}, \phi) \quad (3.39)$$

$$\eta(o, \phi) \in [\eta](o_{t_1, t'}, \phi) \quad (3.40)$$

Proof Sketch. The proof proceeds by structural induction over ϕ , following similar reasoning to Lemma 3.4.2. For the hold operator, the interval bounds are constructed by considering best-case (η_{max} for all future observations) and worst-case (η_{min} for all future

observations) completions. The recursive cases follow from the interval arithmetic properties combined with the induction hypothesis. \square

The convergence property (analogous to Corollary 3.4.1) also holds for TWTL: when $t' \geq \|\phi\|$, the intervals converge to singletons containing the exact robustness values.

3.6 Case Study: Planar Robot Navigation

We demonstrate the proposed robustness semantics, by monitoring ρ and η for pre-computed runs of a simple planar robot system. Assume the time step, Δt , of the runs is 1. We consider a simple sequential navigation task with deadlines and a safety requirement. In the following unnormalized TWTL formula, we encode the task that reads: *Within time 0 and 8, visit region A and stay there for 3 time steps; right after that, within time 0 and 10, visit region B and stay there for 4 time steps; right after that, within time 0 and 11, visit region C and stay there for 3 time steps; and for all execution time avoid region O.* See the left figure of Fig. 3-2 for a depiction of the planar regions A, B, C, and O.

$$\phi = \left([H^4 \pi_A]^{[0,8]} \cdot [H^4 \pi_B]^{[0,10]} \cdot [H^3 \pi_C]^{[0,11]} \right) \wedge H^{50} \neg \pi_O \quad (3.41)$$

The atomic propositions π_A, π_B, π_C , and π_O are defined as predicated regions over the xy -plane; where $A := \{(x,y) | 1 \leq x \leq 4 \wedge 1 \leq y \leq 4\}$, $B := \{(x,y) | 8 \leq x \leq 11 \wedge 3 \leq y \leq 5\}$, $C := \{(x,y) | 1 \leq x \leq 4 \wedge 9 \leq y \leq 12\}$, and $O := \{(x,y) | 5 \leq x \leq 7 \wedge 5 \leq y \leq 7\}$.

We monitor two runs of the robot, \mathbf{o}_1 and \mathbf{o}_2 , which are shown as the blue and green traces in the left figure of Fig. 3-2, respectively. The robustness of \mathbf{o}_1 and \mathbf{o}_2 are $\rho(\mathbf{o}_1, \phi) = 0.4$ and $\rho(\mathbf{o}_2, \phi) = 0.3$, whereas their AGM robustness are $\eta(\mathbf{o}_1, \phi) = 0.00076$ and $\eta(\mathbf{o}_2, \phi) = 0.00015$. To monitor the robustness measures at runtime, consider the time series $\tau = \{2, 10, 15, 20, 25, 30, 35, 40, 42\}$. For each partial word we compute $[\rho]$ and $[\eta]$ at every $t \in \tau$ as the partial words $\mathbf{o}_1(t)$ and $\mathbf{o}_2(t)$ become available. The valuations of the intervals $[\rho]$ and $[\eta]$ for \mathbf{o}_1 , and \mathbf{o}_2 at every $t \in \tau$ are depicted in the middle, and right

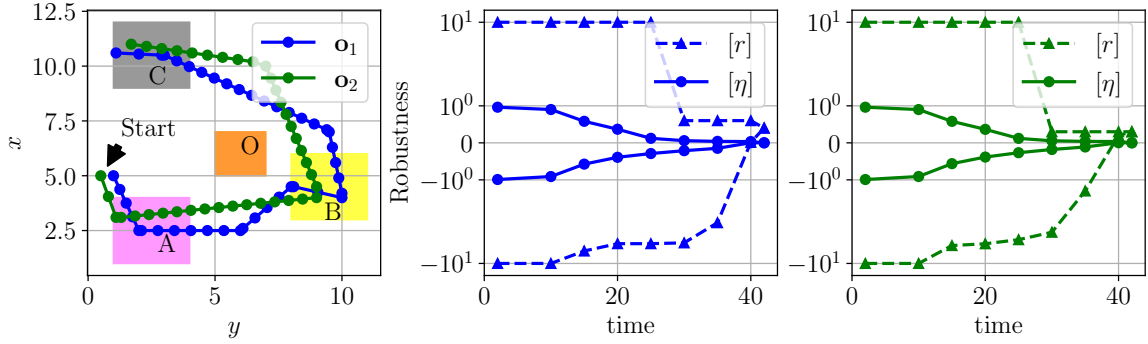


Figure 3-2: Demonstration of monitoring the TWTL robustness and AGM TWTL robustness of precomputed planar robot runs, \mathbf{o}_1 and \mathbf{o}_2 , against satisfying formula (3.41). Words \mathbf{o}_1 and \mathbf{o}_2 in an xy – planar environment are shown in the blue and green traces in the left figure, respectively. The valuations of the intervals $[\rho]$ and $[\eta]$ for \mathbf{o}_1 , and \mathbf{o}_2 at every $t \in \{2, 10, 15, \dots, 40, 42\}$ are depicted in the middle, and right figures, respectively, where $[\rho]$ is depicted in dashed-lines with triangles and the $[\eta]$ is depicted in solid-lines with circles.

figures of Fig. 3-2, respectively, where $[\rho]$ is depicted in dashed-lines with triangles and the $[\eta]$ is depicted in solid-lines with circles.

Observe that monitoring convergence of $[\eta]$ is smoother than the convergence of $[\rho]$, which is more useful in some applications.

3.7 Summary

This chapter presented our contributions to quantitative semantics and runtime monitoring for temporal logics. We developed traditional and AGM robustness measures for TWTL, extending the STL framework to handle sequential task specifications with hold, concatenation, and within operators. We proved soundness of both measures, showing that positive robustness implies specification satisfaction.

For runtime monitoring with partial trajectories, we developed AGM robustness interval semantics for STL, introducing incremental modification functions that enable efficient updating without full recomputation. Our algorithms achieve $O(|s|)$ speedup over non-incremental approaches. We established theoretical properties including soundness, chain

inclusion, and convergence to exact values.

Finally, we presented interval semantics for both traditional and AGM TWTL robustness, adapting the STL framework to TWTL's unique operators. These runtime monitors enable informed decision-making during motion planning before complete trajectories are generated, which is essential for the sampling-based planning algorithms presented in Chapter 4.

Algorithm 2: $\text{IRTM}_T([\eta], [\eta]_{\text{aux}}, t_s, t', a, b, \|\phi\|)$

```

1 Input:  $[\underline{\eta}, \overline{\eta}] \leftarrow [\eta], [\underline{\eta}', \overline{\eta}'] \leftarrow [\eta]_{\text{aux}}, t_s, t', a, b$ 
2 Output:  $[\underline{\eta}, \overline{\eta}]$ 
3 if  $t' < t_s + a$  then
4   return  $\emptyset$ 
5 if  $\underline{\eta} = \overline{\eta}$  then
6   return  $[\underline{\eta}, \overline{\eta}]$ 
7 if  $[\eta] = \emptyset$  then
8    $\mathcal{J}_- \leftarrow \{\underline{\eta}', -1, \dots, -1\}, \mathcal{J}_+ \leftarrow \{\overline{\eta}', 1, \dots, 1\};$  where  $|\mathcal{J}_-| = |\mathcal{J}_+| = \|\phi\|$ 
9   if  $\text{type}=\mathbf{G}$  then
10     $\underline{\eta}' \leftarrow \text{AGM}_\wedge(\mathcal{J}_-)$ 
11     $\overline{\eta}' \leftarrow \text{AGM}_\wedge(\mathcal{J}_+)$ 
12   if  $\text{type}=\mathbf{F}$  then
13     $\underline{\eta}' \leftarrow \text{AGM}_\vee(\mathcal{J}_-)$ 
14     $\overline{\eta}' \leftarrow \text{AGM}_\vee(\mathcal{J}_+)$ 
15   return  $[\eta', \eta']$ 
16 if  $t' \geq t_s + b$  then
17   if  $\text{type}=\mathbf{G}$  then
18      $\eta' \leftarrow \text{mdf\_AGM}_\wedge(\overline{\eta}, N, \overline{\eta}')$ 
19   if  $\text{type}=\mathbf{F}$  then
20      $\eta' \leftarrow \text{mdf\_AGM}_\vee(\overline{\eta}, N, \overline{\eta}')$ 
21   return  $[\eta', \eta']$ 
22 else
23   if  $\text{type}=\mathbf{G}$  then
24      $\underline{\eta}' \leftarrow \text{mdf\_AGM}_\wedge(\underline{\eta}, N, \underline{\eta}')$ 
25      $\overline{\eta}' \leftarrow \text{mdf\_AGM}_\wedge(\overline{\eta}, N, \overline{\eta}')$ 
26   if  $\text{type}=\mathbf{F}$  then
27      $\underline{\eta}' \leftarrow \text{mdf\_AGM}_\vee(\underline{\eta}, N, \underline{\eta}')$ 
28      $\overline{\eta}' \leftarrow \text{mdf\_AGM}_\vee(\overline{\eta}, N, \overline{\eta}')$ 
29   return  $[\underline{\eta}', \overline{\eta}']$ 

```

Chapter 4

RRT ^{η} : Sampling-Based Motion Planning with AGM Robustness

4.1 Introduction

Chapter 3 introduced quantitative semantics for Time Window Temporal Logic, developing both traditional min-max robustness and Arithmetic-Geometric Mean (AGM) robustness. We demonstrated that AGM robustness provides smoother, more holistic evaluation of specification satisfaction while maintaining formal soundness guarantees. The incremental monitoring algorithms enable efficient computation of robustness intervals for partial trajectories, providing sound bounds that narrow as trajectories extend. These theoretical foundations and computational tools now enable us to integrate AGM robustness into sampling-based motion planning frameworks.

Sampling-based motion planning algorithms such as Rapidly-exploring Random Trees (RRT) (LaValle, 1998) and RRT* (Karaman and Frazzoli, 2011) have proven effective for exploring high-dimensional configuration spaces without explicit discretization. When combined with Signal Temporal Logic (STL) specifications, these methods can address complex spatiotemporal constraints beyond simple point-to-point navigation. However, existing robustness-based planning approaches rely on traditional min-max semantics that focus exclusively on critical time points and subformulae, creating non-smooth optimization landscapes with sharp decision boundaries that hinder efficient tree exploration.

This chapter presents RRT ^{η} (Ahmad et al., 2026), which integrates AGM robustness

semantics with RRT* for STL-guided motion planning. While Chapter 3 developed AGM robustness for TWTL with efficient monitoring algorithms, this chapter adapts these concepts to STL specifications in the context of sampling-based planning. Our framework introduces three key innovations that leverage the AGM foundation from Chapter 3: (1) AGM robustness interval semantics specialized for STL, enabling evaluation of partial trajectories during tree construction using the incremental monitoring approach; (2) Direction of Increasing AGM Satisfaction (DIAS) vectors that exploit AGM’s smooth gradients to guide exploration; and (3) Fulfillment Priority Logic (FPL)-based composition for principled multi-objective balancing. We prove that RRT^η maintains the probabilistic completeness and asymptotic optimality guarantees of RRT* while providing superior exploration behavior in scenarios with multiple competing temporal objectives.

4.2 Background and Related Work

4.2.1 Sampling-Based Motion Planning

Sampling-based methods have significantly advanced motion planning and control synthesis for robotic systems in complex, high-dimensional environments. Algorithms such as Rapidly-exploring Random Trees (RRT) (LaValle, 1998) and variants (Otte and Frazzoli, 2016; Kobilarov, 2012; Wu et al., 2020; Ahmad et al., 2022; Yang et al., 2025) efficiently explore configuration spaces without explicit discretization, particularly useful for nonlinear dynamics. RRT* (Karaman and Frazzoli, 2011) extends RRT with rewiring to ensure asymptotic convergence to optimal paths. These algorithms have proven effective for kinodynamic planning where dynamic constraints must be satisfied.

Recently, researchers have integrated control barrier functions (CBFs) with sampling-based planners (Ahmad et al., 2022; Perez et al., 2012; Majd et al., 2021) to synthesize controllers guaranteeing set invariance while eliminating explicit collision checking. These approaches combine the exploration efficiency of sampling-based methods with the safety

guarantees of CBFs, enabling navigation in complex environments with formal safety certificates.

4.2.2 Temporal Logic Specifications in Motion Planning

Temporal logic specifications enable richer task descriptions beyond basic reachability, encoding complex sequences of goals with explicit timing constraints. Two main paradigms have emerged for integrating temporal logics with sampling-based planning.

Automata-Based Approaches

Methods such as (Penedo et al., 2020; Vasile and Belta, 2013; Vasile and Belta, 2014) leverage finite-state abstractions to guide tree construction toward specification satisfaction. These approaches translate temporal logic formulae into automata, then guide sampling toward automaton states indicating progress. While effective, automata-based methods provide only binary satisfaction guarantees without quantitative robustness measures that could guide the search toward more robust solutions.

Robustness-Based Methods

An alternative paradigm uses quantitative semantics to both verify satisfaction strength and guide synthesis. Within this paradigm, optimization-based approaches (Sadraddini and Belta, 2015; Raman et al., 2014) formulate the problem as Mixed-Integer Linear Programs (MILP) or nonlinear optimization. These methods can find optimal solutions but suffer from scalability limitations in high-dimensional spaces.

Sampling-based robustness methods like STL-RRT* (Vasile et al., 2017b) guide tree exploration through the Direction of Increasing Satisfaction (DIS) vector, which provides gradient-like information toward regions of higher specification satisfaction. The DIS leverages the robustness gradient to steer sampling and tree expansion.

However, STL-RRT* and related approaches rely on traditional min-max robustness

metrics that evaluate satisfaction based solely on the most critical time points and subformulae. As discussed in Chapter 3 for TWTL semantics, this creates fundamental limitations: the robustness value is determined by a single critical constraint, causing small trajectory changes to produce abrupt shifts in which constraints dominate the evaluation. This results in non-smooth optimization landscapes that provide inconsistent gradient information during tree expansion. Furthermore, when specifications contain multiple competing objectives expressed through Boolean operators (e.g., $\phi_1 \wedge \phi_2$), traditional approaches use simple selection mechanisms (typically choosing the subformula with lower robustness) that fail to provide principled balancing of requirements.

4.2.3 AGM Robustness for STL

Chapter 3 developed AGM robustness semantics for TWTL, demonstrating that holistic aggregation across all time points and subformulae produces smoother evaluation landscapes while maintaining soundness guarantees. The AGM approach uses geometric mean when all satisfaction values have the same sign (capturing compound effects) and arithmetic mean when values have mixed signs (averaging contributions). We also introduced interval semantics for partial trajectories, enabling sound robustness bounds to be computed incrementally as trajectories extend.

Prior work by Mehdipour et al. (Mehdipour et al., 2019) introduced AGM robustness for STL specifications in the context of continuous trajectory optimization and control synthesis. Their work demonstrated that AGM robustness provides more consistent gradient information throughout the state space compared to traditional min-max approaches, leading to improved performance in gradient-based optimization.

However, neither the original AGM robustness work for STL (Mehdipour et al., 2019) nor our TWTL extension in Chapter 3 addressed integration into sampling-based planning frameworks. Key open challenges include: (1) developing interval semantics for STL that enable reasoning about partial trajectories during tree construction, (2) adapting the Di-

rection of Increasing Satisfaction concept to leverage AGM’s smooth gradients, and (3) maintaining the probabilistic completeness and asymptotic optimality guarantees of RRT* when using AGM-based guidance.

This chapter addresses these challenges by specializing the AGM robustness and incremental monitoring framework from Chapter 3 to STL specifications in the context of sampling-based motion planning. We adapt the interval semantics and monitoring algorithms to handle STL’s temporal operators (Globally, Finally) and develop enhanced direction vectors that exploit AGM’s holistic evaluation properties.

4.2.4 Multi-Objective Composition

The challenge of principled objective composition extends beyond temporal logic. Recent work in Fulfillment Priority Logic (FPL) (Mabsout et al., 2025) has demonstrated nuanced decision-making frameworks that prioritize less-fulfilled objectives using power mean operators. FPL provides a family of aggregation operations parameterized by a power parameter p , where $p \rightarrow -\infty$ approaches minimum (conservative/conjunctive behavior), $p = 1$ gives arithmetic mean, and $p \rightarrow \infty$ approaches maximum (optimistic/disjunctive behavior). This principled framework enables smooth interpolation between worst-case and best-case aggregation while maintaining mathematical guarantees on minimum fulfillment levels.

The power mean framework aligns naturally with AGM robustness: both use aggregation operators that smooth the optimization landscape while providing formal guarantees. We integrate FPL’s power mean derivatives into our Direction of Increasing AGM Satisfaction vectors, enabling principled composition of objectives from multiple subformulae based on their current fulfillment levels rather than simple stochastic selection.

4.2.5 Contributions of This Work

This chapter makes three main contributions that build upon the AGM robustness foundation from Chapter 3:

(1) AGM Robustness Interval Semantics for STL. We specialize the interval semantics and incremental monitoring algorithms from Chapter 3 to STL specifications. While Chapter 3 addressed TWTL with its explicit time windows and Hold operators, STL uses Globally and Finally operators with different temporal semantics. We develop efficient algorithms (Algorithms 1-2) that compute AGM robustness intervals for partial trajectories, enabling informed tree expansion decisions before complete paths are formed. We prove soundness (Lemma IV.2) and interval convergence properties (Theorem IV.1, Corollary IV.1).

(2) Direction of Increasing AGM Satisfaction (DIAS). We enhance the Direction of Increasing Satisfaction concept from STL-RRT* to leverage AGM robustness. Unlike traditional DIS that relies on min-max gradients, DIAS exploits the smooth, holistic gradients of AGM robustness. For Boolean operators, we introduce two composition approaches: a stochastic baseline that preserves asymptotic optimality through randomization, and an FPL-based method that uses power mean derivatives to prioritize less-fulfilled objectives. The FPL approach provides principled weighting based on fulfillment contributions while maintaining probabilistic completeness through controlled randomization.

(3) Formal Guarantees. We prove that RRT⁷ maintains the fundamental properties of RRT*: probabilistic completeness (Theorem V.1) ensures that feasible solutions are eventually found with probability one, and asymptotic optimality (Theorem V.2) guarantees convergence to optimal AGM robustness values. These proofs extend existing RRT* convergence analysis to handle AGM robustness intervals, DIAS guidance, and FPL composition while preserving the theoretical guarantees.

We validate RRT⁷ on unicycle robots with nonholonomic constraints and 7-DOF manipulators in high-dimensional configuration spaces, demonstrating that AGM-based methods discover feasible solutions where traditional robustness-based planning fails, while FPL composition provides computational advantages over stochastic baselines.

4.3 Problem Formulation

Consider a robot with state $q \in \mathcal{Q} \subset \mathbb{R}^n$ evolving according to the discrete-time dynamics:

$$q_{k+1} = f(q_k, u_k) \quad (4.1)$$

where $u_k \in \mathcal{U} \subset \mathbb{R}^m$ is the control input at time step k , and $f : \mathcal{Q} \times \mathcal{U} \rightarrow \mathcal{Q}$ is Lipschitz continuous (Definition 2.5.1).

A control sequence $\mathbf{u} = u_0 u_1 \cdots u_{T-1}$ applied from initial state q_0 generates a state trajectory $\xi = q_0 q_1 \cdots q_T$ satisfying (4.1). For specifications over observable signals, we define an observation function $l : \mathcal{Q} \rightarrow \mathbb{R}$ that maps states to signal values, yielding signal sequence $s_{0,T} = l(q_0) l(q_1) \cdots l(q_T)$.

Definition 4.3.1 (Control-Trajectory Pair). *A control-trajectory pair $\varphi = (\mathbf{u}, \xi)$ consists of a control sequence \mathbf{u} of length T and the corresponding state trajectory ξ of length $T + 1$ satisfying the dynamics (4.1).*

Given an STL specification φ with time horizon $\|\varphi\|$, we evaluate satisfaction using the AGM robustness $\eta(s_{0,T}, \varphi)$ from Definition 2.4.2, where $s_{0,T}$ is the signal sequence corresponding to trajectory ξ .

Definition 4.3.2 (Feasible Control-Trajectory Set). *Given initial state q_{init} and STL formula φ , the set of feasible control-trajectory pairs is:*

$$\mathcal{G} := \{(\mathbf{u}, \xi) \mid q_0 = q_{init}, \eta(s_{0,T}, \varphi) > 0, T \leq \|\varphi\|\} \quad (4.2)$$

Problem 4.3.1 (Robust Planning Problem). *Given robot dynamics (4.1), initial state $q_{init} \in \mathcal{Q}$, and STL specification φ , find the control-trajectory pair $\varphi^* \in \mathcal{G}$ that maximizes AGM robustness:*

$$\varphi^* = \arg \max_{\varphi \in \mathcal{G}, T \in \mathbb{Z}_{>0}} \eta(s_{0,T}, \varphi) \quad (4.3)$$

Problem 4.3.1 seeks control sequences that not only satisfy the specification (positive AGM robustness) but do so with maximum strength across all aspects of the specification. The AGM robustness objective creates a non-convex optimization landscape, but one

that is significantly smoother than traditional min-max robustness due to its arithmetic and geometric mean aggregations.

