BOSTON UNIVERSITY

GRADUATE SCHOOL OF ARTS AND SCIENCES

A Prospectus for Ph.D. Dissertation

Minimizing the Intent-to-Reality Gap

By

BASSEL EL MABSOUT

Submitted in Partial Fulfillment of the
Requirements for the Degree of
Doctor of Philosophy
In the Department of Computer Science

COMMITTEE MEMBERS:

DR. RENATO MANCUSO *First Reader*

DR. SABRINA NEUMAN

DR. KATE SAENKO

DR. BINGZHUO ZHONG

January 8, 2025

1 Abstract

This dissertation research addresses the intent-to-reality gap in robot learning—the challenge of translating high-level intentions into deployable policies. Practitioners face two key obstacles: expressing their intentions as learning objectives (the intent-to-behavior gap) and maintaining policy behavior when moving from simulation to reality (the sim-to-real gap). Current approaches lack principled structure for specifying objectives and rely on optimization targets that break down during transfer, leading to catastrophic forgetting when policies fail to preserve learned behaviors.

I introduce Expressive Reinforcement Learning as a subfield focused on minimizing the intent-to-behavior gap through structured objective specification and composition. Our early work demonstrated the promise of this design space by developing RE+AL, the first reinforcement learning framework to outperform classical PID controllers on quadrotor attitude control. Countless hours wrestling with brittle reward functions and watching otherwise-promising policies fail in spectacular ways taught us the need for more principled foundations. This led us to develop three complementary frameworks: an Algebraic Q-value Scalarization (AQS) method that minimizes the intent-to-behavior gap through intuitive objective composition (with 600% improvement in sample efficiency), anchor critics for preventing catastrophic forgetting during sim-to-real transfer through multi-objective Q-value optimization, and Conditioning for Action Policy Smoothness (CAPS) that reduces power consumption by 80% while maintaining performance. We further show that efficient deployment is possible through asymmetric actor-critic architectures, reducing model size by up to 99%. Through validation on increasingly complex robotic systems, we show how principled abstractions enable practitioners to directly express and compose their intentions, bridging the gap from intent to reality.

Contents

1 Abstract 2

2 Problem Statement and Objectives 5

3 Background and Literature Review 6

3.1 Standard Fundamentals 6

3.1.1 Markov Decision Processes 6

3.1.2 Policies and Returns 6

3.1.3 Value Functions and Q-Functions 7

3.1.4 Multi-Objective RL 7

3.1.5 Sim-to-Real Transfer 7

3.2 Introduced Conceptual Framework 7

3.2.1 Core Concepts 7

3.2.2 Mathematical Framework 8

3.2.3 Evaluation Metrics 9

3.3 Related Work 9

3.3.1 Objective Specification and Composition 9

3.3.2 Sim-to-Real Transfer 9

3.3.3 Efficient Deployment 10

3.3.4 Open Challenges 10

3.3.5 stub 11

4 Published Results 11

4.1 Foundations: How to Train Your Quadrotor 11

4.2 Resource-Aware Control: CAPS 12

4.2.1 Impact on Real-World Deployment 13

4.3 Efficient Neural Architectures through Actor-Critic Asymmetry 13

5 Current Work 14

5.1 Algebraic Q-value Scalarization (AQS) 14

5.2 Anchor Critics for Robust Transfer 15

6 Proposed Methodology 17

6.1 Structured Objective Specification (AQS) 17

6.1.1 Preliminary Results 17

6.1.2 Future Development 17

6.2 Intention-Preserving Transfer (Anchor Critics) 17

6.2.1 Preliminary Results 17

6.2.2 Future Development 18

6.3 Objective Composition and Validation 18

6.3.1 Theoretical Integration 18

6.3.2 Validation Methods 18

7 Timeline and Milestones 18
7.1 January 2024 18
7.2 February - March 2024 18
7.3 February 2025 19
7.4 March - April 2025 19
Bibliography 19

2 Problem Statement and Objectives

The fundamental challenge in robot learning lies in bridging the intent-to-reality gap—the disparity between a practitioner's intended robot behavior and what is achieved in reality. While reinforcement learning offers powerful tools for developing complex controllers, three critical limitations prevent its widespread adoption in real-world robotics:

First, practitioners struggle to translate their high-level intentions into precise learning objectives. Current approaches rely on brittle linear reward composition and manual tuning, making it difficult to express and balance multiple competing objectives. This intent-to-behavior gap leads to policies that either fail to learn desired behaviors or require prohibitive engineering effort to achieve them.

Second, even when policies perform well in simulation, they often fail to preserve learned behaviors when deployed to real systems. Traditional approaches to sim-to-real transfer treat it as a domain adaptation problem, leading to catastrophic forgetting where policies maintain performance on common scenarios but fail in critical edge cases. This challenge is particularly acute in robotics, where failures in rare but important scenarios can lead to system damage or safety violations.

Third, the computational and energy demands of learned policies often make deployment impractical on real robotic systems. Current methods produce controllers with excessive oscillations that waste energy and stress hardware, while standard neural architectures are unnecessarily complex for deployment. These efficiency challenges compound when attempting to deploy multiple policies or adapt to changing conditions.

We argue that addressing these challenges requires fundamentally rethinking how practitioners express and compose their intentions throughout the learning process. Instead of focusing solely on optimization algorithms, we must develop principled frameworks for:

1. Specifying behavioral objectives in ways that capture intended relationships and trade-offs
2. Preserving learned behaviors during transfer while enabling controlled adaptation
3. Ensuring efficient deployment through resource-aware policy design

Our research addresses these challenges through four interconnected objectives:

1. **Structured Objective Specification:** Develop a principled framework for composing behavioral objectives that enables practitioners to directly express intended relationships and trade-offs. This includes both the mathematical foundations for combining objectives and practical tools for specifying complex behaviors.

2. **Intention-Preserving Transfer:** Create methods that maintain critical behaviors during sim-to-real transfer by treating adaptation as multi-objective optimization over Q-values from both domains. This ensures policies can adapt to reality while preserving behaviors learned in simulation.

3. **Resource-Aware Control:** Design techniques for learning controllers that are efficient in both computation and energy usage. This spans from action smoothness for energy efficiency to architectural optimization for reduced inference cost.
4. **Practical Validation:** Demonstrate the effectiveness of these approaches through systematic evaluation on increasingly complex robotic systems, from quadrotor attitude control to multi-agent scenarios. This includes developing open-source tools and frameworks to enable broader adoption.

Through these objectives, we aim to transform how practitioners develop robotic systems by providing principled ways to express intentions, preserve behaviors, and ensure efficient deployment. The following sections detail our progress toward these goals and outline the remaining work needed to complete this vision.

3 Background and Literature Review

3.1 Standard Fundamentals

3.1.1 Markov Decision Processes

The standard formalization of sequential decision making, defined by a tuple (S, A, P, R, gamma), where S is the state space, A is the action space, $\mathrm{P}(s_{\{t+1\}}|s_t,a_t)$ defines state transition probabilities, $\mathrm{R}(s_t,a_t,s_{\{t+1\}})$ specifies rewards, and γ is a discount factor. In robot learning, states typically represent sensor readings and physical configurations, while actions correspond to motor commands.

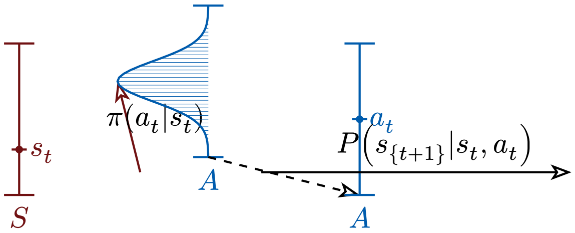

Figure 1: The Markov Decision Process showing how a state s_t in the state space S and action a_t $= \pi(s_t)$ determine the probability distribution $P\left(s_{\{t+1\}}|s_t,a_t\right)$ over next states.

3.1.2 Policies and Returns

A policy $\pi(a_t|s_t)$ defines the probability of taking action a_t in state s_t. The goal in RL is to find a policy that maximizes expected returns, defined as the discounted sum of rewards:

$E\left[\sum \gamma^t R\left(s_t, a_t, s_{\{t+1\}}\right)\right]$. The discount factor γ determines how much to prioritize immediate versus future rewards.

3.1.3 Value Functions and Q-Functions

A value function $V^{\pi(s_t)}$ represents the expected return when following policy π from state s_t. Similarly, a Q-function $Q^{\pi(s_t, a_t)}$ represents the expected return when taking action a_t in state s_t and then following π. These functions are central to our work in two key ways:

1. As a basis for objective composition in AQS, where normalized Q-values enable intuitive specification of behavioral trade-offs
2. As anchors for domain transfer, where simulation Q-values preserve critical behaviors during real-world adaptation

3.1.4 Multi-Objective RL

Multi-objective reinforcement learning extends the standard MDP framework to handle multiple reward signals simultaneously. Instead of a single scalar reward R, the agent receives a vector of rewards (R) corresponding to different objectives. This introduces the challenge of balancing competing objectives and finding Pareto-optimal policies that make principled trade-offs between different goals.