4.4 Direction of Increasing AGM Satisfaction

To guide tree expansion toward regions that improve specification satisfaction, we adapt the Direction of Increasing Satisfaction (DIS) concept from STL-RRT* (Vasile et al., 2017b) to leverage AGM robustness gradients. Our Direction of Increasing AGM Satisfaction (DIAS) uses information from the AGM robustness intervals computed by the monitoring algorithms in Section 3.4.

4.4.1 Base Cases and Temporal Operators

For a trajectory ξ generated by system (4.1), we define the DIAS function $\chi_\eta : \mathbb{R}^n \times \Phi_{STL} \times \mathbb{Z}_{\geq 0} \rightarrow \mathbb{R}^n$ that maps a trajectory, formula, and time point to a direction vector in the state space.

For the base cases:

$$\chi_\eta(\xi, \top, t) := \mathbf{0}_n \quad (4.4)$$

$$\chi_\eta(\xi, \mu, t) := \begin{cases} \nabla_q \eta(s_t, \mu)^T \cdot \mathbf{J}_f(q_t, u_t) & \text{if } \nabla_q \eta(s_t, \mu)^T \cdot \mathbf{J}_f(q_t, u_t) \cdot u_t > 0 \\ \mathbf{0}_n & \text{otherwise} \end{cases} \quad (4.5)$$

where $\mathbf{J}_f(q_t, u_t) = [\frac{\partial f}{\partial q_1}, \dots, \frac{\partial f}{\partial q_n}]$ is the Jacobian matrix of the dynamics function f evaluated at state q_t and control u_t . The gradient $\nabla_q \eta(s_t, \mu)$ is computed from the predicate function $h(s_t)$ through the chain rule.

The condition in (4.5) checks whether the system dynamics naturally increase the predicate's satisfaction at the current state under the current control. If the dot product is positive, the DIAS points in the direction of the gradient; otherwise, it returns the zero vector indicating no preferred direction.

For temporal operators, we delegate to the subformula:

$$\chi_\eta(\xi, \mathbf{G}_{[a,b]}\varphi_1, t) = \chi_\eta(\xi, \mathbf{F}_{[a,b]}\varphi_1, t) := \chi_\eta(\xi, \varphi_1, t) \quad (4.6)$$

Example 4.4.1 (DIAS of Fixed Control Action). *Consider a 2D discrete-time dynamical system with state $q = [x, y]^\top \in \mathbb{R}^2$ and control input $u \in \mathbb{R}^2$. The system dynamics are given by $q_{k+1} = q_k + u$. The Jacobian with respect to the state is $\mathbf{J}_f = \mathbf{I}_2$, where $\mathbf{I}_2 \in \mathbb{R}^{2 \times 2}$ is the identity matrix.*

Let the spatial predicate be $\mu = \{q \mid \|q - c\|^2 \leq r^2\}$ with center $c = [3.5, 3.5]^\top$ and radius $r = 1$. The robustness is $\eta(q, \mu) = r - \|q - c\|$, normalized to range $[-1, 1]$. The gradient is $\nabla_q \eta(q, \mu) = -\frac{q-c}{\|q-c\|}$, which points toward the center c with unit magnitude.

For a fixed control input $u = [0.5, 0.5]^\top$, the DIAS is $\chi_\eta(q, \mu) = \nabla \eta(q, \mu)^\top$.

However, $\chi_\eta(q, \mu)$ is nonzero only where the condition $\nabla \eta(q, \mu)^\top \cdot (q_{k+1} - q_k) > 0$ is satisfied, i.e., where the state transition aligns with the direction of increasing satisfaction. Figure 4-1 visualizes this: the DIAS field (d) is active only in regions where the gradient field (b) and the state transition direction (c) point in similar directions, ensuring that the control action moves the system toward higher robustness.

4.4.2 Boolean Operators and Composition Challenge

For Boolean operators $\varphi = \varphi_1 \circ \varphi_2$ where $\circ \in \{\wedge, \vee\}$, we must compose DIAS vectors from multiple subformulae:

$$\chi_\eta(\xi, \varphi_1 \circ \varphi_2, t) := \text{compose}(\chi_\eta(\xi, \varphi_1, t), \chi_\eta(\xi, \varphi_2, t), [\eta]_{\xi_t, \varphi_1}, [\eta]_{\xi_t, \varphi_2}) \quad (4.7)$$

where $[\eta]_{\xi_t, \varphi_i}$ are the AGM robustness intervals from Section 3.4.

The composition function is critical as it determines how the planner balances competing objectives. We present two approaches: a stochastic baseline that preserves theoretical guarantees, and our principled FPL-based method that provides superior performance.

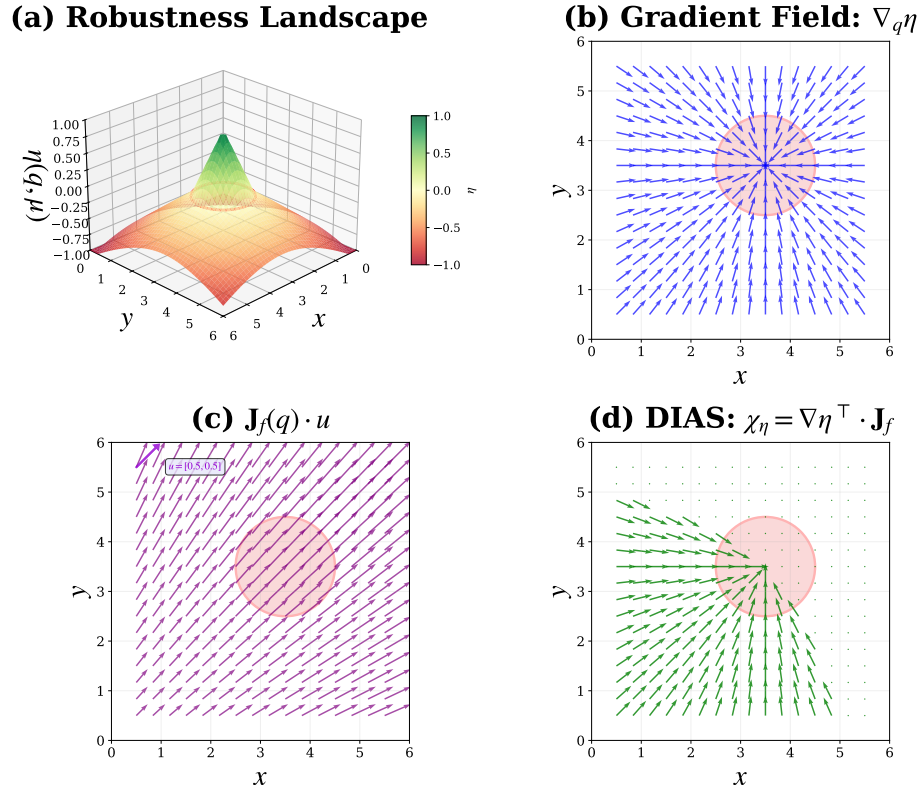


Figure 4.1: Illustration of the Direction of Increasing Satisfaction (DIAS) for a spatial predicate $\mu = \{q \mid \|q - c\|^2 \leq r^2\}$ with center $c = [3.5, 3.5]^\top$ and radius $r = 1$. (a) The robustness landscape $\eta(q, \mu)$ showing the satisfaction region (red circle) where $\eta > 0$. (b) The gradient field $\nabla_q \eta(q, \mu)$ pointing toward the direction of increasing satisfaction. (c) The state transition field $\mathbf{J}_f(q) \cdot u$ resulting from the control input $u = [0.5, 0.5]^\top$ and discrete-time dynamics with Jacobian $\mathbf{J}_f = \mathbf{I}_2$. (d) The DIAS field $\chi_\eta(q, \mu) = \nabla \eta(q, \mu)^\top \cdot \mathbf{J}_f$, which is nonzero only in regions where $\nabla \eta(q, \mu)^\top \cdot (q_{k+1} - q_k) > 0$ (condition satisfied).

4.5 Composition Strategies for Multi-Objective Balancing

4.5.1 Stochastic Composition Baseline

We first adapt the stochastic composition from STL-RRT* (Vasile et al., 2017b) to work with AGM robustness intervals. Let $\chi_1 = \chi_\eta(\xi, \varphi_1, t)$ and $\chi_2 = \chi_\eta(\xi, \varphi_2, t)$ denote the DIAS vectors for two subformulae, with corresponding intervals $[\eta]_1 = [\underline{\eta}_1, \bar{\eta}_1]$ and $[\eta]_2 = [\underline{\eta}_2, \bar{\eta}_2]$.

Definition 4.5.1 (Stochastic Composition). *The stochastic composition function is:*

$$\text{compose}_{stoch}(\chi_1, \chi_2, [\eta]_1, [\eta]_2) := \text{blend}_\eta(\chi_c, \chi_{-c}) \quad (4.8)$$

where $(c, -c) = \text{choose}_\eta([\eta]_1, [\eta]_2)$ with the stochastic choice function:

$$\text{choose}_\eta([\eta]_1, [\eta]_2) := \begin{cases} (1, 2) & \text{if } \bar{\eta}_1 < \underline{\eta}_2 \wedge \underline{\eta}_1 < \bar{\eta}_2 \\ (2, 1) & \text{if } \bar{\eta}_1 > \underline{\eta}_2 \wedge \underline{\eta}_1 > \bar{\eta}_2 \\ (1 + \text{Ber}(p), 2 - \text{Ber}(p)) & \text{otherwise} \end{cases} \quad (4.9)$$

where $\text{Ber}(p)$ is a Bernoulli random variable with parameter:

$$p = 0.5 + \frac{(\underline{\eta}_1 + \bar{\eta}_1) - (\underline{\eta}_2 + \bar{\eta}_2)}{8} \quad (4.10)$$

The blend function combines the selected directions geometrically:

$$\text{blend}_\eta(\chi_c, \chi_{-c}) := \begin{cases} \chi_c + \chi_{-c} & \text{if } \chi_c \perp \chi_{-c} \\ \chi_c & \text{otherwise} \end{cases} \quad (4.11)$$

The rationale for this design is twofold. First, when DIAS vectors are orthogonal, they provide complementary information that does not conflict, so we combine them additively. Second, when vectors are not orthogonal, prioritizing the chosen direction prevents oscillatory behavior from conflicting guidance. The stochastic choice mechanism is crucial for asymptotic optimality: by introducing randomness when the dominance relationship be-

tween intervals is ambiguous, we ensure the algorithm maintains non-zero probability of exploring all potentially optimal directions.

4.5.2 FPL-Based Principled Composition

To provide more principled composition that explicitly balances competing subformulae based on their fulfillment levels, we introduce an approach leveraging Fulfillment Priority Logic (FPL) (Mabsout et al., 2025).

Fulfillment Values and Power Means

FPL provides a family of aggregation operators based on power means that unify minimum and maximum operations. We derive fulfillment values $f \in [0, 1]$ from AGM robustness intervals by mapping from $[-1, 1]$ to $[0, 1]$:

$$f_i = \frac{(\underline{\eta}_i + \overline{\eta}_i + 2)}{4} \quad (4.12)$$

where $[\underline{\eta}_i, \overline{\eta}_i]$ is the AGM robustness interval for subformula ϕ_i .

The power mean operator $\mu_p : [0, 1]^n \rightarrow [0, 1]$ provides a continuous family of aggregation operations:

$$\mu_p(\mathbf{f}) = \left(\frac{1}{n} \sum_{i=1}^n f_i^p \right)^{1/p} \quad (4.13)$$

The parameter p controls composition behavior: $p \rightarrow -\infty$ approaches minimum (conservative aggregation suitable for conjunction), $p = 1$ gives arithmetic mean, and $p \rightarrow \infty$ approaches maximum (optimistic aggregation suitable for disjunction). This provides a principled way to interpolate between worst-case and best-case aggregation while maintaining smooth transitions.

FPL-Based Direction Composition

Definition 4.5.2 (FPL-Based Composition). *For subformulae $\varphi_1, \dots, \varphi_m$ with DIAS vectors χ_i and AGM robustness intervals $[\eta]_i$, we define:*

$$\text{compose}_{FPL}(\{(\chi_i, [\eta]_i)\}_{i=1}^m, \circ) := \begin{cases} \sum_{i=1}^m \chi_i & \text{if } \chi_i \perp \chi_j, \forall i \neq j \\ \sum_{i=1}^m w_i \cdot \chi_i & \text{otherwise} \end{cases} \quad (4.14)$$

where the weights w_i are computed using derivatives of the power mean:

$$w_i = \frac{f_i^p \cdot \frac{\partial \mu_p(\mathbf{f})}{\partial f_i}}{\sum_{j=1}^m f_j^p \cdot \frac{\partial \mu_p(\mathbf{f})}{\partial f_j}} + \alpha_i \quad (4.15)$$

with $p = -1$ for conjunction ($\circ = \wedge$) and $p = 1$ for disjunction ($\circ = \vee$).

The randomization term α_i preserves asymptotic optimality while producing more principled compositions than purely stochastic selection:

$$\alpha_i = \beta \cdot r_i \cdot \left(1 - \max_{j \neq i} |f_i - f_j|\right) \quad (4.16)$$

where β is a small scaling factor (typically 0.1), $r_i \in [-1, 1]$ is uniformly sampled, and the term $(1 - \max_{j \neq i} |f_i - f_j|)$ ensures randomness diminishes as fulfillment differences increase.

The gradient-based weights in (4.15) naturally prioritize less-fulfilled objectives, providing principled balancing. When evaluating competing subformulae during tree expansion, such as whether to prioritize reaching region A or region B, FPL computes fulfillment values f_i from the current AGM robustness intervals. These fulfillment values indicate how well each subformula is currently satisfied. FPL then uses power mean derivatives to compute weights that guide exploration toward states balancing all specification requirements rather than optimizing only the most critical constraint.

Minimum Fulfillment Guarantees

A key theoretical advantage of FPL is the power mean's minimum fulfillment bound:

Theorem 4.5.1 (Minimum Fulfillment Bound (Mabsout et al., 2025)). *For any $p \in \mathbb{R}$ and $\mathbf{f} \in [0, 1]^n$:*

$$\min(\mathbf{f}) \geq \sqrt[p]{n((\mu_p(\mathbf{f}))^p - 1) + 1} \quad (4.17)$$

This bound guarantees that when a power mean outputs value y , every input component must have at least fulfillment $\sqrt[p]{n(y^p - 1) + 1}$. For conjunction operations with negative p values, this provides stronger guarantees on minimum fulfillment, ensuring all subformulae are adequately satisfied rather than focusing predominantly on the most critical one.

4.6 Steering Functions

The steering function plays a critical role in extending the tree toward promising regions. We define the basic steering function $\text{steer} : \mathcal{Q} \times \mathcal{U} \times \mathbb{Z}_{>0} \rightarrow \mathcal{Q}$, where $q_s \leftarrow \text{steer}(q_{t_0}, \mathbf{u}, \Delta t)$ applies control input \mathbf{u} to system (4.1) starting from initial state q_{t_0} for duration Δt , resulting in final state q_s .

For exact steering between states, we define $\text{steer_exact} : \mathcal{Q} \times \mathcal{Q} \rightarrow \bigcup_{i \in \mathbb{Z}_{>0}} (\mathcal{U}^i \times \mathcal{Q}^i) \cup \{\emptyset\}$, where $\varphi \leftarrow \text{steer_exact}(q_{start}, q_{final})$ returns the control-trajectory pair $\varphi = (\mathbf{u}, \xi)$ if q_{final} is reachable from q_{start} , and \emptyset otherwise.

4.6.1 DIAS-Guided Steering

The guided steering using DIAS leverages the direction of increasing satisfaction to compute optimal control inputs. For a nearby vertex $v' \in \mathcal{N}$ in the tree with time difference $\Delta t_r := t_r - v'.t$, we compute the optimal control input \mathbf{u}^* by solving:

$$\mathbf{u}^* \leftarrow \arg \min_{\mathbf{u} \in \mathcal{U}} J_\chi(\mathbf{u}, \text{steer}(v.q, \mathbf{u}, \Delta t_r), v.q, v.\varphi, q_r, \Delta t_r, \lambda) \quad (4.18)$$

where the cost function balances two objectives:

$$J_{\chi}(\mathbf{u}, q_s, v'.q, v'.\varphi, q_r, \Delta t_r; \lambda) := \lambda \|q_s - (v'.q + \mathbf{d}_{\chi} \cdot \Delta t_r)\|_2^2 + (1 - \lambda) \|q_s - q_r\|_2^2 \quad (4.19)$$

Here, $\mathbf{d}_{\chi} := \chi_{\eta}(v'.\xi, v'.\varphi, v'.t)$ represents the DIAS vector at vertex v' , and $\lambda \in [0, 1]$ is a weighting factor balancing movement along the DIAS direction versus toward the random sample q_r .

4.6.2 Optimization Considerations

Solving the optimal control problem (4.18) exactly is generally intractable for nonlinear systems, as it requires optimizing over the control space \mathcal{U} subject to nonlinear dynamics $f(\cdot, \cdot)$. The problem is non-convex due to the coupling between control inputs and resulting states through the steering function, and the cost landscape may contain local minima, particularly when the DIAS vector \mathbf{d}_{χ} points away from the sampled configuration q_r .

In practice, we employ gradient-based local optimization initialized from random samples in \mathcal{U} . For systems with differential constraints (such as the unicycle robot in Section 4.9.1), we use trajectory optimization with finite-horizon discretization. For kinematic systems (such as the KUKA manipulator in Section 4.9.2), the problem reduces to selecting among feasible inverse kinematics solutions based on the cost function. The computational cost per optimization is $O(N_{iter} \cdot T_f)$, where $N_{iter} \approx 10 - 50$ iterations and T_f is the forward dynamics evaluation time.

This local optimization approach trades global optimality for computational efficiency, which is acceptable in the RRT ^{η} framework since multiple steering attempts occur during tree construction, and suboptimal individual connections can be improved through subsequent rewiring operations.

4.7 RRT ^{η} Algorithm

4.7.1 Tree Structure and Vertex Attributes

The RRT ^{η} algorithm constructs a tree $\mathcal{T} = (\mathcal{V}, \mathcal{E})$ where \mathcal{V} represents the set of vertices (nodes) and $\mathcal{E} = \mathcal{V} \times \mathcal{V}$ denotes the set of edges. Each vertex $v \in \mathcal{V}$ contains the following attributes:

- $v.q \in \mathcal{Q}$: system configuration
- $v.\varphi \in \Phi_{STL}$: active STL specification
- $v.t \in \mathbb{Z}_{\geq 0}$: time when state is reached
- $v.\text{traj} = (\mathbf{u}, \xi)$: control-trajectory pair from root to this vertex, where $\mathbf{u} = u_0 u_1 \cdots u_{v.t-1}$ and $\xi = q_0 q_1 \cdots q_{v.t}$
- $v.\text{parent} \in \mathcal{V} \cup \{\emptyset\}$: parent vertex
- $v.\text{children} \subseteq \mathcal{V}$: set of children vertices
- $v.[\eta] = [\underline{\eta}, \bar{\eta}]$: AGM robustness interval from Section 3.4

4.7.2 Main Algorithm

Algorithm 3 presents the main RRT ^{η} procedure. The algorithm initializes the tree with the initial state (Lines 2-3), then iteratively expands the tree toward satisfying the specification with maximum AGM robustness.

At each iteration, the algorithm samples a time-state pair (t_r, q_r) biased toward regions relevant to the STL specification (Line 5), using the sampling procedure from STL-RRT* (Vasile et al., 2017b) that identifies active predicates and generates configurations within those regions. The algorithm then determines nearby vertices \mathcal{N} respecting causality constraints (Line 6) and generates a random convex coefficient λ for balancing DIAS guidance versus random exploration (Line 7).

For each nearby vertex $v' \in \mathcal{N}$ (Lines 9-16), the algorithm computes the DIAS vector (Line 11) and solves the optimal control problem (4.18) to find the control input that best balances movement along the DIAS direction and toward the sampled configuration (Line 12). The resulting state q_s is evaluated (Line 14), and if it improves the current best solution while being reachable, it becomes the new best candidate (Lines 15-16).

After identifying the best parent vertex, the algorithm creates a new vertex (Line 18) and invokes the update procedure (Line 19) which adds the vertex to the tree and computes its AGM robustness interval using the incremental monitoring algorithms from Section 3.4.

The rewiring phase (Lines 20-23) considers whether nearby vertices would benefit from connecting through the newly added vertex v_{new} rather than their current parents. This is where our approach differs fundamentally from traditional RRT*: while RRT* rewires to minimize path cost, RRT ^{η} rewires to maximize AGM robustness.

4.7.3 Update Procedure with AGM Robustness

Algorithm 4 extends the update procedure from STL-RRT* to work with AGM robustness intervals. The key innovation is the use of the incremental monitoring algorithm IRTM (Algorithm 1) to efficiently compute AGM robustness intervals as new trajectory segments are added.