3.1.5 Sim-to-Real Transfer

The process of deploying policies trained in simulation to real-world systems, encompassing both the technical challenges of domain adaptation and the practical constraints of physical deployment. This fundamental concept bridges the gap between idealized training environments and the complexities of real-world operation.

3.2 Introduced Conceptual Framework

3.2.1 Core Concepts

3.2.1.1 The Intent-to-Behavior Gap: The fundamental challenge of translating a practitioner's intended policy behavior into a mathematical specification that reliably produces that behavior when optimized. This gap exists in any reinforcement learning system, as practitioners must convert high-level intentions (like "move smoothly and efficiently") into concrete optimization objectives. Traditional approaches leave practitioners to define scalar rewards and combine objectives through linear scalarization, which often fails to capture the nuanced relationships between different behavioral aspects. More sophisticated specification techniques, such as algebraic composition methods or formal logic frameworks, aim to minimize this gap by providing principled tools for expressing behavioral requirements.

3.2.1.2 The Intent-to-Reality Gap: A compound challenge in robot learning that combines the intent-to-behavior gap (the challenge of specifying desired behaviors through mathematical objectives) with the sim-to-real gap (the challenge of preserving these behaviors during real-world deployment). This gap represents the full distance between a practitioner's intentions and the actual behavior of deployed systems, making it a central challenge in practical robot learning.

3.2.1.3 Expressive Reinforcement Learning: A subfield of reinforcement learning focused on minimizing the intent-to-behavior gap through principled frameworks for objective specification and composition. Key aspects include:

1. Algebraic approaches to objective composition that maintain semantic meaning (e.g., using power-mean operators for intuitive combination of Q-values)
2. Formal methods for specifying policy behaviors (e.g., temporal logic for constraint specification)

The key distinction from traditional multi-objective RL is the focus on providing practitioners with intuitive, mathematically grounded tools for expressing how objectives should be combined, rather than just identifying Pareto-optimal policies.

3.2.1.4 Behavioral Retention: The degree to which a policy maintains its learned behaviors when transferred to new domains or adapted to new conditions. We quantify this through multi-objective evaluation of Q-values across domains, particularly focusing on performance in critical scenarios that may occur rarely but have high importance. This provides a more nuanced view than traditional average-case metrics.

3.2.2 Mathematical Framework

3.2.2.1 Power-Mean as a Logical Operator:

$$M_{p(Q_1,Q_2)} = \left(\frac{Q_1^p+Q_2^p}{2}\right)^{\frac{1}{p}}$$

A fundamental operator for composing objectives, where parameter p controls the logical behavior: as $p \to -\infty$, M_p approaches MIN (AND), and as $p \to \infty$, it approaches MAX (OR). This provides a continuous spectrum between AND and OR operations, enabling flexible composition of objectives while maintaining bounded outputs.

3.2.2.2 Behavioral Objectives: We use this term to encompass both the high-level intentions a practitioner wants to achieve and their formal expression as optimization targets. Our work demonstrates this through several key examples: smooth motor control achieved through temporal and spatial action similarity in CAPS, preservation of critical scenario performance through Q-value anchoring, resource efficiency through architectural optimization, and complex objective composition through algebraic Q-value operations in AQS. This structured approach distinguishes our objectives from simple reward functions, emphasizing their principled nature and composability.

3.2.3 Evaluation Metrics

3.2.3.1 Policy Smoothness: A quantitative measure of a controller's action stability over time, capturing both temporal consistency and spatial coherence. Our work introduces principled metrics for measuring smoothness through frequency analysis of control signals and state-action mapping continuity, enabling objective comparison of different control approaches. This addresses a critical gap in the field where smoothness was previously assessed through ad-hoc or qualitative methods.

3.3 Related Work

This section reviews key developments in reinforcement learning that address the intent-to-reality gap in robot learning, organized around three fundamental challenges: objective specification, sim-to-real transfer, and efficient deployment.

3.3.1 Objective Specification and Composition

Traditional reinforcement learning approaches rely heavily on manual reward engineering [1], which often fails to capture complex behavioral requirements. Several approaches have emerged to address this:

3.3.1.1 Multi-Objective RL: Classical approaches focus on finding Pareto-optimal policies [2] for competing objectives, but often lack intuitive ways to specify trade-offs [3]. These methods typically rely on linear scalarization or constrained optimization, which can be limiting when objectives have complex interactions.

3.3.1.2 Formal Methods: Temporal logic frameworks [4] and propositional logic approaches [5] provide rigorous specifications but can be challenging for practitioners to use effectively. Recent work has explored more accessible formal methods that maintain mathematical rigor while improving usability.

3.3.1.3 Expressive RL: Recent work has explored algebraic approaches to objective composition [3] and structured reward design [6] that maintain semantic meaning. These methods provide practitioners with intuitive tools for specifying how objectives should be combined, going beyond simple linear composition.

3.3.2 Sim-to-Real Transfer

The challenge of transferring policies from simulation to reality remains central to practical robot learning [7]. Current approaches broadly fall into two categories:

3.3.2.1 Environment-Focused Methods: Domain randomization [6] and adaptive architectures [8] attempt to bridge the reality gap through robust training. However, these approaches often struggle with accurately modeling complex real-world dynamics [9].

3.3.2.2 Policy-Focused Methods: While fine-tuning approaches can adapt to real systems, they struggle with catastrophic forgetting [10], where policies maintain high rewards on common scenarios but fail on rare, critical cases [11]. Recent work has shown that structured training environments [7] and multi-objective optimization across domains [12] can help maintain critical behaviors.

3.3.3 Efficient Deployment

Practical deployment requires policies that are efficient in both computation and physical resource usage [4]. Three key areas have emerged:

3.3.3.1 Control Optimization: Explicit regularization [4] and adaptive control methods [13] help maintain smooth and efficient behavior. This builds on classical work in optimal control [14], while addressing the unique challenges of learned policies.

3.3.3.2 Architectural Efficiency: Recent advances in asymmetric actor-critic methods [15] and network compression [16] demonstrate that smaller networks can achieve comparable performance. Liquid neural networks [17] show promise for adaptive control with reduced model complexity.

3.3.3.3 Resource-Aware Design: Joint optimization of computational and physical efficiency [8] has become crucial for practical robotics, particularly in embedded systems [18] where network dependencies can limit autonomy.

3.3.4 Open Challenges

Several fundamental challenges remain in bridging the intent-to-reality gap:

1. **Objective Specification:** While formal methods and expressive frameworks provide tools for composition [3], specifying complex objectives in a way that reliably produces intended behaviors remains difficult [2].
2. **Behavioral Guarantees:** Current approaches lack formal guarantees about behavioral preservation [11], particularly for safety-critical scenarios. This limits deployment in high-stakes applications where performance bounds are required.

This dissertation addresses these challenges through a unified approach that combines expressive objective specification with robust transfer and efficient deployment. Through frameworks like AQS, CAPS, and Anchor Critics, we demonstrate how structured approaches to reinforcement learning can minimize the intent-to-reality gap in practical robot learning.

3.3.5 stub

4 Published Results

Our published work establishes three key foundations for Expressive Reinforcement Learning:

4.1 Foundations: How to Train Your Quadrotor

Our journey began with a fundamental challenge in robotics: achieving reliable low-level control of quadrotor attitude that could outperform classical PID controllers. While PID controllers were straightforward to use, they couldn't learn from experience or adapt to the environment. Previous attempts to apply reinforcement learning to this problem faced two critical issues: poor transfer from simulation to reality (with only one in dozens of trained agents proving controllable on real drones) and unstable, non-smooth control leading to excessive power consumption and hardware wear.

Through careful analysis, we identified a fundamental limitation in how objectives were being specified: the standard practice of linear reward composition required constant manual tuning based on observed behavior, leading to brittle policies that failed to capture true behavioral objectives. This led us to develop RE+AL, which introduced multiplicative reward composition as an alternative that better preserved the intent of each objective, improving behavioral retention (see Section 3.2.1.4) through more robust policy learning.

The results were transformative. We achieved the first successful application of reinforcement learning to outperform classical PID controllers on quadrotor attitude control. The approach yielded a 10× reduction in training time, dropping from 9 hours to under 50 minutes. Additionally, we achieved significant improvements in control signal quality, reducing oscillations from 330Hz to 130Hz while maintaining low tracking errors of 4.2 deg/s.