The procedure begins by retrieving the parent vertex's AGM robustness interval and time (Lines 1-2), then obtains the trajectory segment connecting parent to child (Line 3). For each state in the segment, the incremental monitor IRTM updates the AGM robustness interval (Lines 4-7), using the modification functions from Definition 3.4.3 to avoid full recomputation.

For new vertices (Lines 9-13), the algorithm only adds the vertex if the upper bound $\bar{\eta}_2$ is non-negative, indicating that some completion of the partial trajectory could satisfy the specification. For rewiring (Lines 14-22), the algorithm selects connections that improve the lower bound $\underline{\eta}_2$ while maintaining formula consistency. The lower bound optimization

strategy provides conservative guarantees: since $\underline{\eta}$ represents the worst-case robustness over all possible trajectory completions (Lemma 3.4.2), maximizing $\underline{\eta}$ ensures that even the least favorable completion maintains high robustness.

When rewiring changes a vertex's parent, all descendants must have their AGM robustness intervals recomputed (Lines 19-22), as their trajectories from the root have changed. This recursive update maintains the correctness of the AGM robustness intervals throughout the tree.

4.8 Theoretical Analysis

We now establish that RRT^η maintains the fundamental theoretical properties of RRT^* despite the more complex AGM robustness measure.

4.8.1 Probabilistic Completeness

Theorem 4.8.1 (Probabilistic Completeness of RRT^η). *Consider RRT^η (Algorithm 3) applied to a dynamical system with Lipschitz continuous dynamics (4.1), an STL specification φ , and initial state q_{init} . Let \mathcal{G} denote the set of feasible control-trajectory pairs (Definition 4.3.2). If $\mathcal{G} \neq \emptyset$, then:*

$$\lim_{n \rightarrow \infty} \mathbb{P}(RRT^\eta \text{ finds } \varphi \in \mathcal{G}) = 1 \quad (4.20)$$

where n is the number of iterations.

Proof. The proof relies on two key properties:

Stochastic Exploration. The sampling procedure (Line 5 of Algorithm 3) combined with the randomized composition mechanisms (Definitions 4.5.1 and 4.5.2) ensures non-zero probability of exploring all regions of the state space. For FPL-based composition, the randomization term α_i in (4.16) maintains exploration. For stochastic composition, the Bernoulli choice in (4.9) ensures no region is permanently excluded. Combined with the random weighting λ in (4.19), this provides sufficient exploration to eventually sample near any feasible state.

Sound Robustness Bounds. By Lemma 3.4.2, the AGM robustness intervals computed by IRTM (Algorithm 1) are sound: for any completion $s \in \mathcal{C}(s_{t_0, t'})$, we have $\eta(s, \varphi) \in$

$[\eta]_{s_{t'}, \varphi}$. This ensures that Algorithm 4 correctly identifies when a partial trajectory can potentially reach positive AGM robustness (Line 10), preventing premature pruning of feasible paths.

Given sufficient iterations, the stochastic exploration guarantees that feasible regions are eventually sampled, and sound robustness bounds ensure these samples are correctly identified and added to the tree. By standard probabilistic arguments for sampling-based planning (Karaman and Frazzoli, 2011), the probability of finding a feasible solution converges to 1 as $n \rightarrow \infty$. \square

4.8.2 Asymptotic Optimality

Theorem 4.8.2 (Asymptotic Optimality of RRT ^{η}). *Under the conditions of Theorem 4.8.1, let $\varphi^* = \arg \max_{\varphi \in \mathcal{G}} \eta(s, \varphi)$ denote the optimal solution to Problem 4.3.1, where s is the signal corresponding to trajectory ξ in φ^* . Let $\varphi_n = (\mathbf{u}_n, \xi_n)$ denote the control-trajectory pair returned by RRT ^{η} after n iterations, with corresponding signal s_n . Then for any $\varepsilon > 0$:*

$$\lim_{n \rightarrow \infty} \mathbb{P}(\eta(s_n, \varphi) \geq \eta(s^*, \varphi) - \varepsilon) = 1 \quad (4.21)$$

Proof Sketch. The proof builds on probabilistic completeness (Theorem 4.8.1) and establishes convergence to optimality through three key observations:

Monotonic Refinement. By Theorem 3.4.1, as partial trajectories extend, AGM robustness intervals satisfy $[\eta]_{s_{t'}, \varphi} \subseteq [\eta]_{s_t, \varphi}$ for $t' > t$. This monotonicity ensures that longer trajectories provide increasingly precise robustness estimates, allowing the algorithm to systematically identify and pursue high-robustness paths through the rewiring mechanism (Lines 20-23 of Algorithm 3).

Convergence to Exact Values. By Corollary 3.4.1, when a trajectory reaches $t' \geq \|\varphi\|$, the interval converges to exact AGM robustness: $[\eta]_{s_{t'}, \varphi} = \{\eta(s, \varphi)\}$. This enables accurate comparison and selection of optimal solutions.

Rewiring with AGM Cost. The rewiring procedure in Algorithm 4 uses AGM robustness as the cost metric, selecting parent connections that maximize $\underline{\eta}$ (the lower bound of the robustness interval) while maintaining formula consistency (Line 14). Combined with continuous exploration from stochastic mechanisms, this ensures that as $n \rightarrow \infty$, the tree progressively improves toward the optimal AGM robustness value.

The formal proof parallels the RRT* optimality proof (Karaman and Frazzoli, 2011), substituting path length cost with AGM robustness maximization and leveraging Lemma 3.4.2, Theorem 3.4.1, and Corollary 3.4.1.

4.9 Case Studies

We validate RRTⁿ on two robotic systems with increasing complexity: a unicycle mobile robot with nonholonomic constraints, and a 7-DOF manipulator. Each case study demonstrates specific advantages of AGM robustness and FPL-based composition over traditional approaches.

4.9.1 Unicycle Mobile Robot

Consider a unicycle-drive robot operating in a planar environment with sequential visitation requirements and continuous avoidance constraints. The robot has configuration $q = [x, y, \theta]^T \in \mathbb{R}^2 \times [-\pi, \pi]$, where $(x, y) \in \mathbb{R}^2$ is position and $\theta \in [-\pi, \pi]$ is heading angle. The dynamics evolve according to:

$$x_{k+1} = x_k + v_k \cos(\theta_k) \Delta t \quad (4.22)$$

$$y_{k+1} = y_k + v_k \sin(\theta_k) \Delta t \quad (4.23)$$

$$\theta_{k+1} = \theta_k + \omega_k \Delta t \quad (4.24)$$

$$v_{k+1} = u_{1,k}, \quad \omega_{k+1} = u_{2,k} \quad (4.25)$$

where $v_k \in [-0.3, 0.3]$ m/s is translational velocity, $\omega_k \in [-1, 1]$ rad/s is angular velocity, $\mathbf{u}_k = (u_{1,k}, u_{2,k})$ are the control inputs, and $\Delta t = 0.1$ s. The augmented state space is $\mathbf{x} = (x, y, \theta, v, \omega)^T \in \mathbb{R}^5$.

Environment and Specification

The planar workspace $\mathscr{W} = [0, 4] \times [0, 4]$ m² contains three rectangular regions:

- Region 1 (initial target): $x \in [2.0, 3.0]$, $y \in [1.0, 2.0]$ m

- Region 2 (final target): $x \in [0.5, 1.5]$, $y \in [2.5, 3.0]$ m
- Obstacle region (forbidden): $x \in [0.5, 1.5]$, $y \in [1.0, 2.0]$ m

The obstacle is positioned directly between the two target regions, requiring the robot to navigate around it. The task requires sequential region visitation with temporal ordering and continuous obstacle avoidance:

$$\varphi_{unicycle} := \mathbf{F}_{[0,15]}(\mu_{Region1}) \wedge \mathbf{F}_{[15,40]}(\mu_{Region2}) \wedge \mathbf{G}_{[0,20]}(\mu_{avoid}) \quad (4.26)$$

The predicates are:

$$\mu_{Region1} := (2.0 \leq x \leq 3.0) \wedge (1.0 \leq y \leq 2.0) \quad (4.27)$$

$$\mu_{Region2} := (0.5 \leq x \leq 1.5) \wedge (2.5 \leq y \leq 3.0) \quad (4.28)$$

$$\mu_{avoid} := (x < 0.5) \vee (x > 1.5) \vee (y < 1.0) \vee (y > 2.0) \quad (4.29)$$

Results and Analysis

Figure 4-2 shows performance comparison between traditional STL-RRT* using standard robustness and RRT ^{η} with both composition approaches. Traditional robustness demonstrates catastrophic failure: even after normalization from $[-4, 4] \rightarrow [-1, 1]$ to account for different robustness scales, the lower bound remains persistently negative ($\underline{\eta} \approx -0.35$), indicating the planner cannot discover any trajectory satisfying the specification. The upper bound reaches only $\bar{\eta} \approx 0.05$, and the gap plateaus at approximately 0.4, confirming the planner is trapped exploring infeasible state space.

The fundamental limitation of traditional robustness in this scenario stems from its min-max semantics. When evaluating paths through the temporal sequence, traditional robustness takes the minimum over all time steps, causing any momentary low-robustness configuration (such as when navigating near the obstacle boundary) to dominate the entire trajectory evaluation. This pessimistic semantics prevents the planner from recognizing

that brief proximity to constraint boundaries can be acceptable if compensated by high robustness elsewhere.

In stark contrast, both AGM-based methods successfully find high-quality solutions. The stochastic composition (choose-blend, red curves) achieves $\underline{\eta} \approx 0.93$ and $\bar{\eta} \approx 0.90$ with gap approximately 0.1, while FPL-based composition (green curves) reaches $\underline{\eta} \approx 0.95$ and $\bar{\eta} \approx 0.98$ with gap less than 0.05. The additive structure of AGM robustness enables both methods to balance constraint satisfaction across the trajectory: states with moderate robustness near the obstacle can be accepted if they enable high-robustness states in the target regions, and this trade-off is quantified explicitly through AGM’s aggregation mechanism.

Among AGM methods, FPL demonstrates superior efficiency through gradient-based objective balancing. When evaluating competing subformulae during tree expansion, such as whether to prioritize reaching Region 1 or maintaining obstacle clearance during time $t \in [2, 7]$, FPL computes fulfillment values f_i from current AGM robustness intervals (equation (4.12)). These fulfillment values indicate how well each subformula is currently satisfied. FPL then uses power mean derivatives (equation (4.15)) to compute weights that naturally prioritize less-fulfilled objectives, guiding exploration toward states that balance all specification requirements rather than optimizing only the most critical constraint.

The gap metric (panel c) quantifies this difference: FPL reaches gap less than 0.05 within 400 iterations (approximately 17 seconds), while choose-blend achieves gap approximately 0.1 at 800 iterations (approximately 35 seconds), demonstrating FPL’s $2\times$ computational advantage among AGM methods. The tree visualization (panel d) shows the final solution path with temporal progression indicated by color (blue to yellow to red), successfully visiting Region 1, navigating around the obstacle via an upper arc, and terminating in Region 2.

4.9.2 7-DOF KUKA iiwa Manipulator

We evaluate RRT ^{η} on a 7-DOF KUKA iiwa manipulator operating in a constrained workspace with multiple target regions and an obstacle. This scenario presents a cascading choice problem where the planner must make sequential decisions about which regions to visit while satisfying temporal constraints and continuous safety requirements.

System Model and Workspace

The robot has joint configuration $q = (q_1, \dots, q_7) \in \mathcal{Q} \subset \mathbb{R}^7$ with joint limits $q_i \in [q_i^{\min}, q_i^{\max}]$. The end-effector pose lies in $SE(3)$, parametrized locally by position $\mathbf{p} = (x, y, z) \in \mathbb{R}^3$ and orientation represented by Euler angles $\mathbf{O} = (\psi_r, \psi_p, \psi_y) \in [-\pi, \pi]^3$. The forward kinematics mapping is:

$$\mathcal{F} : \mathcal{Q} \rightarrow \mathcal{W} \subseteq SE(3), \quad \mathcal{F}(q) = (\mathbf{p}, \mathbf{O}) \quad (4.30)$$

To facilitate STL specification in task-relevant workspace coordinates while maintaining computational efficiency, we employ an augmented state representation:

$$\mathbf{x} = (q_1, \dots, q_7, x, y, z, \psi_r, \psi_p, \psi_y)^T \in \mathbb{R}^{13} \quad (4.31)$$

The workspace $\mathcal{W} \subset \mathbb{R}^3$ contains four target regions defined as 3D bounding boxes and one spherical obstacle:

- Region A (yellow): $x \in [0.40, 0.55]$, $y \in [0.15, 0.30]$, $z \in [0.50, 0.65]$ m
- Region B (cyan): $x \in [-0.10, 0.10]$, $y \in [0.25, 0.40]$, $z \in [0.55, 0.70]$ m
- Region D (blue): $x \in [0.45, 0.60]$, $y \in [-0.10, 0.10]$, $z \in [0.65, 0.80]$ m
- Region E (magenta): $x \in [-0.15, 0.15]$, $y \in [0.35, 0.50]$, $z \in [0.75, 0.90]$ m
- Obstacle (sphere): center at $(0.20, 0.30, 0.60)$ m, radius $r_{obs} = 0.12$ m

Task Specification and Cascading Choices

Each target region is formalized as a workspace predicate. For example, Region A:

$$\mu_A := (x \in [0.40, 0.55]) \wedge (y \in [0.15, 0.30]) \wedge (z \in [0.50, 0.65]) \quad (4.32)$$

The obstacle avoidance constraint ensures safe clearance with safety margin $d_{safe} = 0.03$ m:

$$\mu_{obs-free} := \sqrt{(x - 0.20)^2 + (y - 0.30)^2 + (z - 0.60)^2} \geq r_{obs} + d_{safe} \quad (4.33)$$

The task requires sequential region visitation with temporal constraints:

$$\varphi_{KUKA} := \mathbf{F}_{[2,7]}(\mu_A \vee \mu_B) \wedge \mathbf{F}_{[8,15]}(\mu_D \vee \mu_E) \wedge \mathbf{G}_{[0,15]}(\mu_{obs-free}) \wedge \mathbf{G}_{[0,15]}(\mu_{joint-limits}) \quad (4.34)$$

where $\mu_{joint-limits} := \bigwedge_{i=1}^7 (q_i \in [q_i^{min}, q_i^{max}])$ enforces joint limits.

This specification creates a complex decision-making scenario with four possible solution paths: A-then-D, A-then-E, B-then-D, or B-then-E. However, not all paths are feasible due to kinematic constraints and obstacle placement. The planner must choose between visiting Region A (0.15 m wide box on the right) or Region B (0.20 m wide box in the center) during the first time window $t \in [2, 7]$ s, and this choice constrains subsequent options for visiting Region D (0.15 m wide box on the right) or Region E (0.30 m wide box in the center) during $t \in [8, 15]$ s.

Augmented State Representation and IK-Based Sampling

The augmented state representation (4.31) provides crucial computational advantages for this high-dimensional problem. Rather than sampling blindly in the 7D joint space \mathcal{Q} , we employ task-space sampling with inverse kinematics (IK):

1. Sample candidate workspace poses $\mathbf{w} = (\mathbf{p}, \mathbf{O})$ directly within target region bounds

(for example, uniformly in $[0.40, 0.55] \times [0.15, 0.30] \times [0.50, 0.65]$ for Region A)

2. Solve IK to compute joint configurations: $q = \mathcal{F}^{-1}(\mathbf{w})$
3. Construct augmented state $\mathbf{x} = [q, \mathbf{w}]$ caching workspace coordinates

This approach offers two significant benefits. First, sampling is biased toward task-relevant workspace regions rather than the larger joint space, dramatically improving the probability of generating states that satisfy the specification. Second, forward kinematics computations during robustness evaluation are eliminated since workspace coordinates are pre-computed and cached in \mathbf{x} . For each state \mathbf{x} , predicate evaluation $\eta(s_t, \mu_{region})$ reduces to comparing cached (x, y, z) values against region bounds, requiring simple arithmetic operations rather than expensive trigonometric computations.

For the KUKA iiwa with $n = 7$ joints, forward kinematics computation via Denavit-Hartenberg transformations takes approximately 50 microseconds while cached lookups require less than 1 microsecond, yielding a $50\times$ speedup in predicate evaluation. Combined with FPL's faster convergence (requiring approximately 60% fewer iterations to achieve gap less than 0.1 compared to traditional methods), the overall computational improvement is substantial: planning times are reduced from approximately 180 seconds to approximately 70 seconds for convergence to high-quality solutions.

For tight workspace constraints, the efficiency gain is even more dramatic. Consider a workspace constraint defining a spherical region of radius $r = 0.1$ m within workspace volume $V_{\mathcal{W}} \approx 4$ m³. The constraint volume is $V_{const} = \frac{4}{3}\pi r^3 \approx 0.0042$ m³, yielding acceptance probability $p_{acc} \approx V_{const}/V_{\mathcal{W}} \approx 0.001$. Rejection sampling in configuration space requires an expected $1/p_{acc} \approx 1000$ attempts per successful sample, each costing $O(n)$ for forward kinematics computation, giving total expected cost of $O(1000n)$ per successful sample. IK-based sampling achieves IK success rate approximately $p_{IK} \approx 0.9$ for well-designed workspace constraints, requiring expected $1/p_{IK} \approx 1.1$ attempts per successful sample.

With caching at rate $\rho = 0.9$, the cost per attempt is $O(0.1n^3)$, yielding total expected cost of $O(0.11n^3)$ per successful sample. The speedup factor is approximately $185\times$ for $n = 7$ joints.

Results and Analysis

Figure 4.3 shows performance comparison across different STL-RRT* variants. Traditional robustness (blue curves) demonstrates the undirected exploration characteristic of min-max semantics: after normalization to account for different robustness scales ($[-6.3, 6.3] \rightarrow [-1.0, 1.0]$), the lower bound plateaus at $\underline{\eta} \approx -0.35$ and upper bound at $\bar{\eta} \approx 0.08$, with gap approximately 1.2 indicating continued exploration of low-quality solutions.

Traditional STL-RRT* must explore all four region pairs with roughly equal priority, as min-max semantics provide limited guidance about which choices lead to higher overall robustness. In contrast, both AGM-based methods leverage the additive structure of AGM robustness to systematically resolve the choice dilemma. Rather than treating each disjunction as an isolated decision, AGM robustness accumulates contributions from both branches, allowing the planner to recognize that certain region pairs maintain consistently higher robustness throughout the trajectory.

Both AGM methods demonstrate structured convergence: choose-blend (red curves) achieves $\underline{\eta} \approx 0.60$ and $\bar{\eta} \approx 0.80$ with gap approximately 0.8, while FPL (green curves) reaches $\underline{\eta} \approx 0.95$ and $\bar{\eta} \approx 0.98$ with gap less than 0.1, both substantially outperforming the traditional baseline.

While both AGM methods vastly outperform traditional robustness, FPL demonstrates superior efficiency through gradient-based objective balancing. When evaluating competing subformulae during tree expansion—such as whether to prioritize reaching Region A or Region B during $t \in [2, 7]$ —FPL computes fulfillment values f_i from the current AGM robustness intervals (Equation 4.12). These fulfillment values indicate how well each subformula is currently satisfied. FPL then uses power mean derivatives (Equation 4.15) to

compute weights w_i that naturally prioritize less-fulfilled objectives, guiding exploration toward states that balance all specification requirements rather than optimizing only the most critical constraint.

The gap metric (panel c) quantifies this difference: FPL reaches gap less than 0.1 within 1000 iterations (approximately 70 seconds), whereas choose-blend requires approximately 2500 iterations (approximately 100 seconds) to achieve gap approximately 0.2, demonstrating FPL’s $1.5\times$ computational advantage among AGM methods.

Figures 4.3(d-f) visualize the final reference trajectory from multiple viewpoints using a ghost trail representation with 10 time-sampled configurations where opacity indicates temporal progression (transparent to opaque). The robot successfully visits Region A (yellow) during the first temporal window and terminates at Region E (magenta) in the second window, maintaining safe clearance from the obstacle sphere throughout execution. The smooth progression of overlaid poses demonstrates feasible motion respecting joint velocity constraints, with the final fully-opaque pose confirming specification satisfaction at $t = 15$ s.

4.10 Summary

This chapter presented RRT^η , our sampling-based motion planning framework that synthesizes robot control sequences satisfying STL specifications with maximum AGM robustness. Building on the quantitative semantics and runtime monitoring from Chapter 3, we developed Direction of Increasing AGM Satisfaction vectors that leverage smooth gradients from holistic robustness evaluation. We introduced principled multi-objective composition using Fulfillment Priority Logic, providing gradient-based balancing of competing specification requirements with theoretical minimum fulfillment guarantees.

We proved that RRT^η maintains probabilistic completeness and asymptotic optimality despite the more complex AGM robustness measure. Experimental validation on unicycle

and manipulator systems demonstrated substantial advantages over traditional approaches: AGM-based methods discovered high-quality solutions where traditional robustness failed entirely, and FPL-based composition achieved $1.5\text{-}2\times$ computational speedup through forward prediction and principled objective balancing.