These improvements directly addressed resource efficiency (see Section 3.3.5) through reduced training time and better control efficiency.

This early work revealed a crucial insight that would shape my subsequent research: the gap between practitioner intent and realized behavior wasn't just about improving simulators—it was about how we specified objectives to the learning system. This realization led me to develop the concepts of intent-to-behavior and intent-to-reality gaps, and the principles of Expressive Reinforcement Learning.

4.2 Resource-Aware Control: CAPS

A critical but often overlooked challenge in deploying learned policies to real robots is the prevalence of oscillatory control responses. While deep reinforcement learning enables training control policies for complex dynamical systems, these policies often exhibit problematic behaviors detrimental to system integrity. These oscillations manifest as:

- Visible physical system oscillations affecting performance
- Increased power consumption and overheating from high-frequency control signals
- Potential hardware failures due to sustained stress

The challenge is particularly acute in continuous control domains, where controller responses can vary infinitely within acceptable output limits. While we can help shape behavior using reward engineering, we identified that certain objectives - like smoothness - can be directly encoded into the policy optimization process itself. This led to a fundamental insight: by explicitly specifying these objectives through policy regularization, we could work in conjunction with reward signals to shape the desired outcomes.

Our Conditioning for Action Policy Smoothness (CAPS) addressed this by directly shaping the neural network's mapping from states to actions through two complementary objectives, implemented as regularization terms:

1. **Temporal smoothness**: Actions should maintain similarity with previous actions to ensure smooth transitions in controller outputs over time
2. **Spatial smoothness**: Similar states should map to similar actions, providing robustness against measurement noise and modeling uncertainties

We formalize these smoothness objectives through regularization terms:

4.2.0.1 Temporal Smoothness Loss:

$$L_{\text{temporal}} = \tfrac{1}{T} \textstyle\sum_{t=1}^{T} \|a_t - a_{t-1}\|_2$$

where L_{temporal} measures the average change in actions over time, encouraging smooth transitions between consecutive control outputs.

4.2.0.2 Spatial Smoothness Loss:

$$L_{\text{spatial}} = E_{s_1,s_2}\left[\tfrac{\|a_1-a_2\|_2}{\|s_1-s_2\|_2}\right]$$

where L_{spatial} measures the Lipschitz continuity of the policy, ensuring similar states map to similar actions.

The total loss combines these terms with the policy objective:

4.2.0.3 CAPS Total Loss:

$$L_{\text{total}} = L_{\text{policy}} + \alpha L_{\text{temporal}} + \beta L_{\text{spatial}}$$

where α and β are hyperparameters controlling the trade-off between objectives.

Key results include:

- 80% reduction in power consumption while maintaining task performance
- 96% improvement in policy smoothness
- Significant reduction in training time
- Successful flight-worthy controllers using simple reward structures

This work demonstrated a key principle: some objectives are better expressed directly through policy structure rather than through reward engineering. By directly encoding smoothness objectives into the policy structure, CAPS addresses both behavioral retention (see Section 3.2.1.4) by ensuring consistent behavior across conditions and resource efficiency (see Section 3.3.5) through reduced power consumption and hardware stress.

4.2.1 Impact on Real-World Deployment

The success of CAPS in achieving smooth control has led to its widespread adoption in real-world applications. Control smoothness proves crucial across multiple dimensions: it extends hardware longevity and reliability, enhances energy efficiency in battery-powered systems, enables safer human-robot interaction, and ensures more predictable behavior during deployment. This work exemplifies how well-structured specifications can simultaneously address multiple aspects of the intent-to-reality gap, from resource constraints to behavioral requirements.

4.3 Efficient Neural Architectures through Actor-Critic Asymmetry

While CAPS addressed energy efficiency through smoother actions, the computational cost of neural network inference remained a significant barrier to deployment. This led us to question a fundamental implicit assumption in actor-critic methods: that actors and critics should share similar neural network architectures.

A key insight emerged when we examined the different objectives these networks need to satisfy: the critic must develop rich representations to understand both system dynamics and reward structures, while the actor simply needs to learn a mapping that maximizes value. This suggested that the representational power needed to satisfy the actor's objectives might be much smaller than what's needed for the critic's objectives - yet the standard practice was to use identical architectures for both components.

Through systematic evaluation across multiple actor-critic algorithms and environments, we demonstrated that:

- Actors can be reduced by up to 99% in size while maintaining performance
- Average reduction of 77% in model parameters across tasks
- Successful validation across 4 popular actor-critic algorithms
- Consistent results across 9 different environments with varying dynamics

This work revealed a fundamental asymmetry in the representational requirements needed to satisfy actor versus critic objectives. This insight directly addresses resource efficiency (see Section 3.3.5) by dramatically reducing computational requirements without compromising behavioral retention (see Section 3.2.1.4).

5 Current Work

5.1 Algebraic Q-value Scalarization (AQS)

A fundamental limitation in reinforcement learning is the reliance on linear reward composition, which often proves brittle and unintuitive when designing desired policy behaviors. While practitioners can theoretically achieve any behavior through careful reward engineering, this process is often time-consuming, requires significant domain expertise, and can lead to suboptimal results. This challenge is particularly acute in real-world applications with multiple competing objectives.

To address these limitations, we developed Algebraic Q-value Scalarization (AQS), a novel domain-specific language that allows practitioners to express how different objectives should interact. Rather than focusing on reward engineering, AQS enables direct specification of when one objective should take priority over another based on their current satisfaction levels. Key innovations include:

1. Using the power-mean as a logical operator over normalized Q-values
2. Q-value scalarization instead of traditional reward scalarization
3. Q-value normalization for stable learning across objectives
4. Integration with a new DDPG-based algorithm called Balanced Policy Gradient (BPG)

This approach fundamentally changes how we express objectives in reinforcement learning. Rather than trying to encode complex behaviors through reward engineering, AQS provides an intuitive language for specifying how objectives should be prioritized. For example, if one objective is nearly satisfied while another is not, AQS can naturally express that the unsatisfied objective should take priority - similar to how an AND operator works in Boolean logic, but generalized to continuous values.

The results demonstrated comprehensive improvements across multiple dimensions. We achieved up to 600% improvement in sample efficiency compared to Soft Actor Critic, along with substantial reductions in policy variability. By providing a more principled way to express objective relationships, AQS enables practitioners to focus on what behaviors they want rather than how to engineer rewards to achieve those behaviors.

This approach improves both behavioral retention (see Section 3.2.1.4), by making objective relationships explicit and verifiable, and resource efficiency (see Section 3.3.5), by enabling more sample-efficient learning through better objective specification.

While previous work like Anchor Critics showed the power of Q-values for preserving behaviors across domains, AQS demonstrates their potential for expressing complex objective relationships. By normalizing Q-values to [0,1], we can treat them as continuous measures of objective satisfaction:

5.1.0.1 Q-value Normalization $Q_{\text{norm}(s,a)} = \frac{Q(s,a)-Q_{\text{min}}}{Q_{\text{max}}-Q_{\text{min}}}$: The normalized Q-value Q_{norm} is bounded to [0,1], representing the relative satisfaction of an objective.

This normalization enables the use of power-mean operators (see Equation 0) to express logical relationships between objectives.

5.2 Anchor Critics for Robust Transfer

A key challenge in robot learning is maintaining policy behavior when transitioning from simulation to reality. While real-world data is essential for adaptation, critical scenarios often occur rarely in practice. This leads to a fundamental problem: policies optimized on real-world data can maintain good average performance while catastrophically forgetting how to handle important edge cases that were learned in simulation.

This challenge directly relates to behavioral retention (see Section 3.2.1.4) - the ability of a policy to maintain its learned behaviors when transferred to new domains. Traditional approaches focus on average-case performance, but fail to preserve behavior in critical scenarios that occur rarely in real data but are crucial for safe operation.

We formalize this challenge through Q-value optimization:

5.2.0.1 Retention Score:

$$R_{\text{retention}} = \min_{s \in S_{\text{critical}}} \left\{ Q_{\text{sim}(s,\pi(s))} \right\}$$

where $R_{\text{retention}}$ quantifies how well the policy preserves behaviors learned in simulation across critical scenarios S_{critical}.

5.2.0.2 Combined Objective:

$$Q_{\text{combined}} = M_{p(Q_{\text{real}}, R_{\text{retention}})}$$

where Q_{combined} represents the overall objective that balances real-world adaptation with behavioral retention.

Building on our experience with CAPS, we also incorporate smoothness objectives multiplicatively:

5.2.0.3 Smoothness Loss:

$$J_{\pi} = M_0 \left(Q_{\pi}, \frac{w_T}{w_T+L_T}, \frac{w_S}{w_S+L_S}, \frac{w_A}{w_A+\pi^{:-1}} \right)$$

where L_T and L_S are temporal and spatial smoothness terms from Equation 0 and Equation 0, Q_π is the critic's Q-value, $\pi^{:-1}$ is the pre-activation policy output, and w_T, w_S, w_A are penalty thresholds. The geometric mean (M_0) ensures all objectives must be satisfied simultaneously while maintaining the [0,1] bounds established in Section 5.1.0.1.