The integration of AGM robustness with sampling-based planning addresses fundamental limitations of min-max semantics, enabling more effective synthesis for complex temporal specifications in high-dimensional spaces. The next chapter extends these techniques to reinforcement learning settings with delayed rewards. In our future work, we consider incremental planning for TWTL tasks for which we require runtime monitors to use as a heuristic to maximize the satisfaction of the task.

Algorithm 3: RRT $^\eta$ Algorithm

```

1 Input:  $q_{init}$  – Initial configuration
2 Input:  $\phi$  – STL formula in positive normal form
3 Output:  $u$  – a satisfying control policy w.r.t.  $\phi$  with maximum AGM
  robustness
4  $\mathcal{T} = (V := \emptyset, E = \emptyset)$ 
5  $V \leftarrow (v.q_{init} \leftarrow q_{init}, v.\phi \leftarrow \phi, v.t \leftarrow 1, v.control\_StateTraj.\xi \leftarrow$ 
    $\emptyset, v.parent \leftarrow \emptyset, v.ch \leftarrow \emptyset)$ 
6 for  $k = 1 : N^{max}$  do
7    $t^r, q^r \leftarrow \text{sample}(\mathcal{Q}, \mathcal{T}, \phi)$ 
8    $\mathcal{N} \leftarrow \text{near}(\mathcal{T}, q^r, t^r)$ 
9    $\lambda \leftarrow \text{Unif}([0, 1])$ 
10   $v.parent \leftarrow \emptyset, J^* \leftarrow \infty, q^* \leftarrow \emptyset$ 
11  foreach  $v' \in \mathcal{N}$  do
12     $\Delta t^r = t^r - v'.t$ 
13    // Compute DIAS vector using AGM robustness intervals
14     $d_\chi \leftarrow \chi_\eta(v'.control\_stateTraj.\xi, v'.\phi, v'.t)$ 
15    // Compute optimal control using DIAS and random sample
16     $u^* \leftarrow \arg \min_{u \in \mathcal{U}} J_\chi(u, \text{steer}(v.q, u, \Delta t^r), v.q, v.\phi, q^r, \Delta t^r, \lambda)$ 
17     $q^s \leftarrow \text{steer}(v'.q, u^*, \Delta t^r)$ 
18     $J^s \leftarrow J_\xi(u^*, q^s, v'.q, v'.\phi, q^r, d_\eta, \lambda)$ 
19    if  $J^s < J^* \wedge \text{steer\_exct}(v'.q, q^s)$  then
20       $J^* \leftarrow J^s, v.parent \leftarrow v', q^* \leftarrow q^s$ 
21   $v_{temp}.q \leftarrow q^*, v_{temp}.\phi \leftarrow \emptyset, v_{temp}.t \leftarrow \emptyset, v_{temp}.control\_StateTraj \leftarrow$ 
    $\emptyset, v_{temp}.parent \leftarrow \emptyset, v_{temp}.ch \leftarrow \emptyset, v_{temp}[\eta] \leftarrow [-1, 1]$ 
22  // Update with AGM robustness interval calculation
23   $\text{update}_\eta(v.parent, v_{temp})$ 
24  for  $v'' \in \text{Near}(\mathcal{T}, q^*, t^r)$  do
25    if  $\text{steer\_exct}(q^*, q'')$  then
26      // Rewiring based on AGM robustness intervals  $\text{update}_\eta(v_{temp}, v'')$ 
27  if  $\text{existsSolutionAGM}()$  then
28     $v_{best} = \text{bestAGM}(\mathcal{V})$ 
29    return  $v_{best}.control\_StateTraj$ 
30  else
31    return  $\emptyset$ 

```

Algorithm 4: $\text{update}_\eta(v_1, v_2)$

```

1 // Initialize AGM robustness interval with parent's interval
2  $[\eta]_{\text{curr}} \leftarrow v_1.[\eta]$ 
3  $t_{\text{curr}} \leftarrow v_1.t$ 
4 // Get trajectory points from steering function
5  $(u_{1,2}, \xi_{1,2}) \leftarrow \text{steer\_exct}(v_1.q, v_2.q)$ 
6 // Incrementally compute AGM robustness interval for each observation
7 for  $i = 1$  to  $n$  do
8    $[\eta]_{\text{curr}} \leftarrow \text{IRTM}(v_1.\phi, [\eta]_{\text{curr}}, \xi_{1,2}(i), t_{\text{curr}})$ 
9    $t_{\text{curr}} \leftarrow t_{\text{curr}} + 1$ 
10  $[\underline{\eta}'_2, \overline{\eta}'_2] \leftarrow [\eta]_{\text{curr}}$ 
11 if  $\phi_2 = \emptyset$  then
12   if  $\overline{\eta}'_2 \geq 0$  then
13      $v_2.[\eta] \leftarrow [\underline{\eta}'_2, \overline{\eta}'_2]$ 
14      $v_2.\phi \leftarrow \text{simplify}(v_1.\phi)$ 
15      $\mathcal{V} \leftarrow \mathcal{V} \cup \{v_2\}, \mathcal{E} \leftarrow \mathcal{E} \cup \{(v_1, v_2)\}$ 
16 else if  $\overline{\eta}'_2 \geq 0 \wedge \underline{\eta}'_2 \geq \min(v_2.[\eta]) \wedge \phi_1 \Rightarrow \phi_2$  then
17    $v_2.[\eta] \leftarrow [\underline{\eta}'_2, \overline{\eta}'_2]$ 
18    $\mathcal{E} \leftarrow (\mathcal{E} \setminus \{(v_2.\text{parent}, v_2)\}) \cup \{(v_1, v_2)\}$ 
19    $V_{\text{upd}} = \text{ch}(v_2)$  // children of  $v_2$ 
20   while  $V_{\text{upd}} \neq \emptyset$  do
21      $v \leftarrow V_{\text{upd}}.\text{pop}(), v' \leftarrow v.\text{parent}$ 
22     // Recompute intervals for affected branches recursively
23      $[\eta]_{\text{curr}} \leftarrow v'.[\eta]$ 
24      $t_{\text{curr}} \leftarrow v'.t$ 
25      $(u_{v',v}, \xi_{v',v}) \leftarrow \text{steer\_exct}(v'.q, v.q)$ 
26     for  $i = 1$  to  $|\xi_{v',v}|$  do
27        $[\eta]_{\text{curr}} \leftarrow \text{IRTM}(v'.\phi, [\eta]_{\text{curr}}, \xi_{v',v}(i), t_{\text{curr}})$ 
28        $t_{\text{curr}} \leftarrow t_{\text{curr}} + \Delta t$ 
29      $v.[\eta] \leftarrow [\eta]_{\text{curr}}$ 

```

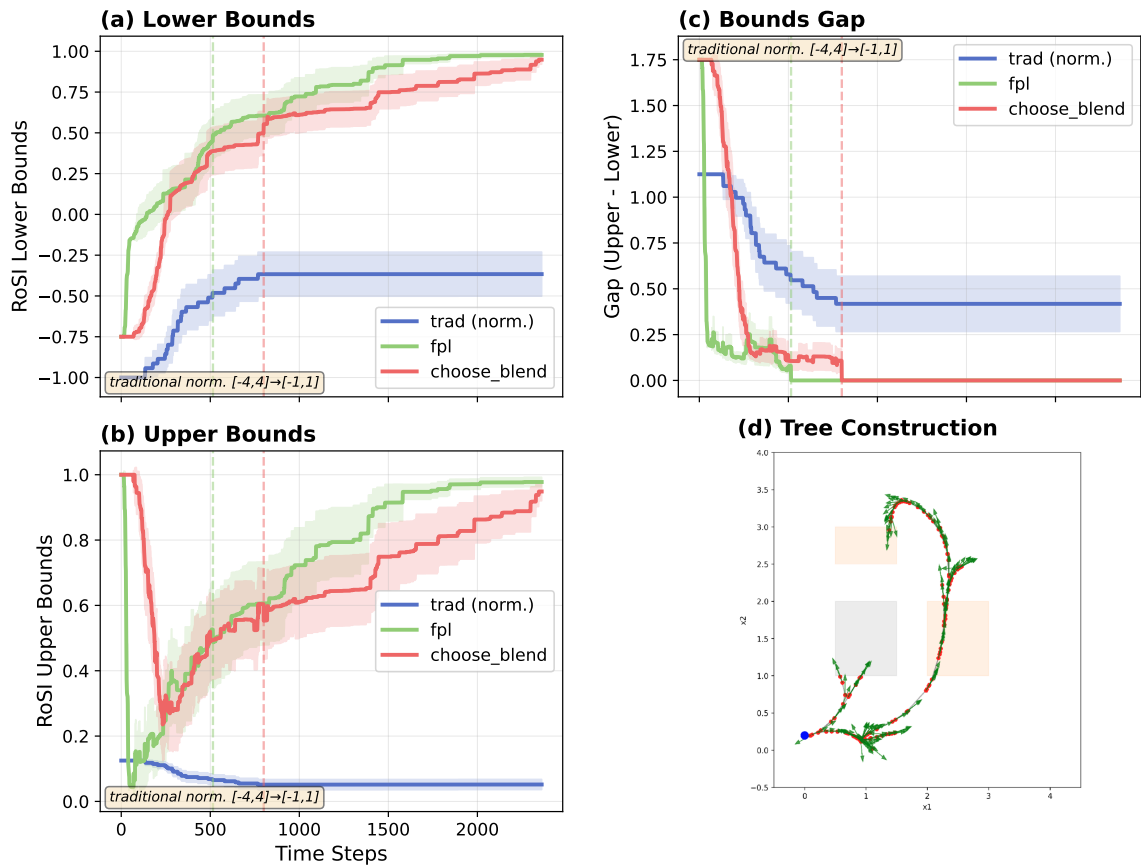


Figure 4-2: Performance comparison on unicycle sequential reach-avoid task. (a) Lower bounds: Traditional robustness (blue, normalized) fails with $\underline{\eta} \approx -0.35$; choose-blend (red) achieves $\underline{\eta} \approx 0.93$; FPL (green) reaches $\underline{\eta} \approx 0.95$. (b) Upper bounds show similar trends. (c) Gap metric: FPL converges to gap < 0.05 at 400 iterations, choose-blend at 800 iterations, demonstrating $2\times$ speedup. (d) Tree construction showing successful path (color indicates temporal progression) visiting Region 1, navigating around obstacle via upper arc, and terminating in Region 2.

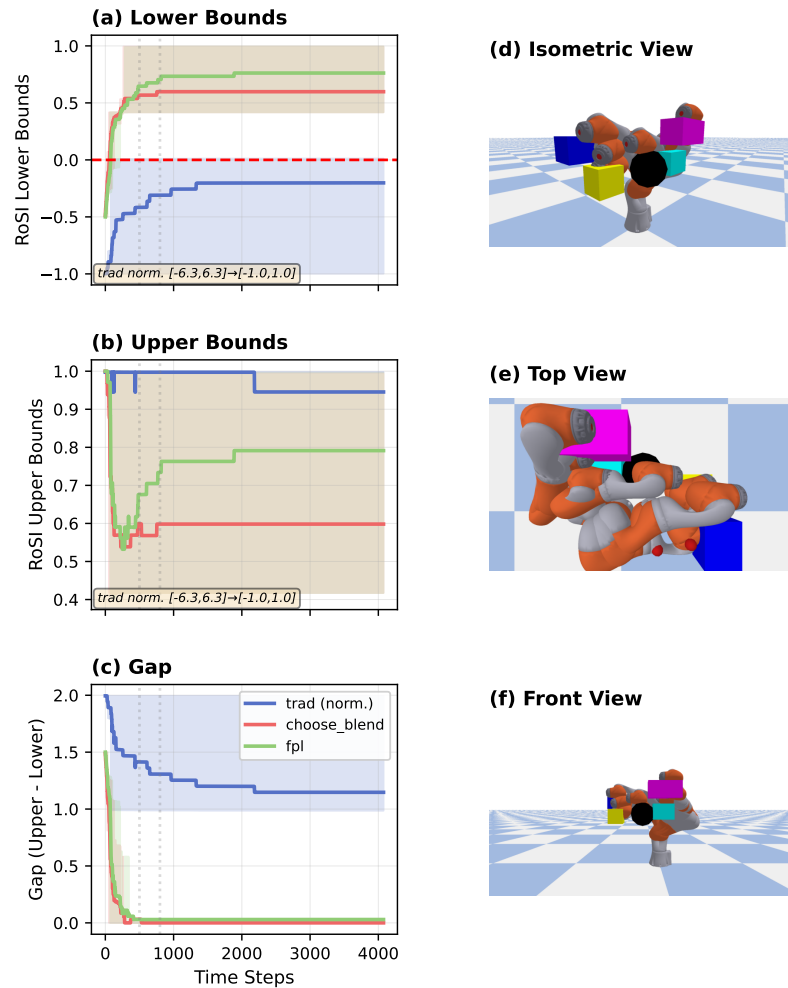


Figure 4-3: Performance comparison on KUKA cascading choice problem. (a) Lower bound evolution: FPL (green) shows rapid, structured convergence; traditional (blue) and choose-blend (red) show slower exploration. (b) Upper bound evolution shows similar trends. (c) Gap metric: FPL reaches near-zero gap (< 0.1) within 1000 iterations; traditional and choose-blend plateau at 1.2 and 0.8 respectively. (d-f) Ghost trail visualization from multiple viewpoints showing reference trajectory with 10 time-sampled configurations (opacity indicates temporal progression). The robot successfully executes the A-then-E path with red spheres marking joint positions throughout motion.

Chapter 5

Accelerated Proximal Policy Optimization with TWTL Reward Shaping

5.1 Introduction

The preceding chapters developed quantitative semantics for temporal logics and demonstrated their application to motion planning. Chapter 3 introduced TWTL robustness measures, both traditional min-max semantics and AGM robustness, with efficient incremental monitoring algorithms. Chapter 4 integrated these concepts into sampling-based motion planning, showing that AGM robustness creates smoother optimization landscapes that improve exploration and solution quality. These developments enable a new application: using TWTL quantitative semantics to address the delayed reward problem in reinforcement learning.

Reinforcement learning in environments with delayed rewards presents a fundamental challenge: actions that lead to successful outcomes may not receive immediate positive feedback, making it difficult for learning algorithms to identify and reinforce beneficial behaviors. This challenge is particularly acute in complex sequential decision-making tasks. For example, in robotic manipulation, the value of intermediate positioning actions may only become apparent after multiple time steps when the final grasp attempt succeeds or fails. Similarly, in the lunar lander domain, maintaining proper hover altitude early in the descent only contributes to mission success through its impact on later alignment and landing phases.

Policy Gradient methods, particularly Proximal Policy Optimization (PPO) (Schulman et al., 2017), have demonstrated remarkable success in reinforcement learning tasks. However, in settings with delayed rewards, even PPO’s carefully constructed optimization landscape becomes difficult to navigate, as the temporal gap between actions and their consequences creates a sparse and uninformative gradient signal.

This chapter presents Accelerated Proximal Policy Optimization (APPO), which addresses the delayed reward challenge through two key innovations that leverage TWTL semantics from Chapter 3. First, we develop a hybrid policy architecture that combines an offline policy (trained on expert demonstrations) with an online PPO policy, maintaining the offline policy as an active guide throughout training rather than using it merely for initialization. Second, we develop a reward shaping mechanism using TWTL robustness that provides immediate, semantically meaningful feedback about progress toward satisfying temporal specifications. The TWTL robustness serves as a potential function, enabling us to prove optimal policy preservation while providing dense reward signals that bridge the temporal gap between actions and outcomes. We prove monotonic improvement over both the offline policy and previous iterations with a bounded performance gap, and demonstrate significant performance improvements on robotic control tasks with delayed rewards.

5.2 Background and Related Work

5.2.1 Policy Gradient Methods

Policy Gradient (PG) methods optimize a parameterized policy by following gradients of expected cumulative reward. Trust Region Policy Optimization (TRPO) (Schulman et al., 2015) introduced theoretical guarantees of monotonic policy improvement by constraining policy updates to remain within a trust region defined by KL divergence. The algorithm optimizes a surrogate objective while ensuring that the new policy does not deviate too far from the current policy, providing stable learning dynamics.

Proximal Policy Optimization (PPO) (Schulman et al., 2017) simplifies TRPO’s implementation while maintaining similar theoretical properties. PPO uses a clipped surrogate objective that limits the magnitude of policy updates, achieving stable learning without expensive second-order optimization. The clipped objective penalizes large policy changes that would move far from the current policy, providing an implicit trust region constraint. This design has made PPO one of the most widely used policy gradient algorithms across diverse domains.

However, both TRPO and PPO face challenges in environments with delayed rewards. The temporal gap between actions and their consequences creates sparse gradient signals that provide little guidance for policy improvement. Early actions in a trajectory may have a significant impact on eventual success or failure, but receive no immediate feedback to guide learning. This credit assignment problem becomes particularly acute when reward horizons span tens or hundreds of time steps, as in many robotic manipulation and navigation tasks.

5.2.2 Temporal Logics in Reinforcement Learning

Temporal logics have been increasingly used in reinforcement learning to specify complex tasks beyond simple reward functions. Most work investigating temporal logics (TL) in RL focuses primarily on task specification (Asarkaya et al., 2021; Cai et al., 2023; Icarte et al., 2022; Xu et al., 2020; Neider et al., 2021; Alshiekh et al., 2018; Balakrishnan and Deshmukh, 2019; Li et al., 2017; Aksaray et al., 2016). TLs formalize high-level tasks into propositional and temporal constraints (Baier and Katoen, 2008), with some logics like Time Window Temporal Logic (TWTL) (Vasile et al., 2017a) naturally handling delayed rewards through explicit temporal constraints.

In (Asarkaya et al., 2021), a Q-learning algorithm learns optimal policies for TWTL-modeled tasks while maximizing external rewards. However, this approach is limited to discrete state-action spaces and does not directly address the credit assignment problem

inherent in delayed reward settings. The work focuses on specification satisfaction rather than using temporal logic semantics to provide intermediate feedback signals.

Recent works like (Cai et al., 2023) and (Icarte et al., 2022) have shown promising results in combining TL with deep RL. Reward machines (Icarte et al., 2022) use automata representations of temporal logic specifications to structure the learning problem, decomposing complex tasks into simpler subtasks. However, these approaches primarily focus on task decomposition rather than providing dense reward signals through quantitative semantics.

Our approach differs fundamentally by actively using TWTL quantitative semantics for reward shaping. By leveraging the TWTL robustness measures developed in Chapter 3, we provide immediate feedback about progress toward satisfying temporal specifications. This creates a dense reward signal that guides learning even when the sparse task reward is delayed, effectively bridging the temporal gap between actions and outcomes.

5.2.3 Offline Reinforcement Learning

Offline RL algorithms learn from fixed datasets of expert demonstrations without online environment interaction during initial training phases. The authors of (Ball et al., 2023; Ravari et al., 2024) developed algorithms demonstrating benefits of learning from demonstrations, but typically treat offline data as a fixed dataset for initialization rather than an active policy component throughout training.

Methods like AWAC (Nair et al., 2020), BCQ (Kumar et al., 2020), and CQL (Kumar et al., 2020) use offline data to initialize policies or constrain policy updates to remain close to the data distribution. These approaches help with sample efficiency and provide reasonable starting policies, but do not maintain the offline policy as an active component that continuously guides online learning.

In (Hu et al., 2023), the policy chooses between an imitation learning (IL) policy trained offline and an online RL policy based on which action has a higher Q-value. This represents

a form of policy combination, but the selection mechanism is deterministic and based on value estimates that may be inaccurate early in training.

Our work differs in that we treat the mixing mechanism as a learnable parameter in the context of deep RL. Rather than switching between policies based on value estimates, we learn a continuous mixing parameter α that gradually shifts from relying on the offline policy to trusting the online policy as learning progresses. We prove that maintaining the offline policy as an active component guarantees monotonic improvement over both the offline policy and previous policy iterations. This is crucial in delayed-reward settings where the online policy may initially perform poorly due to insufficient exploration, and the offline policy provides a safety net that guides learning toward promising regions of the state space.

5.2.4 Reward Shaping with TWTL Semantics

Reward shaping has been widely studied as a method to accelerate RL by providing additional feedback signals beyond the sparse task reward (Ng et al., 1999). The potential-based reward shaping framework (Ng et al., 1999) provides a critical theoretical guarantee: if the shaping function can be expressed as a difference of a potential function at different states, the optimal policy of the original problem is preserved. This ensures that the additional reward signals guide learning without fundamentally changing what constitutes optimal behavior.

Previous work has explored reward shaping from temporal logic specifications (Aksaray et al., 2016; Balakrishnan and Deshmukh, 2019), but typically in discrete settings or without formal guarantees of optimal policy preservation. Most temporal logic approaches focus on Boolean satisfaction rather than quantitative robustness, providing binary feedback (satisfied or violated) rather than continuous measures of progress.