This composition proved more effective for preserving smoothness during transfer, achieving:

- 47% reduction in control jerk compared to additive composition
- 52% lower power consumption on real hardware
- Maintained tracking performance within 5% of simulation baseline

The final objective combines smoothness with our retention score using AQS's power-mean operator:

5.2.0.4 Final Objective:

$$Q_{\text{final}} = M_{p(Q_{\text{combined}}, L_{\text{smooth}})}$$

where Q_{combined} is from Equation 0 and M_p is the power-mean operator from Equation 0. This unifies our three key insights:

- Smoothness through direct policy regularization (CAPS)
- Behavioral retention through Q-value anchoring
- Objective composition through power-mean operators (AQS)

The multiplicative composition creates a natural AND relationship between objectives - both temporal and spatial smoothness must be satisfied simultaneously, aligning with the intuition developed in our AQS work (Equation 0).

Our Anchor Critics method addresses this by treating sim-to-real transfer as multi-objective optimization over Q-values from both domains. By using Q-values from simulation as "anchors", we can preserve important behaviors learned in simulation while still allowing the policy to adapt to reality. This is particularly powerful because simulation allows us to deliberately train on critical scenarios, making those Q-values effective anchors for the behaviors we want to preserve.

Key innovations include:

- Using simulation Q-values as anchors for performance metrics
- Multi-objective optimization to balance adaptation and retention
- Integration with on-board inference for real-time control

Initial results demonstrate the power of this approach:

- 50% reduction in power consumption while maintaining stable flight
- Preservation of critical behaviors while adapting to real-world conditions
- Successful validation in both sim-to-sim and sim-to-real scenarios
- Development of open-source firmware for practical deployment

Importantly, this work addresses both behavioral retention (see Section 3.2.1.4) through Q-value anchoring and resource efficiency (see Section 3.3.5) through reduced power consumption and hardware stress. This demonstrates how properly preserving learned behaviors can simultaneously improve multiple aspects of deployment performance.

This work demonstrates another key principle of expressive reinforcement learning: sometimes objectives are better preserved by directly optimizing them rather than trying to re-learn them through reward engineering.

6 Proposed Methodology

Our proposed work focuses on three key areas that will advance the field of Expressive Reinforcement Learning:

6.1 Structured Objective Specification (AQS)

6.1.1 Preliminary Results

- Novel domain-specific language design
- Power-mean operators and Q-value normalization
- 600% improvement in sample efficiency

6.1.2 Future Development

- Formal semantics and verification
- Safety constraint integration
- Theoretical guarantees

6.2 Intention-Preserving Transfer (Anchor Critics)

6.2.1 Preliminary Results

- Multi-objective Q-value optimization
- Preventing catastrophic forgetting
- Systems Implementation
  - SWANNFlight firmware development
  - On-board inference optimization

6.2.2 Future Development

- System identification integration
- Real-time adaptation strategies
- Hardware-in-the-loop validation

6.3 Objective Composition and Validation

6.3.1 Theoretical Integration

- Unifying Q-value based objectives
  - Compositional properties
  - Theoretical guarantees

6.3.2 Validation Methods

- Behavioral verification metrics
  - Trade-off preservation analysis
  - Compositional property validation

This methodology builds on our preliminary results while advancing both the theoretical foundations and practical applications of expressive reinforcement learning.

7 Timeline and Milestones

The completion of this dissertation involves finalizing two key publications and writing the dissertation itself:

7.1 January 2024

- Submit AQS paper to ICML 2024 (January 30)
  - Complete comparative analysis with recent MORL methods
  - Expand benchmark evaluations
  - Prepare manuscript and code release

7.2 February - March 2024

- Begin dissertation writing

  - Integrate existing published work (RE+AL, CAPS, Actor-Critic)
  - Develop unified theoretical framework
  - Write core chapters

7.3 February 2025

- Submit Anchor Critics paper
  - Refine related works section and introduce the definition of the objectives framework as a contribution
  - Clean up SWANNFlight firmware documentation
  - Prepare manuscript and supplementary materials

7.4 March - April 2025

- Complete dissertation
  - Incorporate all paper contributions
  - Submit dissertation draft to committee
  - Dissertation defense (April)

This timeline builds on our already published work (RE+AL, CAPS, and Actor-Critic papers) while ensuring thorough completion of the remaining publications and dissertation writing.