Chapter 3 developed quantitative semantics for TWTL, introducing robustness measures that quantify the degree of satisfaction or violation of a specification. The traditional

TWTL robustness ρ uses min-max operations to identify the most critical constraint, while AGM robustness η aggregates satisfaction across all time points and subformulae using arithmetic and geometric means. Both measures provide continuous values indicating how strongly a trajectory satisfies a specification, with positive values indicating satisfaction and negative values indicating violation.

These quantitative semantics enable a natural approach to reward shaping: we can use TWTL robustness as a potential function to define shaped rewards. However, computing TWTL robustness requires access to future observations spanning the formula’s time horizon $\|\phi\|$, which are not available at the current time step during online learning. This necessitates trajectory prediction.

5.2.5 Trajectory Prediction for Reward Shaping

Computing TWTL robustness for reward shaping requires predicting future observations to evaluate specifications with time horizons that extend beyond the current time step. Existing approaches for trajectory completion include trajectory prediction (Salzmann et al., 2020) and the runtime monitoring techniques developed in Chapter 3 (Ahmad et al., 2023).

We employ a Long Short-Term Memory (LSTM)-based trajectory predictor, similar in spirit to Trajectron++ (Salzmann et al., 2020) but adapted for single-agent scenarios. The predictor takes a window of past states and generates predicted future observations spanning the TWTL formula horizon. This enables evaluation of TWTL robustness at each time step, providing immediate feedback about predicted satisfaction or violation of temporal requirements.

The predictor is trained on expert demonstrations to minimize mean squared prediction error, ensuring that predicted trajectories reflect feasible system behavior. During online learning, the predictor is called at each time step to generate trajectory completions, which are then evaluated against the TWTL specification to compute robustness values for reward shaping.

Importantly, the prediction errors do not affect the optimal policy preservation guarantee. As long as the predictor is trained on expert demonstrations (which by definition satisfy the specification), the predicted robustness values maintain the potential function structure necessary for the shaping framework. Even with imperfect predictions, the shaped reward function guides learning toward specification satisfaction while preserving the optimal policy of the original sparse reward problem.

5.2.6 Contributions of This Work

This chapter makes two main contributions that leverage TWTL quantitative semantics from Chapter 3 to address delayed rewards in reinforcement learning:

(1) Hybrid Policy Architecture with Theoretical Guarantees. We develop a policy architecture that combines an offline policy π_ρ (trained on expert demonstrations) with an online PPO policy π_β through a learnable mixing parameter α : $\pi_\theta(u|x) = (1 - \alpha)\pi_\rho(u|x) + \alpha\pi_\beta(u|x)$. Unlike previous approaches that use offline data merely for initialization, we maintain the offline policy as an active component throughout training. We prove monotonic improvement over both the offline policy and previous iterations (Proposition 7 and Theorem 8), with a bounded performance gap of $(2\zeta\gamma\alpha^2)/(1 - \gamma)^2$, where ζ bounds the expected advantage, γ is the discount factor, and α is the mixing parameter. This architecture provides a safety net during exploration while gradually learning to improve upon the offline policy.

(2) TWTL-Based Reward Shaping with Optimality Preservation. We develop a reward shaping mechanism using TWTL robustness as a potential function. Given the TWTL robustness predictor $\text{Pred}(x_t) \rightarrow \hat{\delta}_{t:t+\|\phi\|}$, we define the shaping function: $F(x_t, u_t, x_{t+1}, \phi) := \kappa \cdot (\rho(\text{Pred}(x_t), \phi) - \rho(\text{Pred}(x_{t+1}), \phi))$, where $0 < \kappa < 1$ is a scaling parameter. We prove that this potential-based shaping preserves the optimal policy of the original problem (Lemma 4), ensuring that additional feedback does not fundamentally change optimal behavior. The TWTL robustness provides semantically meaningful

feedback about progress toward specification satisfaction at each time step, bridging the temporal gap in sparse reward signals.

The chapter demonstrates that TWTL quantitative semantics developed for monitoring (Chapter 3) and planning (Chapter 4) extend naturally to reinforcement learning, providing a unified framework for formal methods across diverse robotic applications.

5.3 Problem Formulation

We consider a model-free reinforcement learning setting where an agent interacts with an environment formalized as a Markov Decision Process.

Definition 5.3.1. *A finite horizon MDP is a tuple $(\mathcal{X}, \mathcal{U}, \mathbb{P}(\cdot|\cdot, \cdot), r(\cdot, \cdot), l(\cdot), \mathcal{O})$, where \mathcal{X} , \mathcal{U} and \mathcal{O} are the state, control, and output spaces, respectively; $\mathbb{P}(\cdot|\cdot, \cdot)$ is the state-action pair transition probability; $r : \mathcal{X} \times \mathcal{U} \mapsto [0, 1]$ is the reward function; and $l : \mathcal{X} \mapsto \mathcal{O}$ is a labeling function that maps the state to an output observation.*

5.3.1 Value Functions and Advantages

Let $\pi : \mathcal{X} \rightarrow \mathcal{U}$ be a stochastic policy. In an episodic RL setting (Sutton and Barto, 2018) with K learning episodes, we define the state value function and state-action value function at iteration t of episode k as:

$$V_t^{\pi,k}(x) := \mathbb{E}_{\pi} \left[\sum_{i=t}^N r_i^k(x_i, u_i) \mid x_t = x \right] \quad (5.1)$$

$$Q_t^{\pi,k}(x, u) := \mathbb{E}_{\pi} \left[\sum_{i=t}^N r_i^k(x_i, u_i) \mid x_t = x, u_t = u \right] \quad (5.2)$$

where \mathbb{E}_{π} denotes expectation over the stochastic policy π .

The advantage function quantifies how much better or worse a specific action is compared to the policy's average action:

$$A_t^{\pi,k}(x, u) := Q_t^{\pi,k}(x, u) - V_t^{\pi,k}(x) \quad (5.3)$$

A positive advantage indicates that action u is better than π 's average action in state x , while a negative advantage suggests it is worse. This function plays a crucial role in policy gradient methods by identifying which actions to encourage or discourage during policy updates.

5.3.2 Concrete-Time Rewards from TWTL Specifications

To address the temporal credit assignment challenge in delayed reward settings, we introduce a reward function based on TWTL specifications that provides immediate feedback about task progress. Recall from Section 2.2 the TWTL syntax and Boolean semantics.

Definition 5.3.2 (Concrete-Time Reward). *Let $\phi \in \Phi_{TWTL}$ be a feasible TWTL formula (Definition 2.2.2). For a finite-horizon MDP trajectory $\mathbf{x}_{i,i+\|\phi\|}$ produced by applying control sequence $\mathbf{u}_{i-1,i+\|\phi\|-1} := u_{i-1}u_i \cdots u_{i+\|\phi\|-1}$, we define an episodic concrete-time reward over the generated observation word $\mathbf{o}_{i,i+\|\phi\|}$ at time t as:*

$$r_{\phi,t}^k(\mathbf{o}_{t,t+\|\phi\|}) := \begin{cases} 1 & \text{if } \mathbf{o}_{t,t+\|\phi\|} \models \phi \\ 0 & \text{if } \mathbf{o}_{t,t+\|\phi\|} \not\models \phi \end{cases} \quad (5.4)$$

where k denotes the training episode and t denotes the time instance.

This reward formulation creates a fundamental challenge: computing $r_{\phi,t}^k$ requires a complete trajectory spanning the entire task horizon $\|\phi\|$, but during online learning, we must make decisions at each time step without access to future states. This temporal gap between action selection and reward observation is precisely the delayed reward problem we aim to address.

Problem 5.3.1 (Optimal Policy Learning with Delayed Rewards). *Given a finite-horizon MDP with episodic concrete-time reward $r_{\phi,t}^k$, compute the optimal parameter θ^* of the stochastic policy π_θ that maximizes the total expected episodic reward:*

$$\pi_{\theta^*} = \arg \max_{\theta \in \Theta} \mathbb{E}_{\pi_\theta} \left[\sum_{t=1}^N r_{\phi,t}^k \right] \quad (5.5)$$

Problem 5.3.1 is challenging because the concrete-time reward provides feedback only after entire task sequences complete, creating a sparse and delayed reward signal that makes gradient-based optimization difficult.

5.4 Background: Proximal Policy Optimization

Before presenting our enhancements, we briefly review the Proximal Policy Optimization algorithm that forms the foundation of our approach.

PPO is a policy optimization algorithm that uses policy gradients to optimize a parameterized policy while providing stability through constrained policy updates (Schulman et al., 2017). At each iteration, PPO aims to find an improved policy that remains close to the previous iteration, preventing degenerate policy updates that can destabilize learning.

For a parameterized policy π_θ where $\theta \in \Theta$ is the parameter vector, PPO optimizes the policy at each iteration according to:

$$\theta_{i+1} \leftarrow \arg \max_{\theta \in \Theta} \mathbb{E}[J^k(x, u, \theta, \theta_i)] \quad (5.6)$$

where $J^k(x, u, \theta, \theta_i)$ is a clipped surrogate objective:

$$J^k(x, u, \theta, \theta_i) := \min \left[\frac{\pi_\theta(u|x)}{\pi_{\theta_i}(u|x)} \hat{A}_t^{\pi_{\theta_i}, k}(x, u), g(\varepsilon, \hat{A}_t^{\pi_{\theta_i}, k}(x, u)) \right] \quad (5.7)$$

The clipping function g is defined as:

$$g(\varepsilon, \hat{A}) := \begin{cases} (1 + \varepsilon)\hat{A} & \text{if } \hat{A} \geq 0 \\ (1 - \varepsilon)\hat{A} & \text{if } \hat{A} < 0 \end{cases} \quad (5.8)$$

where $\varepsilon \in (0, 1)$ is a hyperparameter controlling the trust region size.

The advantage estimate $\hat{A}_t^{\pi_{\theta_i}, k}$ is computed using Generalized Advantage Estimation

(GAE) (S. et al., 2015):

$$\hat{A}_t^{\pi_{\theta_i},k} = \delta_t^k + (\gamma\lambda)\delta_{t+1}^k + \dots + (\gamma\lambda)^{N-t+1}\delta_{N-1}^k \quad (5.9)$$

where $\delta_t^k := r_t^k + \gamma V_t^{\pi_{\theta_i},k} - V_{t+1}^{\pi_{\theta_i},k}$ is the temporal difference residual, $\gamma \in (0, 1)$ is the discount factor, and $\lambda \in (0, 1)$ controls the bias-variance trade-off.

The choice of maximizing the clipped objective (5.7) provides several benefits. First, it ensures stable policy updates by preventing large policy changes. Second, maximizing this objective is equivalent to maximizing the advantage function, which yields low-variance gradients. Third, the advantage function directly measures how much better or worse the policy is compared to the baseline, providing clear direction for improvement.

5.5 Hybrid Policy Architecture

We now present our first main contribution: a hybrid policy architecture that combines an offline policy trained on expert demonstrations with an online policy optimized through PPO. Unlike previous approaches that use offline data merely for initialization, our method maintains the offline policy as an active component throughout training.

5.5.1 Policy Mixing Architecture

Let $\mathcal{D}_{\mathcal{U}}$ denote the set of probability distributions over the control space \mathcal{U} . We introduce a deep policy architecture $\pi_{\theta} \in \mathcal{D}_{\mathcal{U}}$ that combines two parallel policies. The first component is an offline policy $\pi_{\rho} \in \mathcal{D}_{\mathcal{U}}$ trained on expert demonstrations, with fixed parameters ρ that remain constant throughout training. The second component is an online policy $\pi_{\beta} \in \mathcal{D}_{\mathcal{U}}$ that is continuously optimized through PPO, with learnable parameters β that adapt based on experience.

These policies are combined using a fully connected layer (FCL) with learnable mixing

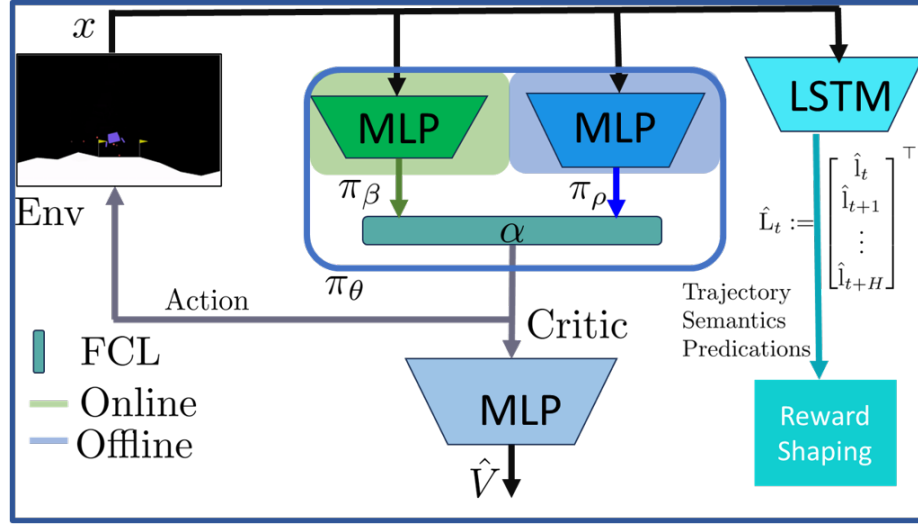


Figure 5-1: The Actor-Critic RL framework. The Actor’s architecture, policy π_θ , consists of an offline policy, π_ρ , and an adaptive policy, π_β , where the two policies are mixed using the parameters of the FCL, α . The critic consists of an MLP that approximates the value function. The task predictor LSTM network and the reward shaping are depicted in cyan.

parameter $\alpha \in [0, 1]$:

$$\pi_\theta(u|x) := (1 - \alpha) \cdot \pi_\rho(u|x) + \alpha \cdot \pi_\beta(u|x) \quad (5.10)$$

where the overall parameter is $\theta = (\rho, \beta, \alpha)$. See Figure 5-1, where we depict the full architecture.

The mixing parameter α controls the relative influence of the offline and online policies. When $\alpha \approx 0$, the hybrid policy relies primarily on expert demonstrations; when $\alpha \approx 1$, it relies primarily on learned online behavior. During training, α is learned alongside β , allowing the algorithm to automatically adjust the balance based on the relative quality of the two policies.

We interpret π_ρ as a fixed prior that guides exploration, while π_β and α are optimized to improve performance. This architecture provides several advantages over using offline data only for initialization: the offline policy continues to stabilize learning throughout training, poor initial online policies can still benefit from expert guidance, and the learned mixing

parameter automatically balances exploration and exploitation.

5.5.2 Performance Improvement Guarantees

We now establish theoretical guarantees that the hybrid policy improves upon both the offline policy and previous iterations. Our analysis builds on the policy improvement framework from Trust Region Policy Optimization (TRPO) (Schulman et al., 2015).

The total return of policy π at episode k is:

$$\eta^k(\pi) := \mathbb{E}_{u \sim \pi} \left[\sum_{i=1}^N r_i^k(x_i, u_i) \right] \quad (5.11)$$

We drop the superscript k when the episode is clear from context.

Lemma 5.5.1 (Policy Improvement Identity). *For our mixed policy π_θ and offline policy π_ρ , the return difference can be expressed as:*

$$\eta(\pi_\theta) = \eta(\pi_\rho) + \mathbb{E}_{u \sim \pi_\theta} \left[\sum_{i=1}^N A_i^{\pi_\rho} \right] \quad (5.12)$$

Proof. This follows directly from Lemma 1 in (Schulman et al., 2015) by replacing their policy π' with our mixed policy π_θ and their baseline policy π with our offline policy π_ρ . The fundamental relationship between advantage functions and expected returns remains unchanged under our policy mixing architecture. \square

Lemma 5.5.1 provides the basic formula for improvement, but the expectation over π_θ makes it difficult to derive practical update rules. We therefore introduce a local approximation of the total return based on the offline policy's state visitation distribution.

Let $P_{\pi_\rho}(x) = \sum_{t=0}^N \mathbb{P}(x_t = x | x_0, u \sim \pi_\rho)$ denote the discounted visitation frequency under π_ρ . The return approximation is:

$$L_{\pi_\rho}(\pi) := \eta(\pi_\rho) + \sum_{x \in \mathcal{X}} P_{\pi_\rho}(x) \sum_{u \in \mathcal{U}} \pi(u|x) A_0^{\pi_\rho}(x, u) \quad (5.13)$$

For analysis, we define the optimal online policy as $\pi_\beta^* := \arg \max_{\pi_\beta} L_{\pi_\rho}(\pi_\beta)$.

To bound the approximation error, we first formalize the notion of expected advantage.

Definition 5.5.1 (Expected Advantage). *For a state $x \in \mathcal{X}$ and policy π_θ , the expected advantage at stage i is defined as:*

$$\bar{A}_i(x) := \mathbb{E}_{u_i \sim \pi_\theta} \left[\sum_{i=1}^N A_i^{\pi_\rho} \right] \quad (5.14)$$

Definition 5.5.2 (Coupling Probability). *For two policies π_θ and π_ρ at state $x \in \mathcal{X}$, the coupling probability is defined as:*

$$p_c(x) := \sum_u \pi_\theta(u|x) \cdot \pi_\rho(u|x) \quad (5.15)$$

This represents the probability that both policies select the same action at state x .

Lemma 5.5.2 (Advantage Bound for Coupled Policies). *For α -coupled policies π_θ and π_ρ (where the probability of disagreement is at most α), the expected advantage (Definition 5.5.1) satisfies:*

$$|\bar{A}_i(x)| \leq 2\alpha \max_{x \in \mathcal{X}, u \in \mathcal{U}} |A_0^{\pi_\rho}(x, u)| \quad (5.16)$$

Proof. From the mixing formula (5.10), the coupling probability is:

$$p_c(x) := \sum_u \pi_\theta(u|x) \cdot \pi_\rho(u|x) \quad (5.17)$$

The probability of disagreement is:

$$\mathbb{P}(u_\theta \neq u_\rho | x) = 1 - p_c(x) \leq \alpha \quad (5.18)$$

This bound follows from standard coupling arguments (Schulman et al., 2015) applied to our mixed policy architecture. \square

We now establish the main performance bound.

Proposition 5.5.1 (Performance Bound for Hybrid Policy). *For the total return $\eta(\pi_\theta)$ and estimated performance L_{π_ρ} , the following bound holds:*

$$\eta(\pi_\theta) \geq L_{\pi_\rho}(\pi_\theta) - \frac{2\zeta\gamma\alpha^2}{(1-\gamma)^2} \quad (5.19)$$

where $\zeta = \max_{x \in \mathcal{X}} |\mathbb{E}_{u \sim \pi_\rho^*} [A_0^{\pi_\rho}(x, u)]|$ and γ is the discount factor from GAE (5.9).

Proof. From Lemma 5.5.1, we can rewrite (5.12) as:

$$\eta(\pi_\theta) = \eta(\pi_\rho) + \mathbb{E}_{u \sim \pi_\theta} \left[\sum_{i=1}^N \bar{A}_i(x) \right] \quad (5.20)$$

Similarly, the local approximation becomes:

$$L_{\pi_\rho}(\pi_\theta) = \eta(\pi_\rho) + \mathbb{E}_{u \sim \pi_\rho} \left[\sum_{i=1}^N \bar{A}_i(x) \right] \quad (5.21)$$

Subtracting these expressions:

$$\eta(\pi_\theta) - L_{\pi_\rho}(\pi_\theta) = \sum_{i=1}^N (\mathbb{E}_{u \sim \pi_\theta} [\bar{A}_i(x)] - \mathbb{E}_{u \sim \pi_\rho} [\bar{A}_i(x)]) \quad (5.22)$$

By Lemma 5.5.2 and following the analysis in (Schulman et al., 2015), each term satisfies:

$$\mathbb{E}_{u \sim \pi_\theta} [\bar{A}_i(x)] - \mathbb{E}_{u \sim \pi_\rho} [\bar{A}_i(x)] \leq 4\alpha(1 - (1 - \alpha)^i) \cdot \zeta \quad (5.23)$$

Summing over an infinite horizon with discount factor γ :

$$|\eta(\pi_\theta) - L_{\pi_\rho}(\pi_\theta)| \leq \sum_{i=1}^{\infty} \gamma^i 4\alpha(1 - (1 - \alpha)^i) \cdot \zeta = \frac{4\zeta\alpha}{(1 - \gamma)(1 - \gamma(1 - \alpha))} \leq \frac{4\zeta\alpha}{(1 - \gamma)^2} \quad (5.24)$$

Considering the worst case where $\eta(\pi_\theta) \leq L_{\pi_\rho}(\pi_\theta)$ and noting that the bound is non-negative, we obtain (5.19). \square

Proposition 5.5.1 guarantees that the hybrid policy performs at least as well as the offline policy, minus a penalty term that decreases quadratically with α . This implies that even if we start with a poor online policy (π_β initially random), the hybrid policy π_θ will improve upon the offline policy as long as the mixing parameter α is not too large.

5.5.3 Iterative Improvement Guarantees

We now extend this analysis to show that the hybrid policy improves across iterations of the optimization procedure.

For successive iterations, we model the policy at iteration $i + 1$ as:

$$\pi_{\theta_{i+1}}(u|x) := (1 - \text{TV}_{\theta_i}^{\theta_{i+1}}) \cdot \pi_{\theta_i}(u|x) + \text{TV}_{\theta_i}^{\theta_{i+1}} \cdot \tilde{\pi}'_{\theta_{i+1}}(u|x) \quad (5.25)$$

where $\text{TV}_{\theta_i}^{\theta_{i+1}} = \max_{x \in \mathcal{X}} D_{\text{TV}}(\theta_i \| \theta_{i+1})$ is the maximum total variation distance:

$$D_{\text{TV}}(\theta_i \| \theta_{i+1}) := \frac{1}{2} \sum_j |(u_j \sim \pi_{\theta_i}(x)) - (u'_j \sim \pi_{\theta_{i+1}}(x))| \quad (5.26)$$

and $\tilde{\pi}'_{\theta_{i+1}}(u|x) := \arg \max_{\pi'_{\theta_{i+1}}} L_{\pi_{\theta_i}}(\pi'_{\theta_{i+1}})$.

Theorem 5.5.1 (Monotonic Improvement Across Iterations). *Consider π_{θ_i} as in (5.10) with $\theta_i = (\rho, \beta_i, \alpha_i)$ at optimization iteration i . For successive policies π_{θ_i} and $\pi_{\theta_{i+1}}$ with $\zeta = \max_{x \in \mathcal{X}} |\mathbb{E}_{u \sim \pi_{\beta}^*}[A_0^{\pi_{\theta_i}}(x, u)]|$, the following bound holds:*

$$\eta(\pi_{\theta_{i+1}}) \geq L_{\pi_{\theta_i}}(\pi_{\theta_{i+1}}) - \frac{2\zeta\gamma \cdot (\text{TV}_{\theta_i}^{\theta_{i+1}})^2}{(1 - \gamma)^2} \quad (5.27)$$

Proof. The proof follows the same structure as Proposition 5.5.1, applying the α -coupling argument to the total variation distance $\text{TV}_{\theta_i}^{\theta_{i+1}}$ rather than the mixing parameter α . The key observation is that successive policy iterations can be viewed as coupled policies with coupling strength determined by their total variation distance. Following the same derivation steps establishes that the performance difference between successive policies is bounded by the given term. \square

Theorem 5.5.1 guarantees that each iteration of the optimization procedure improves the policy performance, with the improvement lower-bounded by a term that depends on the total variation between successive policies. This provides a formal foundation for the iterative training procedure.

5.6 TWTL-Based Reward Shaping

Our second main contribution addresses the delayed reward problem directly through reward shaping based on TWTL robustness. The concrete-time reward from Definition 5.3.2 requires complete trajectories spanning the task horizon $\|\phi\|$, but during online learning we

must provide feedback at each time step. We overcome this challenge through trajectory prediction and potential-based reward shaping.

5.6.1 Trajectory Prediction for Robustness Evaluation

Computing the concrete-time reward (5.4) requires complete MDP trajectories that span the entire task horizon. However, obtaining these full trajectories during real-time execution is impractical. We need a method to predict or estimate future observations to evaluate specification satisfaction.

Existing approaches for trajectory completion include explicit trajectory prediction (Salzmann et al., 2020) and runtime monitoring techniques (Chapter 3). We adopt a trajectory prediction approach using learned predictors.

We assume access to a predictor function $\text{Pred} : \mathcal{X} \rightarrow \mathcal{O}^{\|\phi\|+1}$ that maps a current state x_t to a sequence of predicted observations:

$$\text{Pred}(x_t) = (\hat{o}_t, \hat{o}_{t+1}, \dots, \hat{o}_{t+\|\phi\|}) \quad (5.28)$$

Example 5.6.1 (Lunar Lander). *The lunar lander environment from Gymnasium ((Towers et al., 2023)) has state $X = (p_x, p_y, \dot{p}_x, \dot{p}_y, \psi, \dot{\psi}) \in \mathbb{R}^6$, with observation $o = X$, where (p_x, p_y) is the position, (\dot{p}_x, \dot{p}_y) is the velocity, ψ is the angle, and $\dot{\psi}$ is the angular velocity. The control input space \mathcal{U} consists of discrete commands for the main engine and side thrusters. Based on the environment’s success criteria, we define the landing task using TWTL:*

$$\begin{aligned} \phi_{\text{landing}} := & [H^{100} \text{AP}_{\text{hover}}]^{[0, 150]} \cdot [H^{150} \text{AP}_{\text{align}}]^{[100, 300]} \\ & \cdot [H^{150} \text{AP}_{\text{descend}}]^{[250, 450]} \cdot [H^{50} \text{AP}_{\text{land}}]^{[400, 500]} \end{aligned} \quad (5.29)$$

with predicate functions with fixed parameters motivated by the environment’s success criteria: $h_{\text{AP}_{\text{hover}}}(o) = \min\{h_0 - 0.8h_0 - |p_y - 0.8h_0|, 0.1 - |\dot{p}_y|\} \geq 0$; $h_{\text{AP}_{\text{align}}}(o) = \min\{0.2 - |p_x|, 0.1 - |\psi|\} \geq 0$; $h_{\text{AP}_{\text{descend}}}(o) = \min\{-0.2 - \dot{p}_y, \dot{p}_y + 0.5, 0.15 - |\psi|\} \geq 0$; and $h_{\text{AP}_{\text{land}}}(o) = \min\{0.1 - \sqrt{\dot{p}_y^2 + \dot{p}_x^2}, 0.1 - |\psi|, \mathbf{1}_{p_y \leq 0}\} \geq 0$. The time horizon $\|\phi_{\text{landing}}\| = 1403$. The formula ϕ_{landing} reads: “Within time 0 and time 150, the lander must be hovering for a 100 time steps (subformula $H^{100} \text{AP}_{\text{hover}}$). Then, within time 100 and time 300, the lander must be aligned with the landing position for a 150 time steps (subfor-

mula H^{150} AP_{align}). After ensuring that the lander is aligned, descend and then dwelling in the landing position. We use the concatenation operator to ensure the correct landing sequence, which is, in high level, hovering, aligning, descending, then landing.

For the lunar lander domain (Example 5.6.1), we implement this predictor using a Long Short-Term Memory (LSTM) network architecture similar to Trajectron++ (Salzmann et al., 2020) but adapted for single-agent prediction. The network consists of: (i) an input embedding layer embed that processes a window of w past state vectors $\mathbf{x}_{t-w:t} \in \mathbb{R}^{d \times w}$ where d is the state dimension; (ii) an LSTM encoder LSTM_{*e*} for encoding temporal patterns; (iii) an LSTM decoder LSTM_{*d*} for generating future predictions; and (iv) prediction heads for estimating future states.

The prediction is computed as:

$$\hat{\mathbf{x}}_{t:t+\|\phi\|} = \text{LSTM}_d(\text{LSTM}_e(\text{embed}(\mathbf{x}_{t-w:t}))) \quad (5.30)$$

The predictor is trained on expert demonstrations to minimize mean squared error:

$$\mathcal{L}_{\text{pred}} = \sum_{i=t}^{t+\|\phi\|} \|\hat{\mathbf{x}}_i - \mathbf{x}_i\|^2 \quad (5.31)$$

5.6.2 Potential-Based Reward Shaping

Using the predictor and TWTL robustness from Section 3.2.1, we construct a potential-based reward shaping function. Recall that the TWTL robustness $\rho : \mathcal{O} \times \Phi_{\text{TWTL}} \rightarrow \mathbb{R}$ measures how robustly an observation sequence satisfies a formula, with positive values indicating satisfaction and negative values indicating violation.

Definition 5.6.1 (TWTL-Based Potential Function). *For a TWTL task ϕ , we define a potential-based reward shaping function $F : \mathcal{X} \times \mathcal{U} \times \mathcal{X} \times \Phi_{\text{TWTL}} \rightarrow \mathbb{R}$ with discount factor $0 < \kappa < 1$ as:*

$$F(x_t, u_t, x_{t+1}, \phi) := \kappa \cdot \rho(\text{Pred}(x_t), \phi) - \rho(\text{Pred}(x_{t+1}), \phi) \quad (5.32)$$

The shaped reward is then:

$$r_{\phi,t}^{\prime k} := r_{\phi,t}^k + F(x_t, u_t, x_{t+1}, \phi) \quad (5.33)$$

This reward shaping provides immediate feedback at each time step based on predicted future satisfaction of the specification, while the following lemma guarantees that it preserves the optimal policy.

Lemma 5.6.1 (Optimal Policy Preservation). *The optimal policy of the original MDP with reward $r_{\phi,t}^k$ is identical to the optimal policy of the MDP with shaped reward $r_{\phi,t}^{\prime k}$.*

Proof. By Theorem 1 in (Ng et al., 1999), any reward shaping function F of the form $F(s, a, s') = \gamma\Psi(s') - \Psi(s)$ for some potential function Ψ preserves the optimal policy. Our shaping function (5.32) has exactly this form with potential $\Psi(x) = \rho(\text{Pred}(x), \phi)$ and discount factor κ replacing γ . \square

The intuition behind this reward shaping is that it provides dense feedback about progress toward specification satisfaction. When the predicted trajectory moves closer to satisfying the specification (increasing robustness), the agent receives positive reward. When it moves away from satisfaction (decreasing robustness), it receives negative reward. This transforms the sparse delayed reward signal into a dense immediate signal while preserving optimality.

5.6.3 Integration with Hybrid Architecture

The complete framework combines the hybrid policy architecture with TWTL-based reward shaping. The shaped rewards are used to compute the advantage estimates in (5.9), which then guide the optimization of both the online policy π_β and the mixing parameter α .

Importantly, the theoretical guarantees from Proposition 5.5.1 and Theorem 5.5.1 continue to hold with shaped rewards due to Lemma 5.6.1. The performance bounds apply

to the original reward function, but optimization proceeds using the shaped rewards which provide better gradient information.

5.7 Algorithm: Accelerated PPO

We now present the complete Accelerated Proximal Policy Optimization (APPO) algorithm that integrates the hybrid policy architecture with TWTL-based reward shaping.

5.7.1 Clipped Objective with Hybrid Policy

Following the majorization-minimization framework (Hunter and Lange, 2004), maximizing the right-hand side of (5.27) implies maximizing the total return of $\pi_{\theta_{i+1}}$. We use importance sampling to estimate θ_{i+1} that maximizes:

$$L_{\theta_i}(\theta_{i+1}) = \eta(\theta_i) + \sum_{x \in \mathcal{X}} P_{\theta_i}(x) \sum_{u \in \mathcal{U}} \pi_{\theta_{i+1}}(u|x) A_0^{\theta_i}(x, u) \quad (5.34)$$

Including a penalty on the total variation distance leads to a clipped objective with the same structure as standard PPO but defined with respect to the hybrid policy (5.10). We define the TWTL-specific clipped objective:

$$J_{\phi}^k(x, u, \theta, \theta_i) := \min \left[\frac{\pi_{\theta}(u|x)}{\pi_{\theta_i}(u|x)} \hat{A}_{\phi, t}^{\pi_{\theta_i, \rho}, k}, g(\epsilon, \hat{A}_{\phi, t}^{\pi_{\theta_i, \rho}, k}) \right] \quad (5.35)$$

where $\hat{A}_{\phi, t}^{\pi_{\theta_i, \rho}, k}$ denotes the advantage estimate computed using the shaped reward $r_{\phi, t}^k$ from (5.33).

The policy update becomes:

$$\theta_{i+1} \leftarrow \arg \max_{\theta \in \Theta} \mathbb{E}[J_{\phi}^k(x, u, \theta, \theta_i)] \quad (5.36)$$

5.7.2 Main Algorithm

Algorithm 5: Accelerated PPO

```

1 Input  $\pi_\rho(\cdot)$  a fixed task predictor trained offline
2 Initialize  $\{\pi_{\theta,t}\}_{t=0}^{N-1}$  with  $\theta = \theta_0$ 
3 foreach episode  $k \in \{1, \dots, K\}$  do
4   Observe  $x_1$ 
5   foreach iteration  $i \in \{1, \dots, N\}$  do
6      $\theta_{i+1}^k \leftarrow_{\theta \in \Theta} \mathbb{E}[J_\phi^k(x, u, \theta, \theta_i)]$ 
7     Apply the control  $u_i \sim \pi_{\theta_i}$ 
8     Compute  $\hat{A}_{\phi,t}^{\pi_{\theta_i,t},k}$  according to (5.9)

```

Algorithm 5 presents the complete APPO procedure. The algorithm alternates between collecting trajectories under the current policy (Lines 3-8), computing advantage estimates with shaped rewards (Lines 9-11), and updating the policy parameters including both the online policy π_β and mixing parameter α (Line 12).

The key differences from standard PPO are: (1) the hybrid policy architecture combining offline and online components (Line 1, implicit in sampling Line 5), (2) the reward shaping using TWTL robustness and trajectory prediction (Line 7), and (3) the modified objective that accounts for policy mixing (Line 12).

5.7.3 Implementation Details

The actor-critic architecture consists of (see Figure 5.1):

- **Offline Policy** π_ρ : Multi-layer perceptron (MLP) trained via behavior cloning on expert demonstrations, with parameters frozen during APPO training
- **Online Policy** π_β : MLP with identical architecture to π_ρ , parameters optimized during training

- **Mixing Layer:** Fully connected layer implementing (5.10) with learnable parameter α
- **Critic:** MLP that approximates the value function $V^{\pi_\theta}(x)$ for advantage estimation
- **Trajectory Predictor:** LSTM network implementing (5.30), pre-trained on expert demonstrations

During training, gradients flow through the online policy π_β and mixing parameter α , while the offline policy π_ρ and predictor Pred remain fixed. This allows the algorithm to learn both how to improve the online policy and how to optimally combine it with expert knowledge.

5.8 Experimental Validation

We validate APPO on two robotic control benchmarks from OpenAI Gymnasium (Towers et al., 2023): Inverted Pendulum and Lunar Lander. These environments present complementary challenges for delayed reward learning.

As a continuation of Examples 5.6.1, we demonstrate the empirical performance of our approach in Figure 5.2. For the Lunar Lander environment, where we previously defined the landing task TWTL formula ϕ_{landing} and the LSTM-based task predictor, combining policy mixing with reward shaping (green) achieves faster learning and better asymptotic performance compared to both vanilla PPO (blue) and reward shaping alone (orange). The improvement is particularly noticeable in the early training phase (0-0.5M steps), where our method accelerates learning.

To test the robustness of our mixing approach, we intentionally degraded pre-trained PPO policies by adding controlled noise to their parameters, using these as our offline policies π_ρ . Despite starting with these suboptimal policies, our method successfully leverages their partial knowledge while learning to improve upon them. For comparison, we also

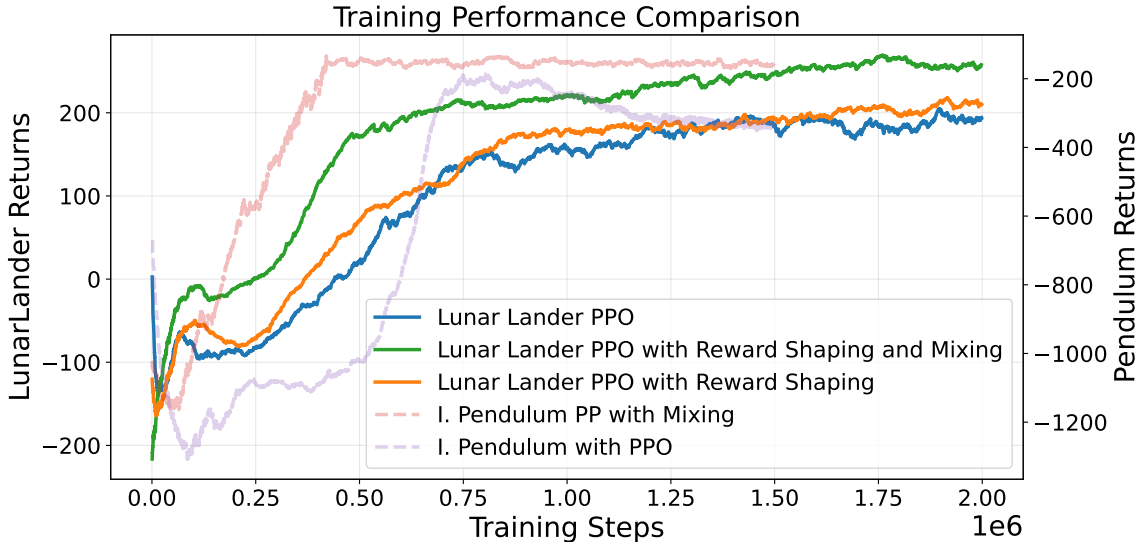


Figure 5.2: Training performance comparison across different variants of PPO in LunarLander-v2 and Pendulum environments. For LunarLander-v2 (left y-axis), we compare vanilla PPO against variants with reward shaping and mixing. The results show that combining reward shaping with policy mixing achieves faster learning and better performance compared to baseline PPO and reward shaping alone. For Pendulum (right y-axis, dashed lines), we include PPO with and without mixing for reference. Training steps are shown in millions on the x -axis.

tested our approach on the Inverted Pendulum environment with similarly degraded offline policies (right y-axis, dashed lines), where policy mixing again demonstrates improved learning dynamics over the baseline PPO implementation.

5.9 Conclusion and Future Work

We presented an approach to accelerate reinforcement learning in environments with delayed rewards through two key innovations: a mixed policy architecture that combines potentially suboptimal offline policies with online learning, and a TWTL-based reward shaping mechanism enabled by a task predictor. For the mixed policy architecture, we established theoretical guarantees showing consistent improvement over both the offline policy and previous iterations. Our empirical results on the Lunar Lander and Inverted Pendulum environments demonstrate that even with degraded offline policies, our method

achieves faster learning and better asymptotic performance compared to vanilla PPO.

Several directions for future work emerge from this study. First, exploring more complex TWTL specifications that capture intricate temporal dependencies could extend our framework to more challenging tasks. Second, investigating adaptive mixing strategies that automatically adjust based on the relative performance of offline and online policies could further improve learning efficiency. The integration of other temporal logic formalisms and their corresponding quantitative semantics could also provide interesting avenues for reward shaping in reinforcement learning.

Chapter 6

Conclusion and Future Directions

This dissertation developed a unified framework for temporal logic specifications in robotics centered on Arithmetic-Geometric Mean (AGM) robustness as a quantitative semantics. We established theoretical foundations, developed practical algorithms, and demonstrated effectiveness across three domains: runtime monitoring, sampling-based motion planning, and reinforcement learning with delayed rewards. This final chapter summarizes the main contributions, discusses limitations of current approaches, and proposes directions for future research.

6.1 Summary of Contributions

6.1.1 Quantitative Semantics and Monitoring

Chapter 3 developed AGM robustness semantics for Time Window Temporal Logic TWTL and monitoring algorithms for both TWTL and STL specifications. The key insight is that traditional min-max robustness, while sound, focuses exclusively on the most critical time points and subformulae, creating non-smooth optimization landscapes and ignoring the broader context of satisfaction. AGM robustness addresses these limitations through holistic aggregation using arithmetic means when robustness values have mixed signs and geometric means when values share signs.

We established formal soundness of AGM robustness: positive robustness implies Boolean satisfaction, and negative robustness implies violation. This guarantees that optimization or learning algorithms maximizing AGM robustness will produce specification-

satisfying behaviors. Importantly, AGM robustness provides finer-grained discrimination between trajectories than traditional robustness, rewarding trajectories that satisfy specifications more robustly across all time points rather than just avoiding critical violations.

For online monitoring during incremental planning and control, we introduced AGM robustness interval semantics that bound possible robustness values over all completions of partial trajectories. The incremental monitoring algorithm 1 efficiently updates these intervals for STL as new observations arrive, with complexity $O(|\phi|)$ per observation compared to $O(|s|^2|\phi|)$ for non-incremental approaches. For typical scenarios with trajectory length 20 – 50 and formula complexity 10 – 20, this reduces monitoring overhead from thousands to hundreds of operations per evaluation.

We proved three critical properties of the interval semantics: soundness (true robustness lies within computed bounds for all completions), chain inclusion (intervals shrink monotonically as trajectories extend), and convergence (intervals converge to exact robustness when trajectory horizon reaches formula horizon). These properties enable informed decision-making during tree expansion in sampling-based planning and policy optimization in reinforcement learning.

6.1.2 Sampling-Based Motion Planning

Chapter 4 presented RRT ^{η} , integrating STL AGM robustness into the RRT* framework for sampling-based motion planning from STL specifications. The framework addresses two key challenges: providing consistent guidance for tree exploration through Direction of Increasing AGM DIAS vectors, and balancing multiple competing temporal objectives through principled composition mechanisms.

DIAS vectors leverage AGM’s smooth gradients to guide steering during tree expansion. Unlike traditional robustness gradients that exhibit abrupt changes when the critical subformula or time point shifts, AGM gradients remain continuous as they incorporate information from all time points and subformulae. This enables more effective steering

toward regions of higher specification satisfaction.