Bibliography

[1] J. Degrave *et al.*, “Magnetic control of tokamak plasmas through deep reinforcement learning,” *Nature*, vol. 602, pp. 414–419, 2022, doi: 10.1038/s41586-021-04301-9.

[2] L. N. Alegre, A. L. Bazzan, D. M. Roijers, A. Nowé, and B. C. da Silva, “Sample-efficient multi-objective learning via generalized policy improvement prioritization,” *arXiv preprint arXiv:2301.07784*, 2023.

[3] W. Koch, R. Mancuso, and A. Bestavros, “Neuroflight: Next Generation Flight Control Firmware,” *CoRR*, 2019, [Online]. Available: http://arxiv.org/abs/1901.06553

[4] W. Koch, “Flight Controller Synthesis Via Deep Reinforcement Learning,” 2019.

[5] K. Nottingham, A. Balakrishnan, J. Deshmukh, and D. Wingate, “Using logical specifications of objectives in multi-objective reinforcement learning,” *arXiv preprint arXiv:1910.01723*, 2019.

[6] A. Molchanov, T. Chen, W. Hönig, J. A. Preiss, N. Ayanian, and G. S. Sukhatme, “Sim-to-(Multi)-Real: Transfer of Low-Level Robust Control Policies to Multiple Quadrotors,”

International Conference on Intelligent Robots and Systems, 2019, [Online]. Available: http://arxiv.org/abs/1903.04628

[7] W. Koch, R. Mancuso, R. West, and A. Bestavros, "Reinforcement Learning for UAV Attitude Control," *ACM Transactions on Cyber-Physical Systems*, vol. 3, no. 2, 2019, doi: 10.1145/3301273.

[8] J. Hwangbo, I. Sa, R. Siegwart, and M. Hutter, "Control of a Quadrotor With Reinforcement Learning," *IEEE Robotics and Automation Letters*, vol. 2, pp. 2096–2103, 2017.

[9] B. Pasik-Duncan, "Adaptive Control," *IEEE Control Syst.*, vol. 16, p. 87–, 1996.

[10] M. Wolczyk *et al.*, "Fine-tuning Reinforcement Learning Models is Secretly a Forgetting Mitigation Problem." [Online]. Available: https://arxiv.org/abs/2402.02868

[11] K. Binici, N. Trung Pham, T. Mitra, and K. Leman, "Preventing Catastrophic Forgetting and Distribution Mismatch in Knowledge Distillation via Synthetic Data," in *2022 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, 2022, pp. 3625–3633. doi: 10.1109/WACV51458.2022.00368.

[12] F. Muratore, F. Ramos, G. Turk, W. Yu, M. Gienger, and J. Peters, "Robot Learning From Randomized Simulations: A Review," *Frontiers in Robotics and AI*, vol. 9, Apr. 2022, doi: 10.3389/frobt.2022.799893.

[13] M. Schreier, "Modeling and Adaptive Control of a Quadrotor." p. , 2012. doi: 10.1109/ICMA.2012.6282874.

[14] A. R. Mahmood, D. Korenkevych, G. Vasan, W. Ma, and J. Bergstra, "Benchmarking Reinforcement Learning Algorithms on Real-World Robots," *Conference on Robot Learning*, 2018, [Online]. Available: http://arxiv.org/abs/1809.07731

[15] R. Islam, P. Henderson, M. Gomrokchi, and D. Precup, "Reproducibility of benchmarked deep reinforcement learning tasks for continuous control," *arXiv preprint arXiv:1708.04133*, 2017.

[16] S. Han, H. Mao, and W. Dally, "Deep Compression: Compressing Deep Neural Network with Pruning, Trained Quantization and Huffman Coding," *arXiv: Computer Vision and Pattern Recognition*, 2016.

[17] R. Hasani, M. Lechner, A. Amini, D. Rus, and R. Grosu, "Liquid time-constant networks," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2021, pp. 7657–7666.

[18] S. Chinchali *et al.*, "Network offloading policies for cloud robotics: a learning-based approach," *Autonomous Robots*, vol. 45, no. 7, pp. 997–1012, Jul. 2021, doi: 10.1007/s10514-021-09987-4.