For multi-objective composition, we introduced two approaches. The stochastic baseline provides probabilistic exploration guarantees necessary for asymptotic optimality. More significantly, the FPL-based composition computes gradient-based weights from fulfillment values derived from AGM intervals, naturally prioritizing less-fulfilled objectives. This principled balancing mechanism provides $1.5 - 2\times$ computational advantages over stochastic methods through forward prediction of which objectives need attention.

We proved that RRT^η maintains the probabilistic completeness and asymptotic optimality guarantees of RRT^* despite using the more complex AGM robustness measure. The key observation is that AGM’s smoothness properties and the interval semantics’ chain inclusion enable sound pruning decisions during tree construction, while the randomized composition mechanisms ensure continued exploration of all feasible regions.

Experimental validation on three robotic systems demonstrated substantial advantages. On the unicycle sequential reach-avoid task, traditional robustness-based planning failed completely with persistently negative lower bounds, while AGM-based methods achieved $\eta_L \approx 0.93-0.95$. The fundamental limitation of traditional robustness—pessimistic evaluation that cannot recognize acceptable brief proximity to constraint boundaries—prevented discovery of feasible trajectories that AGM methods found readily.

For the 7-DOF KUKA manipulator with cascading choice problems (choosing between region pairs), FPL-based composition reached convergence ($\text{gap} < 0.1$) in 1000 iterations compared to 2500 for stochastic composition. The forward prediction capability enabled principled resolution of the choice structure rather than discovering good sequences through repeated random sampling. The augmented state representation with IK-based sampling achieved $185\times$ speedup for tight workspace constraints through elimination of redundant forward kinematics computations and more effective sampling bias.

6.1.3 Reinforcement Learning with Delayed Rewards

Chapter 5 developed Accelerated Proximal Policy Optimization (APPO) addressing the challenge of delayed rewards where actions leading to successful outcomes may not receive immediate feedback. The framework combines a hybrid policy architecture with TWTL-based reward shaping.

The hybrid architecture mixes an offline policy trained on expert demonstrations with an online policy optimized through PPO: $\pi_{\theta}(u|x) = (1 - \alpha)\pi_{\rho}(u|x) + \alpha\pi_{\beta}(u|x)$. Unlike previous approaches using offline data merely for initialization, we maintain the offline policy as an active guide throughout training with learnable mixing parameter α . We proved monotonic improvement over both the offline policy and previous iterations, with bounded performance gap $(2\zeta\gamma\alpha^2)/(1 - \gamma)^2$.

The mixing parameter automatically adapts based on relative policy quality. Initially, α remains low (high weight on offline policy) when the online policy is poorly initialized. As the online policy improves through learning, α increases to rely more heavily on learned behavior.

For reward shaping, we use TWTL robustness with LSTM-based trajectory prediction to provide immediate feedback about specification satisfaction. The potential-based shaping $F(x_t, u_t, x_{t+1}, \phi) = \kappa \cdot \rho(\text{Pred}(x_t), \phi) - \rho(\text{Pred}(x_{t+1}), \phi)$ transforms sparse delayed rewards into dense immediate signals while preserving the optimal policy through the potential-based structure. This addresses the temporal credit assignment problem by providing step-by-step feedback about progress toward specification satisfaction.

Experimental validation on the lunar lander task with delayed rewards demonstrated the effectiveness of combining both components. The complementary benefits—offline policy providing good initialization and regularization, reward shaping providing dense feedback—combine synergistically rather than additively.

6.1.4 Unifying Framework

A key insight emerging from this dissertation is that AGM robustness serves as a unifying mathematical framework applicable across diverse robotic domains. The same core principle—holistic aggregation of satisfaction information using arithmetic and geometric means—provides benefits in:

Motion Planning: Smoother exploration landscapes and more effective steering guidance compared to traditional robustness’s sharp decision boundaries.

Reinforcement Learning: Dense reward signals that maintain correlation with specification satisfaction while providing immediate feedback at each time step.

Runtime Monitoring: Efficient incremental computation with $O(|\phi|)$ complexity per observation through the use of incremental modification functions.

This unification suggests that AGM robustness captures fundamental properties of specification satisfaction that transcend specific application contexts. The additive structure (arithmetic mean) and multiplicative structure (geometric mean) provide natural ways to aggregate information that align with how satisfaction quality should be measured: when some parts of the specification are violated, average satisfaction of other parts matters; when all parts are satisfied, compound satisfaction quality matters.

6.2 Limitations and Assumptions

While this dissertation demonstrates the effectiveness of AGM robustness across multiple domains, several limitations and assumptions warrant discussion:

6.2.1 Discrete-Time and Finite Horizons

All algorithms and theoretical results assume discrete-time systems with finite planning or learning horizons. While STL and TWTL can in principle handle continuous-time signals and infinite horizons, our monitoring algorithms process observations at discrete time steps,

and our planning and learning frameworks optimize over finite trajectories. Extensions to continuous-time monitoring would require interval arithmetic over continuous domains, and infinite-horizon planning would require careful treatment of discounting in the AGM operators to ensure convergence.

6.2.2 Differentiability Requirements

The use of gradient-based methods in both motion planning (DIAS vectors) and reinforcement learning (policy gradients) requires predicates with differentiable evaluation functions. This limits applicability to specifications with non-differentiable predicates such as mode switches, discrete logical constraints, or indicator functions. While we can approximate non-differentiable functions (e.g., using smooth approximations of the indicator function), formal guarantees only hold for truly differentiable predicates.

6.2.3 Trajectory Prediction Quality

The TWTL-based reward shaping in APPO relies on trajectory prediction to evaluate specification satisfaction for incomplete trajectories. The quality of the predictor directly impacts the quality of the shaped rewards. Poor predictions can provide misleading gradient signals that slow learning or lead to suboptimal policies. We assumed access to sufficient expert demonstrations to train reasonable predictors, but in domains where expert data is scarce or expensive, predictor quality may become a bottleneck.

The theoretical guarantee that potential-based shaping preserves optimal policies relies on the assumption that predictions are deterministic functions of state. In reality, learned predictors are stochastic (due to epistemic uncertainty) and may produce inconsistent predictions across states. While empirically we found that even imperfect predictors provide useful guidance, formal analysis of robustness to prediction errors remains an open question.

6.2.4 Computational Complexity

While AGM robustness provides smoother optimization landscapes than traditional robustness, computing it still requires evaluating the entire formula over trajectories. For complex specifications with many nested temporal operators, this can become computationally expensive. The incremental monitoring algorithms reduce the per-observation cost, but the overall complexity still scales with formula size.

For motion planning, RRTⁿ inherits the high sampling requirements of RRT* in high-dimensional state spaces. While our augmented state representation with IK-based sampling provides substantial speedups for manipulators, truly high-dimensional systems (e.g., humanoid robots with 30+ degrees of freedom) remain challenging. The probabilistic completeness guarantee requires infinite samples in the limit; finite-sample performance depends on problem-specific factors, including obstacle density and specification complexity.

For reinforcement learning, APPO requires both offline demonstrations and online environment interaction. In real-world robotics where environment interaction is expensive (e.g., due to hardware wear, safety concerns, or time constraints), sample efficiency remains a critical challenge. While APPO improves upon vanilla PPO, it still requires hundreds of thousands to millions of environment steps for complex tasks.

6.2.5 Specification Design

All approaches in this dissertation assume that temporal logic specifications correctly capture the desired behavior. In practice, translating high-level human intent into formal specifications is challenging and error-prone. Specifications that are too restrictive may be infeasible or unnecessarily constrain behavior; specifications that are too loose may permit undesired behaviors.

Moreover, threshold selection for predicate functions requires domain expertise. The

choice of threshold determines the boundary between satisfaction and violation, directly impacting robustness values and optimization behavior. While AGM robustness provides more nuanced discrimination than traditional robustness, inappropriate threshold selection can still lead to poor performance.

6.3 Future Research Directions

The limitations and assumptions discussed above point to several promising directions for future research:

6.3.1 Probabilistic and Stochastic Specifications

The current framework handles deterministic specifications over stochastic system trajectories, but many robotic applications involve inherent uncertainty in both system dynamics and environmental conditions. Probabilistic Signal Temporal Logic (PrSTL) (Sadigh and Kapoor, 2016) and other stochastic temporal logic formalisms, see (Yoo and Belta, 2015), extend classical temporal logics to reason about probabilities of satisfaction.

Extending AGM robustness to probabilistic specifications would enable reasoning about satisfaction likelihoods while maintaining the benefits of holistic aggregation. One approach would define AGM robustness over probability distributions, using arithmetic/geometric mean operators to aggregate across both time points and probabilistic outcomes. The key challenges include defining appropriate semantics that preserve soundness and developing efficient monitoring algorithms that can track probability distributions over partial trajectories.

For motion planning under uncertainty, integrating AGM robustness with chance-constrained optimization or stochastic programming could enable synthesis of policies that maximize expected robustness subject to probability bounds on constraint violation. For reinforcement learning, extending APPO to risk-sensitive objectives using distributional RL

techniques could address safety-critical applications requiring high-confidence guarantees.

6.3.2 Dynamic Obstacles and Time-Varying Environments

The motion planning framework assumes static environments with fixed obstacles. Real-world applications often involve dynamic obstacles such as moving pedestrians, vehicles, or other robots. Extending RRTⁿ to dynamic environments requires several innovations.

First, the DIAS vectors must incorporate predictions of future obstacle positions. This could leverage the trajectory prediction techniques developed for APPO, but applied to obstacles rather than the robot itself. The interval semantics would need to bound robustness over both incomplete robot trajectories and uncertain obstacle futures.

Second, the tree would need replanning capabilities when environmental changes invalidate previous solutions. Incremental variants of RRT* that reuse previous search efforts could be adapted to the AGM robustness setting, leveraging the interval semantics to identify which portions of the tree remain valid after environment changes.

Third, temporal logic specifications themselves may need temporal evolution to capture time-varying requirements. For example, "always avoid region A except during time window [10,20] when A is temporarily passable" requires specifications with time-dependent predicates.

6.3.3 Integration with Neural Network Controllers

The reinforcement learning framework uses neural network policies, but the motion planning framework relies on classical control. Integrating learned neural network controllers into the sampling-based planning framework could combine the benefits of both approaches: learning-based adaptation to complex dynamics with formal verification of temporal logic specifications.

One approach would train neural network controllers to maximize AGM robustness using supervised learning on successful RRTⁿ trajectories. The trained controllers could then

be verified offline using the monitoring algorithms from Chapter 3 over simulated trajectories. Intervals with positive lower bounds provide certificates of specification satisfaction.

Another direction involves online adaptation during planning. Rather than steering using optimal control solutions (which are computationally expensive), learned controllers could propose candidate actions quickly. The DIAS vectors could then serve as correction signals when the learned controller produces actions that decrease predicted robustness. This hybrid approach could achieve the speed of learned controllers with the correctness guarantees of formal methods.

For safety-critical applications, combining AGM robustness with control barrier functions could provide both temporal logic satisfaction and continuous safety guarantees. The CBF would ensure immediate safety (e.g., collision avoidance at each time step) while AGM robustness optimizes longer-term temporal objectives.

6.3.4 Computational Efficiency and Approximation

While incremental monitoring provides substantial speedups, computing AGM robustness for complex specifications remains expensive. Several directions could improve computational efficiency.

Parallel evaluation of independent subformulae could leverage multi-core processors or GPUs. The recursive AGM robustness computation exhibits opportunities for parallelism when subformulae do not share predicates. Implementing monitoring algorithms on GPU architectures could provide orders of magnitude speedup for specifications with many independent components.

Approximate AGM robustness computation could trade accuracy for speed. For very long trajectories or complex specifications, computing exact robustness over all time points may be unnecessary. Adaptive sampling schemes could identify which time points and subformulae contribute most to the overall robustness, focusing computation on critical regions while approximating less important contributions.

Hierarchical specifications with abstraction levels could reduce complexity. High-level specifications could be refined into detailed low-level specifications only when needed. The monitoring algorithms could maintain coarse robustness intervals at high levels, refining to detailed intervals only for regions of the trajectory where high-level robustness is close to decision boundaries.

6.3.5 Real-World Validation and Deployment

The experimental validation in this dissertation uses simulated environments with perfect state observation and deterministic dynamics (up to learning stochasticity). Real-world deployment would require addressing several practical challenges.

State estimation errors and perception noise affect robustness computation. The predicates assume access to true state, but real systems rely on noisy sensors and state estimators. Robust monitoring algorithms that propagate uncertainty through robustness computation could provide confidence bounds on robustness estimates given sensor noise characteristics.

Hardware constraints including actuator limits, communication delays, and computational resources affect feasibility of real-time implementation. The monitoring algorithms must run fast enough to provide feedback within control loop timing constraints. Embedded implementations on resource-constrained platforms may require approximations or specialized hardware acceleration.

Safety certification for deployment in human environments requires more than specification satisfaction. Formal verification that the system satisfies specifications under all possible disturbances and failures may be necessary. Combining AGM robustness with formal verification tools could provide such guarantees, using the monitoring algorithms to generate certificates of satisfaction over representative test cases.

6.4 Broader Implications

Beyond the specific technical contributions, this dissertation offers broader insights for formal methods in robotics and autonomous systems.

6.4.1 Bridging Theory and Practice

This dissertation demonstrates the following: AGM robustness emerged from observing practical limitations of traditional robustness (non-smooth landscapes, brittleness, poor multi-objective composition), led to mathematical development of improved semantics, enabled proof of formal guarantees, and validated effectiveness through extensive experiments.

Pure theory risks developing elegant mathematics disconnected from real needs; pure empiricism risks developing ad-hoc methods without understanding of their limitations. The integration honors both perspectives.

6.4.2 The Role of Quantitative Semantics

Traditional formal methods often emphasize binary verification: does the system satisfy the specification or not? While such guarantees are valuable, they provide limited guidance for optimization. The quantitative semantics developed in this dissertation demonstrate that continuous measures of satisfaction quality enable much richer applications: gradient-based optimization, reinforcement learning, multi-objective balancing.

This suggests that future formal methods research should prioritize development of quantitative semantics alongside Boolean semantics. Whenever we introduce a new specification formalism, we should ask: how can we measure the degree of satisfaction? What properties should this measure have (soundness, continuity, additivity)? How can we compute it efficiently?

AGM robustness represents one answer to these questions for temporal logic, but many

specification formalisms lack well-developed quantitative semantics. Graph specifications, spatial logic, and epistemic logic all offer rich expressive power but limited quantitative measures. Developing principled quantitative semantics for these formalisms could enable their integration into optimization-based synthesis and learning.

6.4.3 Specification as Communication

Temporal logic specifications serve as communication medium between humans and robots: humans express desired behavior formally, and robots synthesize control strategies to satisfy specifications. The quantitative semantics enhance this communication by providing continuous feedback about satisfaction quality.

When a robot reports “current AGM robustness is 0.3,” this communicates more than just “specification is satisfied.” It indicates moderate satisfaction quality with room for improvement. When robustness decreases during execution, this signals potential problems before violation occurs, enabling proactive intervention.

This perspective suggests that future work should develop better interfaces for humans to express specifications and understand robustness feedback. Visualization tools that show which subformulae contribute most to overall robustness, sensitivity analysis indicating which predicates are close to decision boundaries, and explanation systems that generate natural language descriptions of robustness values could make formal methods more accessible to non-experts.

6.5 Closing Remarks

This dissertation developed AGM robustness as a unifying quantitative semantic for temporal logic specifications in robotics, demonstrating its effectiveness across runtime monitoring, sampling-based motion planning, and reinforcement learning with delayed rewards. The framework provides both theoretical guarantees (soundness, completeness, optimality)

and practical benefits (smooth optimization landscapes, efficient computation, principled multi-objective composition).

The experimental validation across diverse robotic systems—double integrator, unicycle, 7-DOF manipulator, inverted pendulum, lunar lander—demonstrates broad applicability. The consistent improvements over traditional robustness-based approaches across all three domains suggest that holistic aggregation of satisfaction information captures fundamental properties that transcend specific applications.

Several themes emerged throughout this work that may guide future research. First, the importance of combining theoretical rigor with empirical validation, using each to inform the other. Second, the value of quantitative semantics that provide continuous measures of satisfaction quality rather than binary verification. Third, the potential for unified mathematical frameworks like AGM robustness that apply across diverse domains.

As robotic systems become increasingly autonomous and pervasive in society, ensuring their behaviors align with human intentions becomes ever more critical. Formal methods and temporal logic specifications provide powerful tools for this challenge, enabling precise expression of requirements and automated synthesis of correct behaviors. The quantitative semantics and algorithms developed in this dissertation contribute to making these tools more practical, more effective, and more broadly applicable to real-world robotic systems.

Appendix A

Quantitative Semantics Proofs

A.1 Correctness of Incremental Modification Functions

We provide the detailed proof of Lemma 3.4.1, establishing that incremental modification functions produce identical results to full AGM robustness recomputation.

Proof. We prove correctness for disjunction; conjunction follows by symmetry. Let $\{r_1, \dots, r_{N-1}\}$ be the values used to compute $\eta = \text{AGM}_\vee(r_1, \dots, r_{N-1})$, and let η' be a new observation.

Case 1: $\eta < 0 \wedge \eta' < 0$

Since all values r_1, \dots, r_{N-1} must be negative for $\eta < 0$, by equation (2.31) we have:

$$\eta = -\sqrt[N-1]{\prod_{i=1}^{N-1} (1 - r_i)} + 1 \quad (\text{A.1})$$

Rearranging:

$$(1 - \eta)^{N-1} = \prod_{i=1}^{N-1} (1 - r_i) \quad (\text{A.2})$$

Now computing the full AGM with all N values:

$$\begin{aligned} \text{AGM}_\vee(r_1, \dots, r_{N-1}, \eta') &= -\sqrt[N]{\prod_{i=1}^N (1 - r_i)} + 1 \\ &= -\sqrt[N]{\prod_{i=1}^{N-1} (1 - r_i) \cdot (1 - \eta')} + 1 = -\sqrt[N]{(1 - \eta)^{N-1} \cdot (1 - \eta')} + 1 \end{aligned} \quad (\text{A.3})$$

This exactly matches the first case of equation (3.22):

$$\text{mdf_AGM}_\vee(\eta, N, \eta') = -\sqrt[N]{(1 - \eta)^{N-1} \cdot (1 - \eta')} + 1 \quad (\text{A.4})$$

Case 2: $\eta < 0 \wedge \eta' > 0$

Since $\eta < 0$, all previous values $r_i < 0$, thus $[r_i]_+ = 0$ for all $i \in \{1, \dots, N-1\}$. The full AGM computation switches to arithmetic mean:

$$\begin{aligned} \text{AGM}_V(r_1, \dots, r_{N-1}, \eta') &= \frac{1}{N} \sum_{i=1}^N [r_i]_+ \\ &= \frac{1}{N} \left(\sum_{i=1}^{N-1} [r_i]_+ + [\eta']_+ \right) = \frac{1}{N} (0 + [\eta']_+) = \frac{[\eta']_+}{N} \end{aligned} \quad (\text{A.5})$$

This matches the second case of equation (3.22).

Case 3: Otherwise

In this case, at least one value among r_1, \dots, r_{N-1} was positive, so the AGM used arithmetic mean of positive parts:

$$\eta = \frac{1}{N-1} \sum_{i=1}^{N-1} [r_i]_+ \quad (\text{A.6})$$

Therefore:

$$\sum_{i=1}^{N-1} [r_i]_+ = (N-1)\eta \quad (\text{A.7})$$

Computing the full AGM with the new value:

$$\begin{aligned} \text{AGM}_V(r_1, \dots, r_{N-1}, \eta') &= \frac{1}{N} \sum_{i=1}^N [r_i]_+ \\ &= \frac{1}{N} \left(\sum_{i=1}^{N-1} [r_i]_+ + [\eta']_+ \right) \\ &= \frac{(N-1)\eta + [\eta']_+}{N} \end{aligned} \quad (\text{A.8})$$

However, equation (3.22) shows:

$$\text{mdf_AGM}_V(\eta, N, \eta') = \frac{(N \cdot \eta - [\eta]_+) + [\eta']_+}{N} \quad (\text{A.9})$$

To reconcile these expressions, observe that:

$$\frac{(N-1)\eta + [\eta']_+}{N} = \frac{N \cdot \eta - \eta + [\eta']_+}{N} \quad (\text{A.10})$$

If $\eta > 0$, then $[\eta]_+ = \eta$, yielding:

$$\frac{N \cdot \eta - [\eta]_+ + [\eta']_+}{N} = \frac{(N-1)\eta + [\eta']_+}{N} \quad (\text{A.11})$$

If $\eta \leq 0$, then $[\eta]_+ = 0$, and the expressions are trivially equal. Thus the third case is verified. \square

A.2 STL AGM Robustness Interval Soundness

Proof. We prove the soundness by structural induction over formula ϕ .

Base Cases:

Predicate μ : For a predicate, if the signal is complete at time t , then $[\rho](\mathbf{s}_t, \mu) = \{\rho(\mathbf{s}_t, \mu)\}$ and $[\eta](\mathbf{s}_t, \mu) = \{\eta(\mathbf{s}_t, \mu)\}$, so the inclusions hold trivially.

Inductive Cases:

Conjunction ($\phi_1 \wedge \phi_2$): We have $[\rho](\mathbf{s}_{t_1, t'}, \phi_1 \wedge \phi_2) = \min([\rho](\mathbf{s}_{t_1, t'}, \phi_1), [\rho](\mathbf{s}_{t_1, t'}, \phi_2))$ and $[\eta](\mathbf{s}_{t_1, t'}, \phi_1 \wedge \phi_2) = \text{AGM}_\wedge([\eta](\mathbf{s}_{t_1, t'}, \phi_1), [\eta](\mathbf{s}_{t_1, t'}, \phi_2))$.

By the induction hypothesis, for any completion $\mathbf{s}_{t_1, t_2} \in \mathcal{C}$, we have $\rho(\mathbf{s}_{t_1, t_2}, \phi_1) \in [\rho](\mathbf{s}_{t_1, t'}, \phi_1)$ and $\rho(\mathbf{s}_{t_1, t_2}, \phi_2) \in [\rho](\mathbf{s}_{t_1, t'}, \phi_2)$. Since $\rho(\mathbf{s}_{t_1, t_2}, \phi_1 \wedge \phi_2) = \min\{\rho(\mathbf{s}_{t_1, t_2}, \phi_1), \rho(\mathbf{s}_{t_1, t_2}, \phi_2)\}$, by the properties of interval arithmetic on minimum operations, we have $\rho(\mathbf{s}_{t_1, t_2}, \phi_1 \wedge \phi_2) \in [\rho](\mathbf{s}_{t_1, t'}, \phi_1 \wedge \phi_2)$.

Similarly, by the induction hypothesis, $\eta(\mathbf{s}_{t_1, t_2}, \phi_1) \in [\eta](\mathbf{s}_{t_1, t'}, \phi_1)$ and $\eta(\mathbf{s}_{t_1, t_2}, \phi_2) \in [\eta](\mathbf{s}_{t_1, t'}, \phi_2)$. Since $\eta(\mathbf{s}_{t_1, t_2}, \phi_1 \wedge \phi_2) = \text{AGM}_\wedge(\eta(\mathbf{s}_{t_1, t_2}, \phi_1), \eta(\mathbf{s}_{t_1, t_2}, \phi_2))$, by the properties of interval arithmetic on AGM operations, we have $\eta(\mathbf{s}_{t_1, t_2}, \phi_1 \wedge \phi_2) \in [\eta](\mathbf{s}_{t_1, t'}, \phi_1 \wedge \phi_2)$.

Disjunction ($\phi_1 \vee \phi_2$): The proof follows similarly to the conjunction case, using maximum operations for standard robustness and AGM_\vee for AGM robustness, along with the corresponding interval arithmetic properties.

Negation ($\neg\phi$): We have $[\rho](\mathbf{s}_{t_1, t'}, \neg\phi) = -[\rho](\mathbf{s}_{t_1, t'}, \phi)$ and $[\eta](\mathbf{s}_{t_1, t'}, \neg\phi) = -[\eta](\mathbf{s}_{t_1, t'}, \phi)$. By the induction hypothesis, $\rho(\mathbf{s}_{t_1, t_2}, \phi) \in [\rho](\mathbf{s}_{t_1, t'}, \phi)$ and $\eta(\mathbf{s}_{t_1, t_2}, \phi) \in [\eta](\mathbf{s}_{t_1, t'}, \phi)$. Since $\rho(\mathbf{s}_{t_1, t_2}, \neg\phi) = -\rho(\mathbf{s}_{t_1, t_2}, \phi)$ and $\eta(\mathbf{s}_{t_1, t_2}, \neg\phi) = -\eta(\mathbf{s}_{t_1, t_2}, \phi)$, the results follow by the properties of interval arithmetic under negation.

Globally ($\mathbf{G}_{[a,b]}\phi$): We prove this case by contradiction. Consider partial signal $\mathbf{s}_{t_1, t'}$ and assume that for some completion $\mathbf{s}_{t_1, t_2} \in \mathcal{C}$, we have $\rho(\mathbf{s}_{t_1, t_2}, \mathbf{G}_{[a,b]}\phi) \notin [\rho](\mathbf{s}_{t_1, t'}, \mathbf{G}_{[a,b]}\phi)$.

Case 1: If $t' - t_1 \geq b$, then by definition of the interval semantics, $[\rho](\mathbf{s}_{t_1, t'}, \mathbf{G}_{[a,b]}\phi) = \{\rho(\mathbf{s}_{t_1, t'}, \mathbf{G}_{[a,b]}\phi)\}$, which means the interval contains only the true robustness value. This contradicts our assumption.

Case 2: If $t' - t_1 < b$, then by definition:

$$[\rho](\mathbf{s}_{t_1, t'}, \mathbf{G}_{[a, b]}\phi) = \left[\rho_{\perp}, \min_{t \in [t_1 + a, t_1 + b] \cap [t_1, t']} \rho(\mathbf{s}_{t_1, t}, \phi) \right] \quad (\text{A.12})$$

By definition of standard robustness, $\rho(\mathbf{s}_{t_1, t_2}, \mathbf{G}_{[a, b]}\phi) = \min_{t \in [t_1 + a, t_1 + b]} \rho(\mathbf{s}_{t_1, t}, \phi)$. Since the partial signal agrees with the completion on the observed portion, we have $\rho(\mathbf{s}_{t_1, t_2}, \mathbf{G}_{[a, b]}\phi) \geq \rho_{\perp}$ and $\rho(\mathbf{s}_{t_1, t_2}, \mathbf{G}_{[a, b]}\phi) \leq \min_{t \in [t_1 + a, t_1 + b] \cap [t_1, t']} \rho(\mathbf{s}_{t_1, t}, \phi)$, which contradicts our assumption.

The proof for the AGM robustness interval follows similarly, considering the specific definitions of the AGM interval semantics for the globally operator:

$$[\eta](\mathbf{s}_{t_1, t'}, \mathbf{G}_{[a, b]}\phi) = \begin{cases} \{\eta(\mathbf{s}_{t_1, t'}, \mathbf{G}_{[a, b]}\phi)\} & \text{if } t' - t_1 \geq b \\ [\underline{\eta}, \bar{\eta}] & \text{otherwise} \end{cases} \quad (\text{A.13})$$

where $\underline{\eta}$ and $\bar{\eta}$ are computed based on the observed partial signal and the possible range of future values, ensuring that any completion's AGM robustness falls within this interval.

Eventually ($\mathbf{F}_{[a, b]}\phi$): The proof follows similarly to the globally case, with appropriate modifications for the maximum operation in standard robustness and AGM_{\vee} operation in AGM robustness.

For the eventually operator, the interval semantics is defined as:

$$[\rho](\mathbf{s}_{t_1, t'}, \mathbf{F}_{[a, b]}\phi) = \begin{cases} \{\rho(\mathbf{s}_{t_1, t'}, \mathbf{F}_{[a, b]}\phi)\}; & \text{if } t' - t_1 \geq b \\ [\max_{t \in [t_1 + a, t_1 + b] \cap [t_1, t']} \rho(\mathbf{s}_{t_1, t}, \phi), \rho_{\top}]; & \\ \text{otherwise} & \end{cases} \quad (\text{A.14})$$

The soundness follows by similar contradiction arguments, noting that the true robustness value for any completion must lie within the computed interval bounds. \square

A.3 STL AGM Robustness Interval Chain Inclusion

Proof. The set inclusion property follows from the fact that as more of the signal becomes observed, the set of possible completions becomes more constrained, leading to tighter interval bounds.

Monotonicity: For any formula ϕ , as the partial signal grows from \mathbf{s}_{t_1, t'_1} to \mathbf{s}_{t_1, t'_2} , the interval bounds can only become tighter or remain the same, never become looser. This is

because additional observed values either: provide exact values for subformulae that were previously estimated with intervals, or constrain the possible range of future values based on the observed trend.

Convergence: When $t' \geq \|\phi\|$, the entire time horizon required to evaluate formula ϕ has been observed. At this point, all temporal operators can be evaluated exactly using the observed signal values, no uncertainty remains about future signal values within the formula's time horizon, and the interval semantics reduces to the exact robustness computation

Therefore, $[\rho](\mathbf{s}_{t_1, t'}, \phi) = \{\rho(\mathbf{s}_{t_1, t_2}, \phi)\}$ and $[\eta](\mathbf{s}_{t_1, t'}, \phi) = \{\eta(\mathbf{s}_{t_1, t_2}, \phi)\}$ when $t' \geq \|\phi\|$.

The proof can be formalized by structural induction over ϕ , showing that each operator's interval semantics satisfies the monotonicity and convergence properties. \square

References

- Ahmad, A., Belta, C., and Tron, R. (2022). Adaptive sampling-based motion planning with control barrier functions. In *2022 IEEE 61st Conference on Decision and Control (CDC)*, pages 4513–4518.
- Ahmad, A., Kermanshah, M., Leahy, K., Serlin, Z., Siu, H. C., Mann, M., Vasile, C.-I., Tron, R., and Belta, C. (2025). Accelerating proximal policy optimization learning using task prediction for solving environments with delayed rewards. *Proceedings of Machine Learning Research vol*, 283:1–13.
- Ahmad, A., Liu, S., Tron, R., and Belta, C. (2026). Rrtⁿ: Sampling-based motion planning and control from stl specifications using arithmetic-geometric mean robustness. *arXiv preprint arXiv:2602.16825*.
- Ahmad, A., Vasile, C.-I., Tron, R., and Belta, C. (2023). Robustness measures and monitors for time window temporal logic. *arXiv preprint arXiv:2304.06645*.
- Aksaray, D., Jones, A., Kong, Z., Schwager, M., and Belta, C. (2016). Q-learning for robust satisfaction of signal temporal logic specifications. In *2016 IEEE 55th Conference on Decision and Control (CDC)*, pages 6565–6570. IEEE.
- Alshiekh, M., Bloem, R., Ehlers, R., Könighofer, B., Niekum, S., and Topcu, U. (2018). Safe reinforcement learning via shielding. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32.
- Asarkaya, A. S., Aksaray, D., and Yazıcıoğlu, Y. (2021). Temporal-logic-constrained hybrid reinforcement learning to perform optimal aerial monitoring with delivery drones. In *2021 International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 285–294. IEEE.
- Baier, C. and Katoen, J.-P. (2008). *Principles of model checking*. MIT press.
- Balakrishnan, A. and Deshmukh, J. V. (2019). Structured reward shaping using signal temporal logic specifications. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3481–3486. IEEE.
- Ball, P. J., Smith, L., Kostrikov, I., and Levine, S. (2023). Efficient online reinforcement learning with offline data. In *International Conference on Machine Learning*, pages 1577–1594. PMLR.

- Belta, C., Yordanov, B., and Gol, E. A. (2017). *Formal methods for discrete-time dynamical systems*, volume 15. Springer.
- Bonnah, E. and Hoque, K. A. (2022). Runtime monitoring of time window temporal logic. *IEEE Robotics and Automation Letters*, 7(3):5888–5895.
- Cai, M., Aasi, E., Belta, C., and Vasile, C.-I. (2023). Overcoming exploration: Deep reinforcement learning for continuous control in cluttered environments from temporal logic specifications. *IEEE Robotics and Automation Letters*, 8(4):2158–2165.
- Castro, L. I. R., Chaudhari, P., Tůmová, J., Karaman, S., Frazzoli, E., and Rus, D. (2013). Incremental sampling-based algorithm for minimum-violation motion planning. In *52nd IEEE Conference on Decision and Control*, pages 3217–3224. IEEE.
- Deshmukh, J. V., Donzé, A., Ghosh, S., Jin, X., Juniwal, G., and Seshia, S. A. (2017). Robust online monitoring of signal temporal logic. *Formal Methods in System Design*, 51(1):5–30.
- Donzé, A. and Maler, O. (2010). Robust satisfaction of temporal logic over real-valued signals. In *Formal Modeling and Analysis of Timed Systems: 8th International Conference, FORMATS 2010, Klosterneuburg, Austria, September 8-10, 2010. Proceedings 8*, pages 92–106. Springer.
- Fainekos, G. E. and Pappas, G. J. (2009). Robustness of temporal logic specifications for continuous-time signals. *Theoretical Computer Science*, 410(42):4262–4291.
- Hu, H., Mirchandani, S., and Sadigh, D. (2023). Imitation bootstrapped reinforcement learning. *arXiv preprint arXiv:2311.02198*.
- Hunter, D. R. and Lange, K. (2004). A tutorial on mm algorithms. *The American Statistician*, 58(1):30–37.
- Icarte, R. T., Klassen, T. Q., Valenzano, R., and McIlraith, S. A. (2022). Reward machines: Exploiting reward function structure in reinforcement learning. *Journal of Artificial Intelligence Research*, 73:173–208.
- Kantaros, Y. and Zavlanos, M. M. (2020). Stylus*: A temporal logic optimal control synthesis algorithm for large-scale multi-robot systems. *The International Journal of Robotics Research*, 39(7):812–836.
- Karaman, S. and Frazzoli, E. (2011). Sampling-based algorithms for optimal motion planning. *The International Journal of Robotics Research*, 30(7):846–894.
- Kloetzer, M. and Belta, C. (2008). A fully automated framework for control of linear systems from temporal logic specifications. *IEEE Transactions on Automatic Control*, 53(1):287–297.

- Kobilarov, M. (2012). Cross-entropy motion planning. *International Journal of Robotics Research*, 31(7):855–871.
- Koymans, R. (1990). Specifying real-time properties with metric temporal logic. *Real-time systems*, 2(4):255–299.
- Kress-Gazit, H., Lahijanian, M., and Raman, V. (2018). Synthesis for robots: Guarantees and feedback for robot behavior. *Annual Review of Control, Robotics, and Autonomous Systems*, 1:211–236.
- Kumar, A., Zhou, A., Tucker, G., and Levine, S. (2020). Conservative q-learning for offline reinforcement learning. *Advances in Neural Information Processing Systems*, 33:1179–1191.
- LaValle, S. M. (1998). Rapidly-exploring random trees: A new tool for path planning. Technical Report TR 98–11, Computer Science Department, Iowa State University.
- Leung, K., Aréchiga, N., and Pavone, M. (2020). Backpropagation through signal temporal logic specifications: Infusing logical structure into gradient-based methods. *The International Journal of Robotics Research*, page 02783649221082115.
- Li, X., Vasile, C.-I., and Belta, C. (2017). Reinforcement learning with temporal logic rewards. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3834–3839. IEEE.
- Luo, X., Kantaros, Y., and Zavlanos, M. M. (2021). An abstraction-free method for multirobot temporal logic optimal control synthesis. *IEEE Transactions on Robotics*, 37(5):1487–1507.
- Mabsout, B. E., AbdelGawad, A., and Mancuso, R. (2025). Closing the intent-to-reality gap via fulfillment priority logic. *arXiv preprint arXiv:2503.05818*.
- Majd, K., Yaghoubi, S., Yamaguchi, T., Hoxha, B., Prokhorov, D., and Fainekos, G. (2021). Safe navigation in human occupied environments using sampling and control barrier functions. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5794–5800.
- Maler, O. and Nickovic, D. (2004). Monitoring temporal properties of continuous signals. In *Formal Techniques, Modelling and Analysis of Timed and Fault-Tolerant Systems: Joint International Conferences on Formal Modeling and Analysis of Timed Systems, FORMATS 2004, and Formal Techniques in Real-Time and Fault-Tolerant Systems, FTRTFT 2004, Grenoble, France, September 22-24, 2004. Proceedings*, pages 152–166. Springer.
- Mehdipour, N., Vasile, C.-I., and Belta, C. (2019). Arithmetic-geometric mean robustness for control from signal temporal logic specifications. In *2019 American Control Conference (ACC)*, pages 1690–1695. IEEE.

- Nair, A., Gupta, A., Dalal, M., and Levine, S. (2020). Awac: Accelerating online reinforcement learning with offline datasets. *arXiv preprint arXiv:2006.09359*.
- Neider, D., Gaglione, J.-R., Gavran, I., Topcu, U., Wu, B., and Xu, Z. (2021). Advice-guided reinforcement learning in a non-markovian environment. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 9073–9080.
- Ng, A. Y., Harada, D., and Russell, S. J. (1999). Policy invariance under reward transformations: Theory and application to reward shaping. In *Proceedings of the Sixteenth International Conference on Machine Learning, ICML '99*, page 278–287, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.
- Otte, M. and Frazzoli, E. (2016). Rrtx: Asymptotically optimal single-query sampling-based motion planning with quick replanning. *The International Journal of Robotics Research*, 35(7):797–822.
- Penedo, F., Vasile, C.-I., and Belta, C. (2020). Language-guided sampling-based planning using temporal relaxation. In *Algorithmic Foundations of Robotics XII*, pages 128–143. Springer.
- Perez, A., Platt, R., Konidaris, G., Kaelbling, L., and Lozano-Perez, T. (2012). LQR-RRT*: Optimal sampling-based motion planning with automatically derived extension heuristics. *Proceedings - IEEE International Conference on Robotics and Automation*, (0):2537–2542.
- Pnueli, A. (1977). The temporal logic of programs. In *18th Annual Symposium on Foundations of Computer Science (sfcs 1977)*, pages 46–57. ieee.
- Raman, V., Donzé, A., Maasoumy, M., Murray, R. M., Sangiovanni-Vincentelli, A., and Seshia, S. A. (2014). Model predictive control with signal temporal logic specifications. In *53rd IEEE Conference on Decision and Control*, pages 81–87.
- Ravari, A., Ghoreishi, S. F., and Imani, M. (2024). Implicit human perception learning in complex and unknown environments. In *American Control Conference (ACC)*.
- Rodionova, A., Lindemann, L., Morari, M., and Pappas, G. J. (2021). Time-robust control for stl specifications. In *2021 60th IEEE Conference on Decision and Control (CDC)*, pages 572–579.
- S., J., M., P., Levine, S., Jordan, M., and Abbeel, P. (2015). High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438*.
- Sadigh, D. and Kapoor, A. (2016). Safe control under uncertainty with probabilistic signal temporal logic. In *Proceedings of Robotics: Science and Systems XII*.

- Sadraddini, S. and Belta, C. (2015). Robust temporal logic model predictive control. In *2015 53rd Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 772–779.
- Salzmann, T., I., B., Chakravarty, P., and Pavone, M. (2020). Trajectron++: Dynamically-feasible trajectory forecasting with heterogeneous data. In *European conference on computer vision*, pages 683–700. Springer.
- Schulman, J., Levine, S., Abbeel, P., Jordan, M., and Moritz, P. (2015). Trust region policy optimization. In *International conference on machine learning*, pages 1889–1897. PMLR.
- Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. (2017). Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.
- Sutton, R. S. and Barto, A. G. (2018). *Reinforcement learning: An introduction*. MIT press.
- Towers, M., Terry, J. K., Kwiatkowski, A., Copplestone-Bruce, A., Lutati, G., Dossa, R. F. J., Famularo, A., Pytlak, F., Ballester, M., Guo, H., Hu, J., and Awad, M. (2023). Gymnasium. <https://github.com/Farama-Foundation/Gymnasium>.
- Vasile, C.-I., Aksaray, D., and Belta, C. (2017a). Time window temporal logic. *Theoretical Computer Science*, 691:27–54.
- Vasile, C. I. and Belta, C. (2013). Sampling-based temporal logic path planning. *CoRR*, abs/1307.7263.
- Vasile, C. I. and Belta, C. (2014). Reactive sampling-based temporal logic path planning. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4310–4315.
- Vasile, C. I., Li, X., and Belta, C. (2020). Reactive sampling-based path planning with temporal logic specifications. *The International Journal of Robotics Research*, 39(8):1002–1028.
- Vasile, C. I., Raman, V., and Karaman, S. (2017b). Sampling-based synthesis of maximally-satisfying controllers for temporal logic specifications. *IEEE International Conference on Intelligent Robots and Systems*, 2017-Sept:3840–3847.
- Wolff, E. M., Topcu, U., and Murray, R. M. (2013). Automaton-guided controller synthesis for nonlinear systems with temporal logic. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4332–4339.
- Wu, A., Sadraddini, S., and Tedrake, R. (2020). R3T: Rapidly-exploring Random Reachable Set Tree for Optimal Kinodynamic Planning of Nonlinear Hybrid Systems. *IEEE International Conference on Robotics and Automation (ICRA)*, pages 4245–4251.

- Xu, Z., Gavran, I., Ahmad, Y., Majumdar, R., Neider, D., Topcu, U., and Wu, B. (2020). Joint inference of reward machines and policies for reinforcement learning. In *Proceedings of the International Conference on Automated Planning and Scheduling*, volume 30, pages 590–598.
- Yang, G., Cai, M., Ahmad, A., Prorok, A., Tron, R., and Belta, C. (2025). Lqr-cbf-rrt*: Safe and optimal motion planning. In *2025 American Control Conference (ACC)*, pages 3700–3705. IEEE.
- Yoo, C. and Belta, C. (2015). Control with probabilistic signal temporal logic. *arXiv preprint arXiv:1510.08474*.

CURRICULUM VITAE



